Нейроинформатика / А.Н.Горбань, В.Л.Дунин-Барковский, А.Н.Кирдин и др. - Новосибирск: Наука. Сибирское предприятие РАН, 1998. - 296с.

НЕЙРОИНФОРМАТИКА

А.Н.Горбань, В.Л.Дунин-Барковский, А.Н.Кирдин, Е.М.Миркес, А.Ю.Новоходько, Д.А.Россиев, С.А.Терехов, М.Ю.Сенашова, В.Г.Царегородцев

План-проспект с изложением содержания по главам

Введение

НЕЙРОКИБЕРНЕТИКА, НЕЙРОИНФОРМАТИКА, НЕЙРОКОМПЬЮТЕРЫ

В предшествующее десятилетие развитие средств вычислительной техники во всем мире испытало сильное влияние со стороны инициированной Японией программы "Пятое поколение компьютеров". Основным лозунгом этой программы было развитие систем искусственного интеллекта на основе алгоритмических языков. В 1992 г. программа "Пятое поколение" была завершена и ей на смену пришла программа "Вычисления в Реальном мире".

В ней речь идет прежде всего о том, чтобы дать вычислительным и управляющим системам возможность самостоятельно, без помощи "переводчика"-человека воспринимать воздействия внешнего мира и действовать в нем. Авторы программы огромную роль - до 30-40% ее содержания отводят исследованию естественных и созданию искусственных нейросетевых систем.

Искусственные нейронные сети, они же коннекционистские или связевые системы представляют собой устройства, использующие огромное число элементарных условных рефлексов, называемых по имени недавно умершего канадского физиолога синапсами Хебба. Такой синапс, как основу возможных механизмов памяти и поведения, Д.Хебб описал теоретически в 1949 году, т.е. в первые годы после рождения кибернетики. Уже сейчас искусственные нейронные сети применяются для решения очень многих задач обработки изображений, управления роботами и непрерывными производствами, для понимания и синтеза речи, для диагностики заболеваний людей и технических неполадок в машинах и приборах, для предсказания курсов валют и результатов скачек. Та часть работ, которая связана с разработкой устройств переработки информации на основе принципов работы естественных нейронных систем относится к области нейроинформатики или нейровычислений (нейрокомпьютинга). Термины эти появились недавно - в середине 80-х годов.

Суть всех подходов нейроинформатики: разработка методов создания (синтеза) нейронных схем, решающих те или иные задачи. Нейрон при этом выглядит как устройство очень простое: нечто вроде усилителя с большим числом входов и одним выходом. Различие между подходами и методами - в деталях представлений о работе нейрона, и, конечно, в представлениях о работе связей. Собственно, как уже отмечалось выше, устройства нейроинформатики представляют собой связевые системы. В отличие от цифровых микропроцессорных систем, представляющих собой сложные комбинации процессорных и запоминающих блоков, нейропроцессоры содержат память, распределенную в связях между очень простыми процессорами. Тем самым основная нагрузка на выполнение конкретных функций процессорами ложится на

архитектуру системы, детали которой в свою очередь определяются межнейронными связями.

Во введении дан краткий исторический обзор, обсуждаются значения основных понятий и свойства биологических прототипов, основные движущие мотивы исследований в области нейроинформатики.

Введение

НЕЙРОКИБЕРНЕТИКА, НЕЙРОИНФОРМАТИКА, НЕЙРОКОМПЬЮТЕРЫ

В.Л. Дунин-Барковский

НИИ нейрокибернетики им. А.Б. Когана[1]
Ростовского Госуниверситета, Ростов-на-Дону

I. Темп прогресса в науке

В настоящее время все окружающее изменяется необычайно быстро. Разумеется, так было всегда. Единственное различие между прошлым и настоящим состоит в изменении субъекта быстрых изменений. Сами же изменения и быстрый, порой мгновенный рост подчиняются широко известному закону экспоненциального роста в ограниченном объеме. Я напомню читателям в двух словах, в чем состоит этот закон. Он прекрасно иллюстрируется задачей о банке с микробами.

Итак: отдельные особи микробов вида М делятся каждую секунду (одна на две) и находятся в стеклянной (трехлитровой) банке. По условию задачи, если в начальный момент времени в банке находится один микроб, то ровно через одни сутки банка наполнится до краев.

Вопрос задачи: "За какое время банка наполнится наполовину?" Ответ - за одни сутки без одной секунды.

Не укладывается в воображении то, что за одни сутки без 20 секунд банка заполнена лишь на одну миллионную часть. Решение этой задачи можно проиллюстрировать графиком - переход с 0 на 1 происходит в течение 10 секунд, т.е. практически за одну десятитысячную часть суток и на графике выглядит как резкая ступенька.

Знания о механизмах мозга представляют типичный пример экспоненциального роста в ограниченном объеме. Специалисты уже сейчас могут констатировать: нам известно уже очень много: 1/10, 1/1000 или 1/1000000 часть всех механизмов мозга. Время удвоения знаний - год (или 5 лет, что в общем-то не очень существенно). В любом случае мы находимся вблизи той точки во времени, когда практически все о

[1] В настоящее время президент Российской ассоциации нейроинформатики и нерокомпьютинга В.Л.Дунин-Барковский работает в Институте проблем передачи информации РАН, Москва.

работе мозга нам станет известно. Таково мое восприятие "десятилетия мозга", провозглашенного Дж. Бушем в 1990 г. То есть сейчас исследователи мозга, выражаясь языком Колумба, "видят берег". Постараюсь дать читателям хотя бы некоторое представление об очертаниях этого берега.

2. Отступление I: Российская физиология и прогресс мировой науки

Великий Иван Петрович Павлов обнаружил простейший существенный фундаментальный элементарный блок, на использовании которого строится, в основном, работа мозга. Это - элементарный (или, скорее - атомарный) условный рефлекс. Сейчас это бесспорно и признается всей мировой наукой. Но, разумеется, как прогресс в современной физике зиждется на достижениях Ньютона, но отнюдь к ним не сводится, так и прогресс в науке о мозге в существенной степени базируясь на новом понимании достижений Павлова, весьма многосторонен и многопланов.

Не удержусь и помяну недоброй памяти годы, когда - о, ирония судьбы, - в нашей стране утверждалось, что все, кроме условных рефлексов - буржуазное мракобесие (Павловская сессия Академии медицинских наук СССР, 1950 г.). Этот пример показывает, что даже в основе своей верные предпосылки могут отнюдь не вести к храму знаний. Для тех, кто не застал те годы, напомню: в 30-е - 50-е годы "советская физиологическая наука" догматизировала результаты нескольких отечественных столпов науки (условные рефлексы Павлова, диалектику торможения через возбуждение (парабиоз, доминанта и т.д.) Сеченова, Введенского и Ухтомского, фазовую концепцию нервного возбуждения Насонова и Александрова), а все остальное, включая кибернетику, предавала анафеме как "механицизм и метафизику".

Итак, достаточно давно стало ясно, что интеллектуальные процессы идут в коре больших полушарий. Павлов указал на условный рефлекс, как на один из основных "атомов" этих процессов в коре. В российской науке поколение чуть постарше нас (лет на 10-15) в силу только что упомянутых причин разделилось на адептов и сокрушителей идеологии "коры больших полушарий". Эта кора очень
сложна и, как казалось "диссидентам", слишком сложна для того чтобы всерьез надеяться понять детали происходящих в ней нервных процессов. Гораздо больше шансов было разобраться с механизмами работы более просто устроенных структур: спинного мозга, мозжечка, гиппокампа.

Я сформировался как специалист по устройству и работе мозжечка. Надо сказать, что придя студентом МФТИ в 1962 г. в теоретический отдел Института биофизики АН СССР, возглавлявшийся уже тогда великим математиком и ныне здравствующим академиком И.М. Гельфандом, я был всерьез обижен тем, что мне вместо мышления предложили заниматься управлением движениями, а вместо большого мозга (т.е. коры больших полушарий) - мозжечком. Молодость, молодость..., это только в самое последнее время стало понятно, что распространенное (к сожалению) высказывание о футболистах, мол, ноги есть - ума не надо, глубоко неверно. Современные системы искусственного интеллекта прекрасно играют в шахматы, а вот в футбол - научатся нескоро. Словом, интеллектуальная сфера есть только малая надводная часть айсберга мыслительной деятельности. Я не говорю о подсознательном в традиционном фрейдистском понимании (это - отдельный предмет), а о массе процессов переработки огромных объемов поступающей в мозг первичной информации, соотнесении ее с тем, что хранится в мозге, и синтезе сигналов управления огромным числом степеней свободы организма.

Эти информационные процессы, особенно их периферическая часть, связанная непосредственно с сенсорами и преобразователями внешних воздействий в энергию нервных импульсов, а также первичная обработка потоков этих импульсов, - требуют огромных вычислительных ресурсов, имеющихся в нервной системе, и плохо получающихся в технических системах.

Сказанное зафиксировано в названиях и идеологии всемирных современных научно-технических суперпроектов (программ) века.

В предшествующее десятилетие развитие средств вычислительной техники во всем мире испытало сильное влияние со стороны инициированной Японией программы "Пятое поколение компьютеров". Основным лозунгом этой программы было развитие систем искусственного интеллекта на основе алгоритмических языков. В 1992 г. программа "Пятое поколение" была завершена и ей на смену пришла программа "Вычисления в Реальном мире".

3. Real World Computing (RWC)

Я привожу исходное название этой международной программы. Оно возникло и провозглашено в весьма патриотичной и отнюдь не англоязычной Японии, именно в

той форме, как здесь написано, и без перевода понятно научной аудитории любой национальности.

Итак, программа "Вычисления в Реальном мире". Речь идет прежде всего о том, чтобы дать вычислительным и управляющим системам возможность самостоятельно, без помощи "переводчика"-человека воспринимать воздействия внешнего мира и действовать в нем. Авторы программы огромную роль - до 30-40% ее содержания отводят исследованию естественных и созданию искусственных нейросетевых систем.

Искусственные нейронные сети, они же коннекционистские или связевые системы представляют собой устройства, использующие огромное число элементарных условных рефлексов, называемых по имени недавно умершего канадского физиолога синапсами Хебба. Такой синапс, как основу возможных механизмов памяти и поведения, Д.Хебб описал теоретически в 1949 году, т.е. в первые годы после рождения кибернетики. Уже сейчас искусственные нейронные сети применяются для решения очень многих задач обработки изображений, управления роботами и непрерывными производствами, для понимания и синтеза речи, для диагностики заболеваний людей и технических неполадок в машинах и приборах, для предсказания курсов валют и результатов скачек. Я ограничусь перечислением, поскольку подробности займут слишком много места.

Таков один из "слонов", на котором сейчас стоит платформа современного научно-технического прогресса и одно из важнейших его направлений - программа RWC, провозглашенная Японией, как лидером современного научно-технического прогресса (во всяком случае - в отношении производства предметов и товаров в невоенных целях).

4. Нейрокибернетика

На протяжении трех последних десятилетий, раз в 2-4 года со всего Советского Союза съезжались в Ростов-на-Дону специалисты на всесоюзные с международным участием конференции по нейрокибернетике. Бессменным организатором этих конференций был профессор Александр борисович Коган (1912-1989). В 1992 г. прошла десятая конференция этого цикла. Она была посвящена памяти А.Б.Когана, человека, проявившего выдающиеся научные и организаторские способности, позволившие ему не только создать единственный в стране Институт

нейрокибернетики (с 1992 г. институт носит имя своего основателя), но и организовать и скоординировать усилия исследователей "тайн мозга" во всесоюзном масштабе.

Собственно нейрокибернетика представляет собой широкое поле наук и технологий, связанных с пониманием устройства нейронных систем и применением полученных знаний в технике и медицине.

В частности, в Институте нейрокибернетики имеются научно-исследовательские отделы (в скобках указаны руководители подразделений):

- внутриклеточной организации нейрона, включающий лабораторию электронной микроскопии (к.б.н. В.Н.Гусатинский, к.б.н. Г.М.Федоренко);

-организации нейронных сетей, включающий лабораторию самообучающихся нейронных систем (к.б.н. А.Г.Сухов, к.б.н. В.Н.Ефимов);

- организации информационных и управляющих систем мозга (д.б.н. А.А.Буриков);

- организации высших функций мозга человека, включающий лабораторию нейрофизиологических механизмов психической деятельности (к.б.н. А.Э.Тамбиев, д.б.н. В.Н.Кирой);

- моделирования нервных механизмов и робототехники (А.И.Самарин);

- автоматизации медицинских исследований (д.б.н. Г.А.Кураев);

лаборатории:

- улучшения и восстановления функций сенсорных систем (д.б.н. Е.Б.Компанеец);

- физмологии и биохимиии нейропепетидов (д.б.н. А.М.Менджерицкий);

- методов и средств моделирования нейробионических систем и нейрокомпьютеров (Б.А.Финкельштейн);

- нейроинформатики сенсорных и моторных систем (к.б.н. Л.Н.Подладчикова);

- механизмов организации нервных процессов и управления поведением (к.б.н. Е.В.Вербицкий);

- методов анализа экспериментальных данных (д.б.н. Б.М.Владимирский);

а также - группа компьютерной графики (к.б.н. В.Д.Цукерман).

Среди задач, в решении которых в последнее время в Институте нейрокибернетики были существенные достижения: алгоритмы и детали механизмов распознавания полутоновых изображений человеком и животными, системы технического зрения, устройства и методы анализа электрического поля мозга человека, метод и устройство электрической стимуляции кожи века, излечивающие ряд заболеваний, связанных с частичной атрофией сетчатки и зрительного нерва,

программные и аппаратные модели нейронных сетей и процессов обучения в них, выявление отдельных деталей нейронных механизмов памяти и анализа сенсорных сигналов, анализ механизмов и функций сна и гипнотических состояний, межполушарной ассиметрии, разработка методов и приборов психофизиологического естирования и диагностики оптимальных профессиональных склонностей.

5. Нейроинформатика

Та часть работ, которая связана с разработкой устройств переработки информации на основе принципов работы естественных нейронных систем относится к области нейроинформатики или нейровычислений (нейрокомпьютинга). Термины эти появились недавно - в середине 80-х годов. В то же время сравнение электронного и биологического мозга ведется постоянно на протяжении всей истории существования вычислительной техники. Знаменитая книга Н.Винера "Кибернетика", ознаменовавшая рождение этой науки в 1948 г., имеет подзаголовок "Управление в живых системах, технике и обществе".

Новый мощный импульс идея нейро-бионики, т.е. создание технических средств на нейро-принципах, получила в 1982-1984 гг. В это время размеры элементарных деталей вычислительных устройств сравнялось с размерами элементарных "преобразователей информации" в нервной системе. Быстродействие электронных элементов в миллионы раз больше, а эффективность решения задач , особенно связанных с ориентировкой и принятием решений в сложной естественной среде, у биологических систем выше.

Возникло подозрение, что живые организмы выигрывают за счет своих механизмов процессов переработки информации. Резко активировались фундаментальные и прикладные исследования в этой области. Сейчас нейроинформатика как наука уже обладает собственным аппаратом и предметом исследований.

Суть всех подходов нейроинформатики: разработка методов создания (синтеза) нейронных схем, решающих те или иные задачи. Нейрон при этом выглядит как устройство очень простое: нечто вроде усилителя с большим числом входов и одним выходом. Различие между подходами и методами - в деталях представлений о работе нейрона, и, конечно, в представлениях о работе связей. Собственно, как уже отмечалось выше, устройства нейроинформатики представляют собой связевые системы. В отличие

от цифровых микропроцессорных систем, представляющих собой сложные комбинации процессорных и запоминающих блоков, нейропроцессоры содержат память, распределенную в связях между очень простыми процессорами. Тем самым основная нагрузка на выполнение конкретных функций процессорами ложится на архитектуру системы, детали которой в свою очередь определяются межнейронными связями.

Значительную роль в общем подъеме интереса к нейропроблемам сыграла теория, предложенная Джоном Хопфилдом в 1982 г. Она буквально заворожила на продолжительное время физиков-теоретиков. И хотя с точки зрения нейро- теоретиков и технологов эта теория мало что дала, возбужденные ей аналогии и каскады головокружительных теоретических вычислений доставили немало эстетических радостей адептам науки. Кроме того, теория эта представляет хорошее поле для освоения понятий и упражнений, помогающих образовывать и развивать аналитические и моделирующие способности студентов.

Не углубляясь в технические детали, скажу, что благодаря работам Хопфилда возникло мнение, что наряду с ферромагнетиками и антиферромагнетиками возможны нейромагнетики, т.е. теоретические конструкции того же класса, что и первые два, но разительно от них отличающиеся и описывающие только запоминание дискретных образов (или так называемых паттернов информации) в нейронных сетях.

Другой важный класс нейронных систем введен в рассмотрение финном Тейво Кохоненом. У этого класса красивое название: самоорганизующиеся отображения состояний, сохраняющие топологию сенсорного пространства. теория Кохонена активно использует теорию адаптивных систем, развивавшуюся на протяжении многих лет академиком РАН Я.З.Цыпкиным.

Весьма популярна сейчас во всем мире оценка возможностей обучающихся систем, в частности, нейронных сетей (теория размерности Вапника-Червоненкиса). Ее основа была создана еще в 1966 г. советскими исследователями А.Я.Червонекисом и В.Н.Вапником.

Еще один класс нейроподобных моделей представляют сети с обратным распространением ошибок. Метод уходит корнями в довольно отдаленное прошлое. В развитии его современных модификаций ведущую роль сыграли французский исследователь ле Кун и проф. А.Н.Горбань из Красноярска.

Нейроинформатика стала внедряться в окружающую нас действительность в виде нейропроцессоров и нейрокомпьютеров. Перейдем к ним.

6. Отступление 2: нейронная бомба и психотронное оружие

Когда в журнале "Успехи физических наук" стали появляться статьи в стиле теории Хопфилда, т.е. связанные с фазовыми переходами в нейронных системах корректоры упорно исправляли в этих статьях слово нейрон на слово нейтрон. Аналогичным образом - по аллитерации нейрон-нейтрон возникло сочетание "нейронная бомба". Однако связь здесь оказалась в действительности более глубокой. Во всем мире военные проявляют огромный интерес к нейровычислительным системам. В частности, нейросетевые исследования и разработки финансируются в рамках исследовательских программ всех родов войск США: во многих научных журналах опубликованы условия конкурсов на финансирование, проводимых этими ведомствами. Российские военные также проявляют интерес к научным конференциям и выставкам по нейросетевой тематике, проводимым в России. Не исключено, что на вооружении каких-то стран уже имеются нейронные снаряды-комикадзе, чей нейросетевой "интеллект" направлен на уничтожение каких-то конкретных целей. Таким образом нейронные технологии таят в себе определенные потенциальные опасности и должны постоянно находиться в сфере общественного внимания.

Чисто "по смежности" проблематики упомяну и проблему "психотронного оружия". Речь идет о том, что какими-то незаметными электромагнитными или просто слабыми воздействиями можно "зомбировать" большие массы людей. На мой взгляд такая возможность нереальна. Она время от времени муссируется средствами массовой информации и возможно даже и вызывает какую-то реакцию у потенциальных "заказчиков" - разработчиков оружия или противооружия. Но, к счастью, сейчас сколько-нибудь серьезной опасности отсюда не исходит. Если же, - не дай бог, - у серьезных специалистов возникнут опасения, что мечты о психотронном оружии становятся реальностью, то, разумеется, нужны глобальные (на уровне ООН) скоординированные действия мирового сообщества, видимо, такого же масштаба, как действия по защите от ядерной радиации или действия по борьбе с наркотиками.

7. Нейрокомпьютеры: необходима осторожность

Прежде, чем рассказать что-то по существу о современных нейрокомпьютерных устройствах, упомяну еще об одной опасности, "профессионально" связанной с

нейрокомпьютингом. Я бы назвал ее сверх-угрозой халтуры, пожалуй, внутренне присущей этому роду занятий.

На высокую степень опасности этой угрозы для нашей профессиональной области указывает хотя бы на тот факт, что в непосредственной близости от нее процветает область, полностью основанная на - сознательном или бессознательном - жульничестве. Я имею в виду все оттенки парапсихологиии и "космической биоэнергетики". Это, конечно, крайность, но ее следует иметь в виду и постоянно опасаться к ней скатиться.

Время от времени в СМИ (Средства Массовой Информации) появляются подвалы и многополосья о необыкновенных способностях и возможностях нейрокомпьютинга, который можно сделать нашей отечественной сохой за кратчайший срок и - разом вырваться на мировой уровень вперед, далеко опередив мощью своего интеллекта недалеких в общем-то американцев и, тем более, японцев.

Такие психологические атаки бывают успешными не только в СМИ, но и в кабинете ЛПР (Лиц Принимающих Решения). В последнем случае появляются решения, отправляющие в песок очередную огромную порцию общественных средств.

Таков негатив. Что же в позитиве?

В мире имеется несколько десятков специализированных фирм, выпускающих продукцию в области нейроинформатики и, кроме того, многие гиганты индустрии (IBM, Siemence, Mitsubishi и др.) ведут исследования и разработки в этой области. То есть намерения здесь в настоящее время серьезные. Каковы же достигнутые результаты?

Что такое нейрокомпьютер? На самом деле сейчас между понятиями компьютер и нейрокомпьютер примерно такое же соотношение как между понятиями государь и милостивый государь. То есть сейчас любой нейрокомпьютер не претендует на звание компьютера, но создается для решения какого-то фиксированного круга задач. Похоже, что широкие приложения получат устройства, основанные на комбинированных технологиях, включающие по мере необходимости те или иные нейропроцессорные устройства.

Некоторую рекламу и, соответственно, некоторые средства в Японии получили приборы бытовой техники: пылесосы, кондиционеры, электропечки и стиральные машины, использующие нейропроцессоры в устройствах управления.

Некоторое время назад появилась мода на применения искусственных нейронных сетей в финансово-экономических приложениях. Причиной появления моды стали успехи нейросетевых систем в области предсказания будущего. В этой и других задачах

такого рода речь идет о том, чтобы по имеющемуся числовому и событийному ряду предсказать следующие его члены. Нейросетевые конструкции порой решают эту задачу лучше, чем изощренные статистические методики. В чем здесь дело, пока непонятно, но финансистам это не важно.

На основании таких же методов А.Н.Горбань со своим американским аспирантом Ваксманом еще летом 1992 г. предсказал победу Биллу Клинтону на выборах в США. Более того, они могли также дать совет Бушу, какие действия предпринять, чтобы не потерять власть, но Буш их не услышал.

Традиционной областью применения неропроцессоров остаются задачи узнавания изображений, речи, радарных сигналов.

Одно из новых, но впечатляющих приложений - физика высоких энергий (элементарных частиц). В задачах этой науки необходимо в огромном потоке данных от многочисленных датчиков сигналов от элементарных частиц в детекторах ускорителей разного рода, найти комбинации данных, означающих наличие определенных известных или предполагаемых событий. Предварительная обработка информации в этих случаях может быть выполнена нейропроцессорами, "натренированными" методами численного моделирования соответствующих процессов на поиск заданных событий.

И все же основной областью применения нейропроцессоров, скорее всего, станет использование в системах управления и поведения роботами. Глава фирмы, занимающей бесспорно первое место в мире по приложениям нейросетевых систем, автор термина NEUROCOMPUTER, американский профессор из калифорнийского города Сан-Диего Роберт Хехт-Нильсен полагает, что основной продукцией, производимой промышленными фирмами через 10 лет, станут "нейровычислительные роботы" (НВР). В частности, появятся разнообразные работы для выполнения домашней работы (няньки, прачки, кухарки...). Производство НВР быстро превзойдет по объему производство автомобилей и перейдет во все подавляющее производство роботов, производящих роботов...

Представляет такого рода виток спирали опасность человечеству, или нет, я сейчас судить не берусь. Но реальные события этой предполагаемой совершенно новой промышленной революции начнутся лет через пять, т.е. года с 1999 (см. раздел 1 данной статьи).

8. Научно-технический прогресс и целеполагание (вместо заключения)

Нобелевская речь американского физиолога, лауреата нобелевской премии 70-х годов Роджера Сперри меня потрясла тем, что речь в ней шла о... научном коммунизме. Я тогда тоже пришел к этой идее, самостоятельно рассуждая о кошмаре нашего тогдашнего социалистического общества в отблесках пожара, пылавшего на развалинах "Пражской весны социализма с человеческим лицом". Мой товарищ Сергей Адамович Ковалев тогда если не сидел уже в тюрьме, то во всяком случае на всех порах двигался в том направлении. Мне же казалось, что в эпоху глобальных неустойчивостей, порожденных научно-техническим прогрессом, серьезное изменение политической системы нереально. Поэтому было естественно попытаться найти умозрительный хотя бы выход из тупика социализма. Работая младшим научным сотрудником, я, естественно, искал научного выхода. Ну и, конечно, серьезный толчок в этом направлении давали самиздатские "Мысли..." Андрея Дмитриевича Сахарова.

Основная идея моего (совпавшего с мыслями Р.Сперри) научного коммунизма состояла в том, чтобы науку, т.е. добывание новых знаний, поставить в качестве цели существования общества. Естественно, что общество, объединенное некоторой единой целью, является коммунистическим. Оттуда и появилось сочетание "научный коммунизм". Чтобы отличить эту идею от ненавистного марксизма-ленинизма, я употреблял термин "естественно-научный коммунизм" (ЕНК). В упомянутой статье Р.Сперри тоже очень боялся, что его сочтут за сторонника марксизма, и весьма энергично от этого защищался.

Общим местом в наших со Сперри (шутка, конечно) теориях было то, что к целеполаганию общества мы шли со стороны попыток понять устройство мозга. Современный бурный прогресс науки и техники в этом месте, вместе с растущей неустойчивостью глобальных мировых социальных и экологических процессов вводит, мне кажется, идею ЕНК по крайней мере в число идей, достойных общественного внимания.

В ноябре 1993 г. за чашкой чая в небольшой интернациональной компании, состоящей из стажеров университета Хоккайдо, мне довелось провести бурные дебаты "о смысле жизни" с монахом-иезуитом, настоятелем католического храма в префектуре Саппоро гражданином Германии О.Манфредом. Похоже, что мне удалось убедить католического священника в конкурентоспособности естественно-научного целеполагания, по сравнению с религиозным.

Помимо сказанного выше, дискуссия коснулась еще одного вопроса, традиционно решаемого церковью, - проблемы жизни и смерти. В этой проблеме уже давно заметно влияние естественно-научного фактора. С одной стороны - это известные методики замораживания смертельно больных раком и другими неизлечимыми заболеваниями с надеждой на оживление в отдаленном будущем, "когда позволят достижения науки". С другой стороны - здесь известные теоретические представления о том, что информационная основа любой конкретной личности может быть "считана" в компьютерные системы и обрести как бы небиологическую телесную оболочку. Звучит, во всяком случае пока, - несколько фантастично. Тем не менее конкретные технологические задачи, которые здесь вырисовываются, бесспорно относятся к области нейрокибернетики и нейроинформатики, а для их решения должны потребоваться нейрокомпьютеры.

Глава 1.
Возможности нейронных сетей

Для описания алгоритмов и устройств в нейроинформатике выработана специальная "схемотехника", в которой элементарные устройства – сумматоры, синапсы, нейроны и т.п. объединяются в сети, предназначенные для решения задач. *Стандартный формальный нейрон* составлен из входного сумматора, нелинейного преобразователя и точки ветвления на выходе.

Дано описание основных элементов, из которых составляются нейронные сети. Рассматриваются только нейронные сети, синхронно функционирующие в дискретные моменты времени. Описаны основные архитектуры нейронных сетей.

Нейронные сети вычисляют линейные функции, нелинейные функции *одного переменного*, а также всевозможные суперпозиции - функции от функций, получаемые при каскадном соединении сетей. Что можно получить, используя только такие операции? Какие функции удастся вычислить точно, а какие функции можно сколь угодно точно аппроксимировать с помощью нейронных сетей? Даются ответы на эти вопросы, в частности, приведено доказательство теоремы Колмогорова о представлении непрерывных функций многих переменных суперпозициями функций одного переменного и линейных функций, доказана (впервые публикуется на русском языке) обобщенная аппроксимационная теорема Стоуна для произвольных алгебр функций. Впервые в монографической литературе приводятся результаты о плотности полугрупп непрерывных функций.

Подробно обсуждается вопрос о универсальных аппроксимационных способностях нейронных сетей.

Глава 2.
Решение задач нейронными сетями

В данной главе описано несколько базовых задач для нейронных сетей и основных или исторически первых методов настройки сетей для их решения:

1.Классификация (с учителем) (персептрон Розенблатта).

2.Ассоциативная память (сети Хопфилда).

3.Решение систем линейных уравнений (сети Хопфилда и их обобщения).

4.Восстановление пробелов в данных (сети Хопфилда и их обобщения).

5.Кластер-анализ и классификация (без учителя) (сети Кохонена).

Во вводных разделах главы описаны одноэлементные системы и их приложения:

1.Линейная регрессия и восстановление простейших закономерностей;

2.Линейная фильтрация и адаптивная обработка сигналов;

1. Линейное разделение классов и простейшие задачи распознавания образов.

Глава 3.
Быстрое дифференцирование, двойственность и обратное распространение ошибки

В этой главе излагается материал, значение которого для вычислительной математики и всего того, что по-английски именуют “computer sciences”, выходит далеко за пределы нейроинформатики. необходимые вычислительные затраты на поиск их градиента всего лишь в два-три раза превосходят затраты на вычисление одного

значения функции. Это удивительно - ведь координатами вектора градиента служат n частных производных, а затраты на вычисление одной такой производной в общем случае примерно такие же, как и на вычисление значения функции.

“Чудо” объясняется довольно просто: нужно рационально организовать вычисление производных сложной функции многих переменных, избегая дублирования. Для этого необходимо подробно представить само вычисление функции, чтобы потом иметь дело не с “черным ящиком”, преобразующим вектор аргументов в значение функции, а с детально описанным графом вычислений.

Поиск grad$H$ удобно представить как некоторый двойственный процесс над структурой вычисления H. Промежуточные результаты, появляющиеся при вычислении градиента, являются ни чем иным, как множителями Лагранжа. Оказывается, что если представить H как сложную функцию, являющуюся суперпозицией функций малого числа переменных (а по-другому вычислять функции многих переменных мы не умеем), и аккуратно воспользоваться правилом дифференцирования сложной функции, не производя по дороге лишних вычислений и сохраняя полезные промежуточные результаты, то вычисление всей совокупности $\partial H/\partial x_i$ (i=1,...,n) немногим сложнее, чем одной из этих функций - они все собраны из одинаковых блоков.

Детально со всеми доказательствами описан принцип двойственности для сложных функций общего вида, а также для нейронных сетей.

Глава 4.
Нейросетевые информационные модели сложных инженерных систем

В данной главе обсуждаются нейросетевые методы построения моделей сложных систем, основанные на экспериментальных данных. Подробно рассмотрены постановки типовых задач информационного моделирования (прямых, обратных и смешанных). Изложение сопровождается модельными иллюстрациями и примерами реальных практических применений.

Внутренние регуляризирующие особенности нейронных сетей позволяют решать также обратные и комбинированные задачи *с локальной оценкой точности*. Для некорректно поставленных задач моделирования предложена нейросетевая информационная технология построения гибридной нейроархитектуры, содержащей кластеризующую карту Кохонена и семейство сетей с обратным распространением, обучаемых на данных индивидуальных кластеров. В этой технологии выявляются области частичной корректности задачи, в которых дается решение с высокой локальной точностью. Для остальных областей признакового пространства нейросеть автоматически *корректно* отвергает пользовательские запросы.

В главе рассмотрены примеры применения методики решения обратных задач к моделированию отклика сложной инженерной системы - промышленного контейнера с ядерными отходами - на внешние аномальные условия (тепловая нагрузка вследствие пожара). Результаты исследований могут быть использованы для технических рекомендаций.

Глава 5.
Медицинская нейроинформатика

Анализ применения персональных ЭВМ в медицинских учреждениях показывает, что наибольшее использование компьютеров идет для обработки

текстовой документации, хранения и обработки баз данных, ведения статистики и финансовых расчетов. Отдельный парк ЭВМ используется совместно с различными диагностическими и лечебными приборами.

Многолетние исследования, проводимые с самыми различными явными алгоритмами, показали, что медицинские задачи, имеющие неявный характер, решаются явными методами с точностью и удобством, совершенно недостаточными для широкого практического использования в конкретных задачах диагностики, прогнозирования и принятия решений. Неявные задачи медицины и биологии явились идеальным полем для применения нейросетевых технологий, и именно в этой области наблюдается наиболее яркий практический успех нейроинформационных методов.

В главе дан обзор более 20 нейроэкспертных систем для диагностики заболеваний, предсказания из исходов, оптимизации лечения. Подробно описаны следующие системы:

1. Прогнозирование осложнений инфаркта миокарда.
2. Система назначения оптимальной стратегии лечения больных облитерирующим тромбангиитом и прогнозирования его непосредственных исходов.
3. Система дифференциальной диагностики “острого живота”.
4. Нейросети для изучения иммунореактивности

Обсуждаются достоинства и недостатки нейросетевого подхода к построению медицинских систем. Описаны способы подбора и подготовки медицинских данных для обучения нейросетевых систем. Подробна описана технология построения таких систем, суммирован отечественный и зарубежный опыт их создания и эксплуатация. Большинство результатов впервые публикуется в монографической литературе.

Глава 6.
Погрешности в нейронных сетях

Рассматриваются нейронные сети слоистой структуры, состоящие из слоев стандартных нейронов. Изучаются ошибки, возникающие при технической реализации сетей, а также при шумах и повреждениях.

Определены максимально допустимые погрешности, возможные для сигналов и параметров каждого элемента сети, исходя из условия, что вектор выходных сигналов сети должен вычисляться с заданной точностью. Используются два типа оценок погрешности: гарантированные интервальные оценки и среднеквадратические оценки погрешностей.

Показано, что оценки допустимых погрешностей можно получить в ходе специального процесса “обратного распространения точности”. Он состоит в функционировании сети с той же системой связей, но от выходов к входам и с заменой элементов на двойственные. Эта двойственность принципиально отличается от той, которая используется в классическом методе вычисления градиентов оценки с помощью обратного распространения ошибок (back propagation of errors).

С помощью полученных результатов объясняется наблюдаемая высокая устойчивость нейронных сетей к шумам и разрушениям.

Глава 7.
Скрытые параметры и транспонированная регрессия

Решается классическая проблема восстановления недостающих данных в следующей постановке: найти для каждого объекта наилучшую формулу, выражающую его признаки через признаки других объектов (которых должно быть по возможности меньше). Эта формула должна быть инвариантна относительно смены шкал измерения. Инвариантность достигается тем, что решение представляется в виде суперпозиции однородных дробно - линейных функций.

Строится отношение "объект - опорная группа объектов". Опорная группа выделена тем, что по признакам ее элементов наилучшим образом восстанавливаются признаки исходного объекта. Решение дается с помощью нейронной сети специальной архитектуры. Предлагается способ минимизации опорной группы, использующий преимущества нейросетевого подхода.

Метод транспонированной регрессии применяется к задаче интерполяции свойств химических элементов. Исследуется точность интерполяции потенциалов ионизации химических элементов при помощи транспонированной линейной регрессии. Достигнутая точность позволяет предсказать отсутствующие в справочной литературе значения высших (с 5-го по 10-й) потенциалов ионизации для элементов с атомными номерами от 59-го до 77-го и рекомендовать метод для интерполяции иных физических и химических свойств элементов и соединений.

Глава 8.
Нейронные сети ассоциативной памяти

Рассматриваются нейронные сети ассоциативной памяти, восстанавливающие по искаженному и/или зашумленному образу ближайший к нему эталонный. Исследована информационная емкость сетей и предложено несколько путей ее повышения, в том числе - ортогональные тензорные (многочастичные) сети. Построены способы предобработки, позволяющие конструировать нейронные сети ассоциативной памяти для обработки образов, инвариантной относительно групп преобразований. Описан численный эксперимент по использованию нейронных сетей для декодирования различных кодов.

Глава 9.
Логически прозрачные нейронные сети и производство явных знаний из данных

Производство явных знаний из накопленных данных - проблема, которая намного старше чем компьютеры. Обучаемые нейронные сети могут производить из данных скрытые знания: создается навык предсказания, классификации, распознавания образов и т.п., но его логическая структура обычно остается скрытой от пользователя. Проблема проявления (контрастирования) этой скрытой логической структуры решается в работе путем приведения нейронных сетей к специальному “логически прозрачному” разреженному виду.

Исследуются два вопроса, встающие перед каждым исследователем, решившим использовать нейронные сети: “Сколько нейронов необходимо для решения задачи?” и “Какова должна быть структура нейронной сети?” Объединяя эти два вопроса, мы получаем третий: “Как сделать работу нейронной сети понятной для пользователя (логически прозрачной) и какие выгоды может принести такое понимание?”

Описаны способы получения логически прозрачных нейронных сетей и алгоритмы получения явных знаний из данных. Приведен пример из области социально-политических предсказаний. Для определенности рассматриваются только нейронные сети, обучение которых есть минимизация оценок (ошибок) с использованием градиента. Градиент оценки вычисляется методом двойственности (его частный случай - метод обратного распространения ошибки).

Предисловие

Информатика стремительно меняет свое лицо - только успевай приспосабливаться. Развивается все: и возможности компьютеров растут, и новые программные продукты открывают целый мир ранее недоступных интеллектуальных услуг, и меняются стили программирования - объектный подход, визуальное программирование и прочая, и прочая, и прочая...

Нейроинформатика - один из новых ликов информатики. Это область науки и интеллектуальной практики, переживающая период экспоненциального роста: растет число вовлеченных людей и публикаций, журналов и лабораторий, вложений и изобретений.

Чем это кончится? Поживем - увидим. А пока будем работать сами и изучать чужие результаты, чтобы не отстать, не остаться на перроне, глядя вслед уходящему поезду научно-технического прогресса.

Вот уже четыре года подряд в Красноярске в первую пятницу октября собираются десятки специалистов из разных городов на Всероссийский семинар «Нейроинформатика и ее приложения». Интерес к семинару растет, все чаще научные руководители приезжают на него с аспирантами и молодыми сотрудниками. Одновременно с семинаром уже дважды проводились Всероссийские школы.

В предлагаемой книге собрано несколько основных лекций Всероссийской школы «Нейроинформатика-96» и докладов 4 Всероссийского семинара. Яркая вводная лекция Президента Российской ассоциации нейроинформатики профессора Виталия Львовича Дунина-Барковского, три фундаментальных лекции автора ряда книг и классических результатов профессора Александра Николаевича Горбаня. Эти известные ученые и сопредседатели Оргкомитета открывают книгу. Авторы двух других лекций - молодые, но очень заметные в российской нейроинформатике ученые - С. А. Терехов и Д. А. Россиев. Они оба лидеры: зав. лабораторией знаменитого ВНИИТФ (под Челябинском) Сергей Александрович Терехов многое сделал в разработке технических приложений

нейронных сетей, а тридцатилетний доктор медицинских наук, зав. сектором медицинской нейроинформатики Красноярской медицинской академии Дмитрий Анатольевич Россиев - безусловный лидер в области медицинской нейроинформатики. Их лекции, посвященные, соответственно, техническим и медицинским приложениям нейронных сетей, служат хорошим введением в проблему.

Остальные главы более специальны. Их авторы, в основном, молоды - как и сама нейроинформатика. Оригинальные алгоритмы оценки погрешностей в нейронных сетях, анализ проблемы скрытых параметров и новый подход к восстановлению неизвестных данных (и даже целая таблица предсказаний отсутствующих в справочниках потенциалов ионизации), описание различных обобщений сетей Хопфилда, предназначенных для создания ассоциативной памяти - все это важно для понимания нейроинформатики.

Особое место занимает последняя глава. Ее автор - создатель многих известных в России нейропрограмм Е.М. Миркес - описывает методы производства знаний из данных с помощью нейронных сетей. Это - одна из самых старых проблем науки, старше, чем компьютеры, чем информатика, и, тем более, чем нейроинформатика. Оказывается, что обучаемые нейронные сети даже на персональном компьютере могут производить (с помощью пользователя) нетривиальные теоретические построения. Модельный пример простой, но важной политологической теории, построенный на основе данных о президентских выборах в США, убеждает хотя бы в том, что работать с такими средствами интересно!

Несколько слов о том, как лучше пользоваться предлагаемой книгой. Она рассчитана на две категории читателей. Если Вы математик или физик-теоретик, либо просто имеете вкус к теоретическому мышлению, то Вы получите много удовольствия, прорабатывая первые три лекции - шаг за шагом, страницу за страницей. После этого можно ознакомиться с прекрасным изложением различных аспектов приложений нейронных сетей (главы 4 и 5), не особенно входя в детали, уделить повышенное внимание шестой главе, где изложен

красивый алгоритм оценки погрешностей в работе нейронных сетей, и поискать интересующие Вас вопросы в остальных главах.

Если же Вас больше интересуют приложения нейронных сетей, то надо (не особенно углубляясь) познакомиться с первыми тремя главами, разбирая в первую очередь определения, примеры, формулировки теорем и алгоритмов. Доказательства можно (а при первом чтении - нужно) пропускать. Вам адресованы четвертая и пятая главы - читайте их, по мере необходимости обращаясь к первым трем. И опять же, пробегите остальные главы в поисках интересного для Вас материала.

Ежегодный семинар поддерживается Красноярским краевым фондом науки. Это издание также подготовлено с его помощью.

Ответственный редактор,
доктор физ.-мат. наук, профессор Е.А. Новиков

Глава 1.
Возможности нейронных сетей

А.Н.Горбань

Вычислительный центр СО РАН в г.Красноярске[1]

1. Нейробум: поэзия и проза нейронных сетей

В словах «искусственные нейронные сети» слышатся отзвуки фантазий об андроидах и бунте роботов, о машинах, заменяющих и имитирующих человека. Эти фантазии интенсивно поддерживаются многими разработчиками нейросистем: рисуется не очень отдаленное будущее, в котором роботы осваивают различные виды работ, просто наблюдая за человеком, а в более отдаленной перспективе – человеческое сознание и личность перегружаются в искусственную нейронную сеть – появляются шансы на вечную жизнь.

Поэтическая игра воображения вовлекает в работу молодежь, поэзия рекламы создает научную моду и влияет на финансовые вложения. Можете ли Вы четко различить, где кончается бескорыстная творческая игра и начинается реклама? У меня такое однозначное различение не получается: это как вопрос о искренности – можно сомневаться даже в своей собственной искренности.

Итак: игра и мода как важные движущие силы.

В словах «модное научное направление» слышится нечто неоднозначное - то ли пренебрежение, смешанное с завистью, то ли еще что-то. А вообще, мода в науке – это хорошо или плохо? Дадим три ответа на этот вопрос.

1. Мода – это хорошо! Когда в науке появляется новая мода, тысячи исследователей, грустивших над старыми темами, порядком надоевшими еще со времени писания диссертации, со свежим азартом бросаются в дело. Новая мода позволяет им освободиться от личной истории.

Мы все зависим от своего прошлого, от привычных дел и привычных мыслей. Так давайте же приветствовать все, что освобождает нас от этой

[1] 660036, Красноярск-36, ВЦК СО РАН. E-mail: amse@cckr.krasnoyarsk.su

зависимости! В новой модной области почти нет накопленных преимуществ – все равны. Это хорошо для молодежи.

2. Мода – это плохо! Она противоречит глубине и тщательности научного поиска. Часто "новые" результаты, полученные в погоне за модой, суть всего-навсего хорошо забытые старые, да еще нередко и перевранные. Погоня за модой растлевает, заставляет переписывать старые работы и в новой словесной упаковке выдавать их за свои. Мода - источник сверххалтуры. Примеров тому – тысячи.

"Гений – это терпение мысли." Так давайте же вслед за Ньютоном и другими Великими культивировать в себе это терпение. Не будем поддаваться соблазну моды.

3. Мода в науке – это элемент реальности. Так повелось во второй половине XX века: наука стала массовой и в ней постоянно вспыхивают волны моды. Можно ли относиться к реальности с позиций должного: так, дескать, должно быть, а этак – нет? Наверное, можно, но это уж точно непродуктивно. Волны моды и рекламные кампании стали элементом организации массовой науки и с этим приходится считаться, нравится нам это или нет.

Нейронные сети нынче в моде и поэтическая реклама делает свое дело, привлекает внимание. Но стоит ли следовать за модой? Ресурсы ограничены – особенно у нас, особенно теперь. Все равно всего на всех не хватит. И возникают вопросы:

1) нейрокомпьютер – это интеллектуальная игрушка или новая техническая революция?
2) что нового и полезного может сделать нейрокомпьютер?

За этими вопросами скрыты два базовых предположения:

1) на новые игрушки, даже высокоинтеллектуальные, средств нет;
2) нейрокомпьютер должен доказать свои новые возможности – сделать то, чего не может сделать обычная ЭВМ, – иначе на него не стоит тратиться.

У энтузиастов есть свои рекламные способы отвечать на заданные вопросы, рисуя светлые послезавтрашние горизонты. Но все это в будущем. А сейчас? Ответы парадоксальны:

1) нейрокомпьютеры – это новая техническая революция, которая приходит к нам в виде интеллектуальной игрушки (вспомните – и персональные ЭВМ были придуманы для игры!);
2) для любой задачи, которую может решить нейрокомпьютер, можно построить более стандартную специализированную ЭВМ, которая решит ее не хуже, а чаще всего – даже лучше.

Зачем же тогда нейрокомпьютеры? Вступая в творческую игру, мы не можем знать, чем она кончится, иначе это не Игра. Поэзия и реклама дают нам фантом, призрак результата, погоня за которым - важнейшая часть игры. Столь же призрачными могут оказаться и прозаичные ответы - игра может далеко от них увести. Но и они необходимы - трудно бегать по облакам и иллюзия практичности столь же важна, сколь и иллюзия величия. Вот несколько вариантов прозаичных ответов на вопрос «зачем?» - можно выбрать, что для Вас важнее:

А. Нейрокомпьютеры дают стандартный способ решения многих нестандартных задач. И неважно, что специализированная машина лучше решит один класс задач. Важнее, что один нейрокомпьютер решит и эту задачу, и другую, и третью – и не надо каждый раз проектировать специализированную ЭВМ – нейрокомпьютер сделает все сам и почти не хуже.

Б. Вместо программирования – обучение. Нейрокомпьютер учится – нужно только формировать учебные задачники. Труд программиста замещается новым трудом – учителя (может быть, надо сказать – тренера или дрессировщика). Лучше это или хуже? Ни то, ни другое. Программист предписывает машине все детали работы, учитель – создает "образовательную среду", к которой приспосабливается нейрокомпьютер. Появляются новые возможности для работы.

В. Нейрокомпьютеры особенно эффективны там, где нужно подобие человеческой интуиции – для распознавания образов (узнавания лиц, чтения рукописных текстов), перевода с одного естественного языка на другой и т.п. Именно для таких задач обычно трудно сочинить явный алгоритм.

Г. Гибкость структуры: можно различными способами комбинировать простые составляющие нейрокомпьютеров – нейроны и связи между ними. За счет этого на одной элементной базе и даже внутри "тела" одного нейрокомпьютера можно создавать совершенно различные машины. Появляется еще одна новая профессия – "нейроконструктор" (конструктор мозгов).

Д. Нейронные сети позволяют создать эффективное программное обеспечение для высокопараллельных компьютеров. Для высокопараллельных машин хорошо известна проблема: как их эффективно использовать – как добиться, чтобы все элементы одновременно и без лишнего дублирования вычисляли что-нибудь полезное? Создавая математическое обеспечения на базе нейронных сетей, можно для широкого класса задач решить эту проблему.

Если перейти к еще более прозаическому уровню повседневной работы, то нейронные сети - это всего-навсего сети, состоящие из связанных между собой простых элементов - формальных нейронов. Значительное большинство работ по нейроинформатике посвящено переносу различных алгоритмов решения задач на такие сети.

Ядром используемых представлений является идея о том, что нейроны можно моделировать довольно простыми автоматами, а вся сложность мозга, гибкость его функционирования и другие важнейшие качества определяются связями между нейронами. Каждая связь представляется как совсем простой элемент, служащий для передачи сигнала. Предельным выражением этой точки зрения может служить лозунг: "структура связей – все, свойства элементов – ничто".

Совокупность идей и научно-техническое направление, определяемое описанным представлением о мозге, называется коннекционизмом (по-ангийски connection – связь). Как все это соотносится с реальным мозгом? Так же, как

карикатура или шарж со своим прототипом-человеком - весьма условно. Это нормально: важно не буквальное соответствие живому прототипу, а продуктивность технической идеи.

С коннекционизмом тесно связан следующий блок идей:

1) однородность системы (элементы одинаковы и чрезвычайно просты, все определяется структурой связей);

2) надежные системы из ненадежных элементов и "аналоговый ренессанс" – использование простых аналоговых элементов;

3) "голографические" системы – при разрушении случайно выбранной части система сохраняет свои полезные свойства.

Предполагается, что система связей достаточно богата по своим возможностям и достаточно избыточна, чтобы скомпенсировать бедность выбора элементов, их ненадежность, возможные разрушения части связей.

Коннекционизм и связанные с ним идеи однородности, избыточности и голографичности еще ничего не говорят нам о том, как же такую систему научить решать реальные задачи. Хотелось бы, чтобы это обучение обходилось не слишком дорого.

На первый взгляд кажется, что коннекционистские системы не допускают прямого программирования, то есть формирования связей по явным правилам. Это, однако, не совсем так. Существует большой класс задач: нейронные системы ассоциативной памяти, статистической обработки, фильтрации и др., для которых связи формируются по явным формулам. Но еще больше (по объему существующих приложений) задач требует неявного процесса. По аналогии с обучением животных или человека этот процесс мы также называем обучением.

Обучение обычно строится так: существует задачник – набор примеров с заданными ответами. Эти примеры предъявляются системе. Нейроны получают по входным связям сигналы – "условия примера", преобразуют их, несколько раз обмениваются преобразованными сигналами и, наконец, выдают ответ – также набор сигналов. Отклонение от правильного ответа штрафуется. Обучение состоит в минимизации штрафа как (неявной) функции связей. Примерно

четверть нашей книги состоит в описании техники такой оптимизации и возникающих при этом дополнительных задач.

Неявное обучение приводит к тому, что структура связей становится "непонятной" – не существует иного способа ее прочитать, кроме как запустить функционирование сети. Становится сложно ответить на вопрос: "Как нейронная сеть получает результат?" – то есть построить понятную человеку логическую конструкцию, воспроизводящую действия сети.

Это явление можно назвать "логической непрозрачностью" нейронных сетей, обученных по неявным правилам. В работе с логически непрозрачными нейронными сетями иногда оказываются полезными представления, разработанные в психологии и педагогике, и обращение с обучаемой сетью как с дрессируемой зверушкой или с обучаемым младенцем – это еще один источник идей. Возможно, со временем возникнет такая область деятельности – "нейропедагогика" – обучение искусственных нейронных сетей.

С другой стороны, при использовании нейронных сетей в экспертных системах на PC возникает потребность прочитать и логически проинтерпретировать навыки, выработанные сетью. В главе 9 описаны служащие для этого методы контрастирования – получения неявными методами логически прозрачных нейронных сетей. Однако за логическую прозрачность приходится платить снижением избыточности, так как при контрастировании удаляются все связи кроме самых важных, без которых задача не может быть решена.

Итак, очевидно наличие двух источников идеологии нейроинформатики. Это представления о строении мозга и о процессах обучения. Существуют группы исследователей и научные школы, для которых эти источники идей имеют символическое, а иногда даже мистическое или тотемическое значение. Но это все поэзия, а мы пока перейдем к прозе - к изучению формальных нейронов.

2. Элементы нейронных сетей

Для описания алгоритмов и устройств в нейроинформатике выработана специальная "схемотехника", в которой элементарные устройства – сумматоры, синапсы, нейроны и т.п. объединяются в сети, предназначенные для решения задач.

Интересен статус этой схемотехники – для многих начинающих кажется неожиданным, что ни в аппаратной реализации нейронных сетей, ни в профессиональном программном обеспечении все эти элементы вовсе не обязательно реализуются как отдельные части или блоки. Используемая в нейроинформатике идеальная схемотехника представляет собой особый язык для представления нейронных сетей и их обуждения. При программной и аппаратной реализации выполненные на этом языке описания переводятся на языки другого уровня, более пригодные для реализации.

Самый заслуженный и, вероятно, наиболее важный элемент нейросистем – это *адаптивный сумматор*. Адаптивный сумматор вычисляет скалярное произведение вектора входного сигнала x на вектор параметров α. На схемах будем обозначать его так, как показано на рис. 1. Адаптивным называем его из-за наличия вектора настраиваемых параметров α. Для многих задач полезно иметь линейную неоднородную функцию выходных сигналов. Ее вычисление также можно представить с помощью адаптивного сумматора, имеющего n+1 вход и получающего на 0-й вход постоянный единичный сигнал (рис. 2).

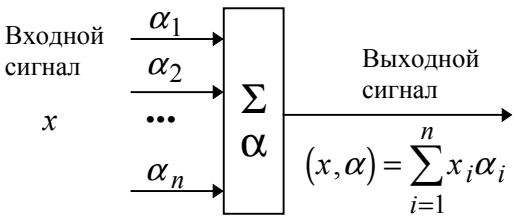


Рис. 1. Адаптивный сумматор.

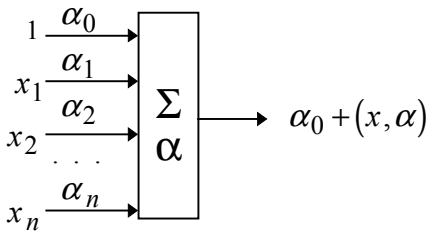


Рис. 2. Неоднородный адаптивный сумматор

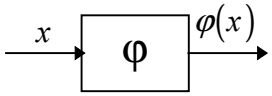


Рис. 3. Нелинейный преобразователь сигнала.

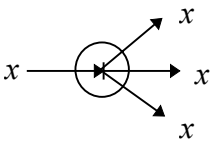


Рис. 4. Точка ветвления

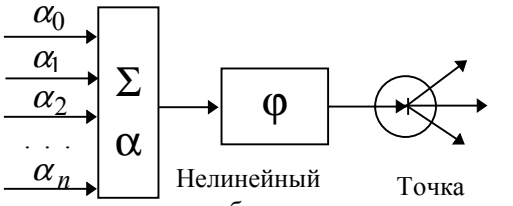


Рис. 5. Формальный нейрон

Нелинейный преобразователь сигнала изображен на рис. 3. Он получает скалярный входной сигнал x и переводит его в $\varphi(x)$.

Точка ветвления служит для рассылки одного сигнала по нескольким адресам (рис. 4). Она получает скалярный входной сигнал x и передает его всем своим выходам. *Стандартный формальный нейрон* составлен из входного сумматора, нелинейного преобразователя и точки ветвления на выходе (рис. 5).

Линейная связь - синапс - отдельно от сумматоров не встречается, однако для некоторых рассуждений бывает удобно выделить этот элемент (рис. 6). Он умножает входной сигнал x на «вес синапса» α.

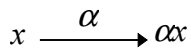


Рис. 6. Синапс.

Также бывает полезно «присоединить» связи не ко входному сумматору, а к точке ветвления. В результате получаем элемент, двойственный адаптивному сумматору и называемый «*выходная звезда*». Его выходные связи производят умножение сигнала на свои веса.

Итак, дано описание основных элементов, из которых составляются нейронные сети.

3. Архитектура нейронных сетей

Перейдем теперь к вопросу: как можно составлять эти сети? Строго говоря, как угодно, лишь бы входы получали какие-нибудь сигналы. Но такой произвол слишком необозрим, поэтому используют несколько стандартных архитектур, из которых путем вырезания лишнего или (реже) добавления строят большинство используемых сетей.

Сначала следует договориться о том, как будет согласована работа различных нейронов во времени. Как только в системе возникает более одного элемента, встает вопрос о синхронности функционирования. Для привычных нам всем программных имитаторов нейронных сетей на цифровых ЭВМ такого вопроса нет только из-за свойств основного компьютера, на котором реализуются нейронные сети. Для других способов реализации такой вопрос весьма важен. Все же здесь и далее рассматриваются только нейронные сети, синхронно функционирующие в дискретные моменты времени: все нейроны срабатывают «разом».

В зоопарке нейронных сетей можно выделить две базовых архитектуры – *слоистые и полносвязные сети.*

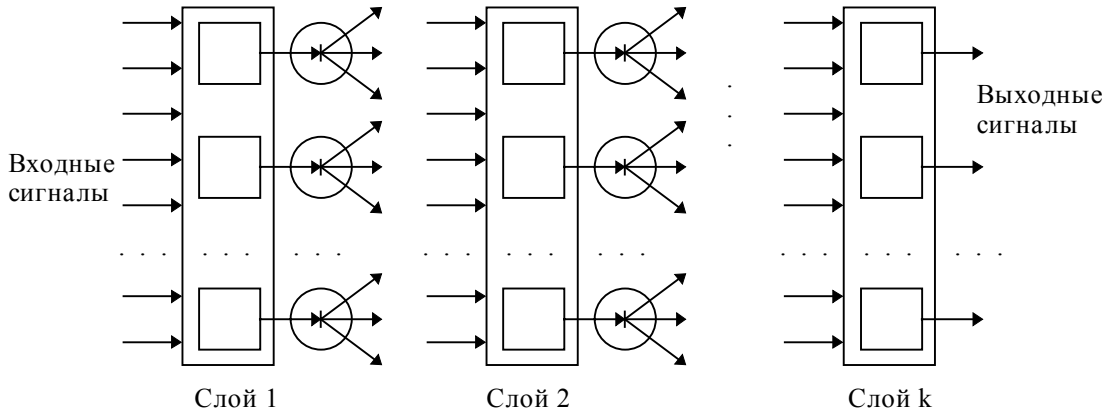


Рис. 7. Слоистая сеть

Слоистые сети: нейроны расположены в несколько слоев (рис. 7). Нейроны первого слоя получают входные сигналы, преобразуют их и через точки ветвления передают нейронам второго слоя. Далее срабатывает второй слой и т.д. до *k*-го слоя, который выдает выходные сигналы для интерпретатора и пользователя. Если не оговорено противное, то каждый выходной сигнал *i*-го слоя подается на вход всех нейронов *i*+1-го. Число нейронов в каждом слое может быть любым и никак заранее не связано с количеством нейронов в других слоях. Стандартный способ подачи входных сигналов: все нейроны первого слоя получают каждый входной сигнал. Особое распространение получили

трехслойные сети, в которых каждый слой имеет свое наименование: первый – входной, второй – скрытый, третий – выходной.

Полносвязные сети: каждый нейрон передает свой выходной сигнал остальным нейронам, включая самого себя. Выходными сигналами сети могут быть все или некоторые выходные сигналы нейронов после нескольких тактов функционирования сети. Все входные сигналы подаются всем нейронам.

Элементы слоистых и полносвязных сетей могут выбираться по-разному. Существует, впрочем, стандартный выбор – нейрон с адаптивным неоднородным линейным сумматором на входе (рис. 5).

Для полносвязной сети входной сумматор нейрона фактически распадается на два: первый вычисляет линейную функцию от входных сигналов сети, второй – линейную функцию от выходных сигналов других нейронов, полученных на предыдущем шаге.

Функция активации нейронов (характеристическая функция) φ – нелинейный преобразователь, преобразующий выходной сигнал сумматора (см. рис. 5) – может быть одной и той же для всех нейронов сети. В этом случае сеть называют *однородной (гомогенной)*. Если же φ зависит еще от одного или нескольких параметров, значения которых меняются от нейрона к нейрону, то сеть называют *неоднородной (гетерогенной).*

Составление сети из нейронов стандартного вида (рис. 5) не является обязательным. Слоистая или полносвязная архитектуры не налагают существенных ограничений на участвующие в них элементы. Единственное жесткое требование, предъявляемое архитектурой к элементам сети, это соответствие размерности вектора входных сигналов элемента (она определяется архитектурой) числу его входов.

Если полносвязная сеть функционирует до получения ответа заданное число тактов k, то ее можно представить как частный случай k-слойной сети, все слои которой одинаковы и каждый из них соответствует такту функционирования полносвязной сети.

Существенное различие между полносвязной и слоистой сетями возникает тогда, когда число тактов функционирования заранее не ограничено – слоистая сеть так работать не может.

4. Существуют ли функции многих переменных ?

Вопрос, вынесенный в заголовок раздела, естественно возникает при изучении возможностей нейронных сетей. Из схем предыдущих разделов видно, что нейронные сети вычисляют линейные функции, нелинейные функции *одного переменного*, а также всевозможные суперпозиции - функции от функций, получаемые при каскадном соединении сетей. Что можно получить, используя только такие операции? Какие функции удастся вычислить точно, а какие функции можно сколь угодно точно аппроксимировать с помощью нейронных сетей? Чтобы изучить возможности нейронных сетей, нужно ответить на эти вопросы.

Заданный вопрос имеет очень большую историю и заведомо старше, чем исследования искусственных нейронных сетей.

Какие функции может вычислять человек? Если мы умеем складывать и умножать числа, то мы можем точно вычислять многочлены и рациональные функции (отношения многочленов) с рациональными коэффициентами от рациональных же аргументов.

Можно, однако, задавать функции с помощью уравнений. Если считать решения нескольких простых уравнений известными, то класс вычисляемых функций расширится - решения некоторых более общих уравнений удастся выразить через эти, более простые функции.

Классический пример: если использовать радикалы - решения уравнений $x^n=a$, то можно явно получить решения произвольных уравнений 2-й, 3-й и 4-й степеней. Так, функция 3-х переменных a, b, c - решение уравнения $ax^2+bx+c=0$ - может быть точно выражена с помощью сложения, умножения, деления и функции одного переменного - квадратного корня.

Вопрос: можно ли представить решение любого алгебраического уравнения с помощью радикалов, был окончательно и отрицательно решен Абелем и Галуа - уже уравнения 5-й степени неразрешимы в радикалах.

Все же можно подбирать другие простые функции небольшого числа переменных - сложнее, чем радикалы, но проще, чем общие решения уравнений высоких степеней. Удастся ли с помощью этих функций построить решение любого уравнения? Вопрос был настолько важен, что Гильберт в списке своих проблем, которые, по его мнению, должны были определять развитие математики XX века, под номером 13 поместил следующую задачу:

Представляется ли корень уравнения

$$x^7+ax^3+bx^2+cx+1=0$$

(как функция коэффициентов) суперпозицией каких-либо непрерывных функций двух переменных?

Для уравнений 5-й и 6-й степени такое представление возможно не только с помощью непрерывных, но даже аналитических функций.

Оказалось полезным абстрагироваться от уравнений и поставить общий вопрос: можно ли произвольную непрерывную функцию *n* переменных получить с помощью операций сложения, умножения и суперпозиции из непрерывных функций двух переменных? Ответ оказался положительным! В серии работ [1-3] А.Н.Колмогоров, затем В.И.Арнольд и вновь А.Н.Колмогоров решили эту проблему: можно получить любую непрерывную функцию *n* переменных с помощью операций сложения, умножения и суперпозиции из непрерывных функций *одного* переменного.

Последняя теорема А.Н.Колмогорова [3] из этой серии настолько проста и изящна, что мы чуть позже приведем ее целиком. А пока - несколько замечаний о условиях теоремы.

От условия непрерывности можно отказаться - тогда получится довольно тривиальный результат связанный, по существу, с равномощностью отрезка и куба любой размерности. Условие непрерывности нельзя значительно усилить: существуют аналитические функции многих переменных, которые не допускают

представления с помощью суперпозиции аналитических функций двух переменных. Более того, все *l* раз непрерывно дифференцируемые функции трех переменных нельзя представить в виде суперпозиций функций двух переменных, каждая из которых дифференцируема [2*l*/3] раз и все частные производные которых порядка [2*l*/3] удовлетворяют условию Липшица (выражение [2*l*/3] означает целую часть числа 2*l*/3). Это доказано А.Г,Витушкиным [4].

А теперь - **теорема Колмогорова**, завершившая серию исследований для непрерывных функций:

Каждая непрерывная функция n переменных, заданная на единичном кубе n-мерного пространства, представима в виде

$$f(x_1, x_2, \ldots, x_n) = \sum_{q=1}^{2n+1} h_q \left[\sum_{p=1}^{n} \varphi_q^p(x_p) \right], \qquad (1)$$

где функции $h_q(u)$ *непрерывны, а функции* $\varphi_q^p(x_p)$*, кроме того, еще и стандартны, т.е. не зависят от выбора функции f.*

В частности, каждая непрерывная функция двух переменных *x, y* представима в виде

$$f(x, y) = \sum_{q=1}^{5} h_q \left[\varphi_q(x) + \psi_q(y) \right]. \qquad (2)$$

Доказательство настолько просто, изящно и поучительно, что мы приведем его практически полностью для случая n=2, следуя изложению В.И.Арнольда [5]. Возможность представления (2) доказывается в несколько этапов.

1. "Внутренние" функции $\varphi_q(x)$ и $\psi_q(y)$ представления (2) совершенно не зависят от разлагаемой функции $f(x, y)$.

Для определения этих функций нам понадобятся некоторые предварительные построения. Рассмотрим изображенный на рис. 8 "город" – систему одинаковых "кварталов" (непересекающихся замкнутых квадратов), разделенных узкими "улицами" одной и той же ширины. Уменьшим гомотетично наш "город" в N раз; за центр гомотетии можно принять, например, точку A_1 – мы получим новый" город", который будем называть "городом ранга 2". "Город

ранга 3" точно также получается из "города ранга 2" гомотетичным уменьшением с коэффициентом гомотетии $\frac{1}{N}$; "город ранга 4" получается гомотетичным уменьшением в N раз "города ранга 3" и т.д. Вообще "город ранга k" получается из исходного "города" (который мы будем называть "городом первого ранга") гомотетичным уменьшением в N^k раз (с центром гомотетии в A_1; впрочем, выбор центра гомотетии не существенен для дальнейшего).

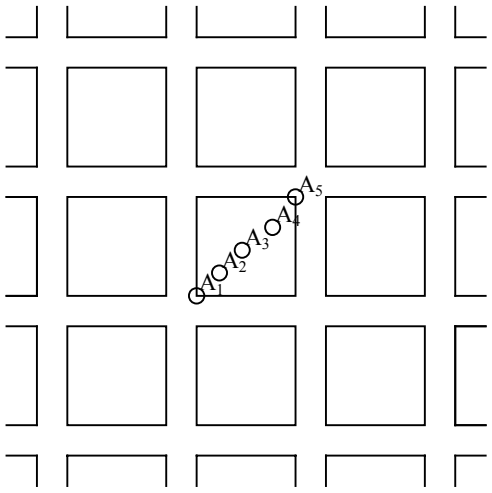


Рис. 8. Система кварталов.

Построенную систему "городов" мы назовем 1-й системой. "Город первого ранга q-й системы" ($q = 2,\ldots,5$) получается из изображенного на рис. 8 "города" при помощи параллельного переноса, совмещающего точку A_1 с точкой A_q. Нетрудно понять, что "улицы" "города" можно выбрать настолько узкими, что каждая точка плоскости будет покрыта по крайней мере тремя кварталами наших пяти "городов первого ранга". Точно так же "город k-го ранга" q-й системы ($k = 2,3,\ldots; q = 2,\ldots,5$) получается из "города k-го ранга 1-й системы" параллельным переносом, переводящим точку A_1^k в точку A_q^k, где A_1^k и A_q^k получаются из точек A_1 и A_q гомотетией, переводящей "город первого ранга" 1-й системы (т.е. наш исходный "город") в "город k-го ранга" той же системы; при этом *каждая точка плоскости будет принадлежать кварталам по крайней мере трех из пяти "городов" любого фиксированного ранга k*.

Функцию

$$\Phi_q(x,y) = \varphi_q(x) + \psi_q(y) \ \ (q = 1,2,\ldots,5)$$

мы определим теперь так, чтобы она разделяла любые два "квартала" каждого "города" системы q, т.е. чтобы множество значений, принимаемых $\Phi_q(x,y)$ на определенном "квартале" "города k-го ранга" (здесь k – произвольное фиксированное число) q-й системы, не пересекалось с множеством значений, принимаемых $\Phi_q(x,y)$ на любом другом "квартале" того же "города". При этом нам, разумеется, будет достаточно рассматривать функцию $\Phi_q(x,y)$ на единичном квадрате (а не на всей плоскости).

Для того, чтобы функция $\Phi_q(x,y)=\varphi_q(x)+\psi_q(y)$ разделяла "кварталы" "города первого ранга", можно потребовать, например, чтобы $\varphi_q(x)$ на проекциях "кварталов" "города" на ось x весьма мало отличалась от различных целых чисел, а $\psi_q(y)$ на проекциях “кварталов” на ось y весьма мало отличалась от различных кратных $\sqrt{2}$ (ибо $m+n\sqrt{2}=m'+n'\sqrt{2}$ при целых m,n,m',n', лишь если $m=m', n=n'$). При этом наложенные условия не определяют пока еще, разумеется, функций $\varphi_q(x)$ и $\psi_q(y)$ (на "улицах" функция $\Phi_q(x,y)=\varphi_q(x)+\psi_q(y)$ вообще пока может задаваться совершенно произвольно); используя это, можно подобрать границы значений $\varphi_q(x)$ и $\psi_q(y)$ на "кварталах" "города второго ранга" так, чтобы функция $\Phi_q(x,y)=\varphi_q(x)+\psi_q(y)$ разделяла не только "кварталы" "города 1-го ранга", но и “кварталы” "города 2-го ранга". Намеченную программу можно осуществить, если N достаточно велико (так что кварталы последующих рангов не соединяют кварталы предыдущих). А.Н. Колмогоров выбрал $N=18$. Привлекая подобным же образом к рассмотрению "города" последующих рангов и уточняя каждый раз значения функций $\varphi_q(x)$ и $\psi_q(y)$, мы в пределе получим непрерывные функции $\varphi_q(x)$ и $\psi_q(y)$ (можно даже потребовать, чтобы они были монотонными), удовлетворяющие поставленным условиям.

2. Функции $h_q(u)$ разложения (2), напротив того, существенно зависят от исходной функции $f(x,y)$.

Для построения этих функций докажем прежде всего, что *любую непрерывную функцию* $f(x,y)$ *двух переменных* x *и* y, *заданную на единичном квадрате, можно представить в виде*

$$f(x,y)=\sum_{q=1}^{5} h_q^{(1)}(\Phi_q(x,y))+f_1(x,y) \tag{3}$$

где $\Phi_q(x,y)=\varphi_q(x)+\psi_q(y)$ *– функции, построенные выше, и*

$$M_1=\max\left|f_1(x,y)\right|\le\frac{5}{6}\max\left|f(x,y)\right|=\frac{5}{6}M, \tag{3а}$$

$$\max\left|h_q^{(1)}(\Phi_q(x,y))\right|\le\frac{1}{3}M,\ q=1,\dots,5. \tag{3б}$$

Выберем ранг k столь большим, чтобы колебание (т.е. разность наибольшего и наименьшего значений) функции $f(x,y)$ на каждом "квартале" любого из "городов ранга k" не превосходило $\frac{1}{6}M$; это, разумеется, возможно, так как с ростом ранга k размеры " кварталов" уменьшаются неограниченно. Далее, пусть $p_1^{(ij)}$ – определенный "квартал" "города 1-й системы" (и выбранного ранга k); в таком случае (непрерывная) функция $\Phi_1(x,y)$ принимает на этом "квартале" значения, принадлежащие определенному сегменту $\Delta_1^{(ij)}$ числовой оси (причем в силу определения функции Φ_1 этот сегмент не пересекается с сегментами значений, принимаемых Φ_1 на всех других "кварталах"). Положим теперь функцию $h_1^{(1)}$ на сегменте $\Delta_1^{(ij)}$ постоянной, равной $\frac{1}{3}$ значения, принимаемого функцией $f(x,y)$ в какой-либо (безразлично какой) внутренней точке $M_1^{(ij)}$ квартала $p_1^{(ij)}$ (эту точку можно назвать "центром квартала"). Таким же образом мы определим функцию $h_1^{(1)}$ на любом другом из сегментов, задаваемых значениями функции $\Phi_1(x,y)$ на "кварталах" "города k-го ранга" 1-й системы; при этом все значения $h_1^{(1)}$ будут

по модулю не превосходить $\frac{1}{3}M$ (ибо значение $f(x,y)$ в "центре" любого "квартала" по модулю не превосходит M). Доопределим теперь функцию $h_1^{(1)}(u)$ при тех значениях аргумента u, при каких она еще не определена, произвольно, с тем лишь, чтобы она была непрерывна и чтобы выполнялось неравенство ($3^б$); совершенно аналогично определим и все остальные функции $h_q^{(1)}(u)$ $(q=2,\ldots,5)$.

Докажем теперь, что разность

$$f_1(x,y)=f(x,y)-\sum_{q=1}^{5}h_q^{(1)}(\Phi_q(x,y))$$

удовлетворяет условию ($3^а$), т.е. что

$$\left|f_1(x_0,y_0)\right|\le\frac{5}{6}M,$$

где (x_0,y_0) – произвольная точка единичного квадрата. Эта точка (как и все точки плоскости) принадлежит по крайней мере трем кварталам "городов ранга k"; поэтому заведомо найдутся такие три из пяти функций $h_1^{(1)}(\Phi_q(x,y))$, которые принимают в точке (x_0,y_0) значение, равное $\frac{1}{3}$ значения $f(x,y)$ в "центре" соответствующего "квартала", т.е. отличающееся от $\frac{1}{3}f(x_0,y_0)$ не более чем на $\frac{1}{18}M$ (ибо колебание $f(x,y)$ на каждом квартале не превосходит $\frac{1}{6}M$); сумма этих трех значений $h_q^{(1)}(\Phi_q(x_0,y_0))$ будет отличаться от $f(x_0,y_0)$ по модулю не более чем на $\frac{1}{6}M$. А так как каждое из оставшихся двух чисел $h_q^{(1)}(\Phi_q(x_0,y_0))$ в силу (3) по модулю не превосходит $\frac{1}{3}M$ то мы получаем:

$$\left|f_1(x_0,y_0)\right|=\left|f(x_0,y_0)-\sum_{q=1}^{5}h_q^{(1)}(\Phi_q(x_0,y_0))\right|\le\frac{1}{6}M+\frac{2}{3}M=\frac{5}{6}M,$$

что и доказывает ($3^а$).

Применим теперь то же разложение (3) к входящей в (3) функции $f_1(x,y)$; мы получим:

$$f_1(x,y)=\sum_{q=1}^{5}h_q^{(2)}(\Phi_q(x,y))+f_2(x,y)$$

или

$$f(x,y)=\sum_{q=1}^{5}h_q^{(1)}(\Phi_q(x,y))+\sum_{q=1}^{5}h_q^{(2)}(\Phi_q(x,y))+f_2(x,y),$$

где

$$M_2=\max\left|f_2(x,y)\right|\le\frac{5}{6}M_1=\left(\frac{5}{6}\right)^2M$$

и $\max\left|h_q^{(2)}(\Phi_q(x,y))\right|\le\frac{1}{3}M_1\le\frac{1}{3}\cdot\frac{5}{6}M\ \ (q=1,2,\ldots,5).$

Затем мы применим разложение (3) к полученной функции $f_2(x,y)$ и т.д.; после n-кратного применения этого разложения мы будем иметь:

$$f(x,y)=\sum_{q=1}^{5}h_q^{(1)}(\Phi_q(x,y))+\sum_{q=1}^{5}h_q^{(2)}(\Phi_q(x,y))+f_2(x,y)+\ldots$$

$$\ldots+\sum_{q=1}^{5}h_q^{(n-1)}(\Phi_q(x,y))+f_n(x,y),$$

где

$$M_n=\max\left|f_n(x,y)\right|\le\left(\frac{5}{6}\right)^nM$$

и

$$\max\left|h_q^{(s)}(\Phi_q(x,y))\right|\le\frac{1}{3}\cdot\left(\frac{5}{6}\right)^{s-1}M\ \ (q=1,2,\ldots,5;\ s=1,2,\ldots,n-1).$$

Последние оценки показывают, что при $n\to\infty$ получим:

$$f(x,y)=\sum_{q=1}^{5}h_q^{(1)}(\Phi_q(x,y))+\sum_{q=1}^{5}h_q^{(2)}(\Phi_q(x,y))+f_2(x,y)+\ldots$$

$$\ldots+\sum_{q=1}^{5}h_q^{(n)}(\Phi_q(x,y))+\ldots$$

где стоящий справа бесконечный ряд сходится равномерно; также и каждый из пяти рядов

$$h_q^{(1)}(\Phi_q(x,y))+h_q^{(2)}(\Phi_q(x,y))+h_q^{(n)}(\Phi_q(x,y))+\ldots\ (q=1,2,\ldots,5)$$

сходится равномерно, что позволяет ввести обозначения

$h_q(u) = h_q^{(1)} + h_q^{(2)} + ... + h_q^{(n)} + ...$ $(q = 1,2,\ldots,5)$.

Итак, окончательно получаем:

$$f(x,y) = \sum_{q=1}^{5} h_q(\Phi_q(x,y)) = \sum_{q=1}^{5} h_q\left[\varphi_q(x) + \psi_q(y)\right],$$

то есть требуемое разложение (2).

До сих пор речь шла о *точном представлении* функций многих переменных с помощью функций одного переменного. Оказалось, что в классе непрерывных функций такое представление возможно. Но кроме вопроса о точном представлении существует еще один - об *аппроксимации*. Можно даже предположить, что он важнее - вычисление большинства функций производится приближенно даже при наличии «точных» формул.

Приближение функций многочленами и рациональными функциями имеет историю, еще более давнюю, чем проблема точного представления. Знаменитая **теорема Вейерштрасса** утверждает, что непрерывную функцию нескольких переменных $f(x_1, x_2, ..., x_n)$ на замкнутом ограниченном множестве Q можно равномерно приблизить последовательностью полиномов: для любого $\varepsilon>0$ существует такой многочлен $P(x_1, x_2, ..., x_n)$, что

$$\sup_{Q} |f(x_1, x_2, ..., x_n) - P(x_1, x_2, ..., x_n)| < \varepsilon.$$

Чтобы сформулировать обобщения и усиления теоремы Вейерштрасса, необходимо перейти к несколько более абстрактному языку. Рассмотрим компактное пространство X и алгебру $C(X)$ непрерывных функций на X с вещественными значениями.

Сильным обобщением теоремы о возможности равномерного приближения непрерывных функций многочленами является **теорема Стоуна** [6,7]:

Пусть $E \subseteq C(X)$ - замкнутая подалгебра в $C(X)$, $1 \in E$ и функции из E разделяют точки в X (то есть для любых различных $x,y \in X$ существует такая функция $g \in E$, что $g(x) \neq g(y)$). Тогда $E=C(X)$.

Теорема Стоуна обобщает теорему Вейерштрасса по двум направлениям. Во-первых, рассматриваются функции на произвольном компакте, а не только

функции многих действительных переменных. Во-вторых, доказано утверждение, новое даже для функций одного переменного (не говоря уже о многих): плотно не только множество многочленов от координатных функций, но вообще кольцо многочленов от любого набора функций, разделяющих точки. Следовательно, плотно множество тригонометрических многочленов, множество линейных комбинаций функций вида $\exp[-(x-x_0,Q(x-x_0))]$, где (x,Qx) - положительно определенная квадратичная форма и др.

Дан рецепт конструирования таких обобщений: достаточно взять произвольный набор функций, разделяющих точки, построить кольцо многочленов от них - и получим плотное в $C(X)$ множество функций.

Разложения по ортогональным системам функций (ряды Фурье и их многочисленные обобщения) не дают, вообще говоря, равномерного приближения разлагаемых функций - как правило, можно гарантировать лишь монотонное стремление к нулю интеграла квадрата остатка «функция минус приближение» с какой-либо положительной весовой функцией. Все же, обращаясь к задаче аппроксимации, нельзя забывать об ортогональных разложениях. Для ряда прикладных задач простота получения коэффициентов такого разложения может оказаться важнее, чем отсутствие гарантированной равномерности приближения.

Так существуют ли функции многих переменных? В каком-то смысле - да, в каком-то - нет. Все непрерывные функции многих переменных могут быть получены из непрерывных функций одного переменного с помощью линейных операций и суперпозиции. Требования гладкости и аналитичности существенно усложняют вопрос. На этом фоне совершенно неожиданно выглядит тот факт, что *любой многочлен от многих переменных может быть получен из одного произвольного нелинейного многочлена от одного переменного с помощью линейных операций и суперпозиции.* Простое доказательство этой теоремы будет дано в разделе 6.

5. Универсальные аппроксимационные способности произвольной нелинейности и обобщенная теорема Стоуна

В этом разделе для множеств непрерывных функций, замкнутых относительно любой нелинейной операции (а не только для колец), доказана обобщенная аппроксимационная теорема Стоуна. Это интерпретируется как утверждение о универсальных аппроксимационных возможностях произвольной нелинейности: с помощью линейных операций и каскадного соединения можно из произвольного нелинейного элемента получить устройство, вычисляющее любую непрерывную функцию с любой наперед заданной точностью.

Рассмотрим компактное пространство X и алгебру $C(X)$ непрерывных функций на X с вещественными значениями.

Кроме аппроксимации функций многочленами и их обобщениями из колец функций, разделяющих точки, в последнее время все большее внимание уделяется приближению функций многих переменных с помощью линейных операций и суперпозиций функций одного переменного. Такое приближение осуществляется специальными формальными "устройствами" – нейронными сетями. Каждая сеть состоит из формальных нейронов. Нейрон получает на входе вектор сигналов x, вычисляет его скалярное произведение на вектор весов α и некоторую функцию одного переменного $\varphi(x,\alpha)$. Результат рассылается на входы других нейронов или передается на выход. Таким образом, нейронные сети вычисляют суперпозиции простых функций одного переменного и их линейных комбинаций.

Доказан ряд теорем [8-10] об аппроксимации непрерывных функций многих переменных нейронными сетями с использованием практически произвольной непрерывной функции одного переменного. В данном разделе мы покажем, что эта функция действительно может быть произвольной и докажем обобщенную теорему Стоуна, естественным образом охватывающую и классическую теорему Стоуна, и аппроксимацию функций многих переменных суперпозициями и линейными комбинациями функций одного переменного.

Чтобы получить требуемое обобщение, перейдем от рассмотрения колец функций к изучению их алгебр, замкнутых относительно некоторой нелинейной унарной операции.

Пусть $E \subseteq C(X)$ - линейное пространство, $C(\mathbf{R})$ - пространство непрерывных функций на действительной оси $\mathbf{R}$, $f \in C(\mathbf{R})$ - нелинейная функция и для любого $g \in E$ выполнено $f(g) \in E$. В этом случае будем говорить, что E замкнуто относительно нелинейной унарной операции f.

Очевидный пример: множество функций n переменных, которые можно точно представить, используя заданную функцию одного переменного и линейные функции, является линейным пространством, замкнутым относительно нелинейной унарной операции f.

Замечание. Линейное пространство $E \subseteq C(X)$ замкнуто относительно нелинейной операции $f(x)=x^2$ тогда и только тогда, когда E является кольцом.

Действительно, $fg = \frac{1}{2}\left[(f+g)^2 - f^2 - g^2\right]$ поэтому для линейного пространства $E \subseteq C(X)$ замкнутость относительно унарной операции $f(x)=x^2$ равносильна замкнутости относительно произведения функций.

Согласно приведенному замечанию, теорема Стоуна может быть переформулирована так.

Пусть $E \subseteq C(X)$ - замкнутое линейное подпространство в $C(X)$, $1 \in E$, функции из E разделяют точки в X и E замкнуто относительно нелинейной унарной операции $f(x)=x^2$. Тогда $E=C(X)$.

Наше обобщение теоремы Стоуна состоит в замене $f(x)=x^2$ на произвольную нелинейную непрерывную функцию.

Теорема 1. Пусть $E \subseteq C(X)$ - замкнутое линейное подпространство в $C(X)$, $1 \in E$, функции из E разделяют точки в X и E замкнуто относительно нелинейной унарной операции $f \in C(\mathbf{R})$. Тогда $E=C(X)$.

Доказательство. Рассмотрим множество всех таких $p \in C(\mathbf{R})$, что $p(E) \subseteq E$, то есть для любого $g \in E$ выполнено: $p(g) \in E$. Обозначим это множество P_E. Оно обладает следующими свойствами:

1) P_E - полугруппа относительно суперпозиции функций;

2) P_E - замкнутое линейное подпространство в $C(\mathbf{R})$ (в топологии равномерной сходимости на компактах);

3) $1 \in P_E$ и $\mathrm{id} \in P_E$ ($\mathrm{id}(x) \equiv x$).

4) P_E включает хоть одну непрерывную нелинейную функцию.

Дальнейшее следует из теоремы 2, которая является, по существу, подготовительной теоремой о полугруппах функций.

Теорема 2. Пусть множество $P \subseteq C(\mathbf{R})$ удовлетворяет условиям 1-4. Тогда $P=C(\mathbf{R})$.

Доказательство опирается на три леммы.

Лемма 1. В условиях теоремы 2 существует дважды непрерывно дифференцируемая функция $g \in P$, не являющаяся линейной.

Äîêàçàòåëüñòâî. Ïóñòü $v(x) \in C^{\infty}(\mathbf{R})$, $v(x)=0$ ïðè $|x| > 1$, $\int_R v(x)dx=1$. Ðàññìîòðèì îïåðàòîð îñðåäíåíèÿ

$$J_\varepsilon f(x) = \int_R f(x+y)\frac{1}{\varepsilon}v\left(\frac{y}{\varepsilon}\right)dy$$

Äëÿ ëþáîãî $\varepsilon>0$ âûïîëíåíî: $J_\varepsilon f(x) \in P$.

Äåéñòâèòåëüíî, $f(x+y) \in E$ äëÿ êàæäîãî ôèêñèðîâàííîãî y (ò.ê. êîíñòàíòû ïðèíàäëåæàò E è E çàìêíóòî îòíîñèòåëüíî ëèíåéíûõ îïåðàöèé è ñóïðåìóìà ôóíêöèé). Èíòåãðàë $J_\varepsilon f(x)$. ïðèíàäëåæèò E, òàê êàê E ÿâëÿåòñÿ çàìêíóòûì ëèíåéíûì ïîäïðîñòðàíñòâîì â $C(\mathbf{R})$, à ýòîò èíòåãðàë - ïðåäåëîì êîíå÷íûõ ñóìì.

Ôóíêöèÿ $J_\varepsilon f(x)$ принадлежит $C^{\infty}(\mathbf{R})$ òàê êàê

$$J_\varepsilon f(x) = \int_R f(x+y)\frac{1}{\varepsilon}v\left(\frac{y}{\varepsilon}\right)dy = \int_R f(z)\frac{1}{\varepsilon}v\left(\frac{z-x}{\varepsilon}\right)dz$$

(íàïîìíèì, ÷òî v – ôóíêöèÿ ñ êîìïàêòíûì íîñèòåëåì).

Ñóùåñòâóåò òàêîå $\varepsilon>0$, ÷òî ôóíêöèÿ $g=J_\varepsilon f$ íå ÿâëÿåòñÿ ëèíåéíîé, ïîñêîëüêó $J_\varepsilon f \to f$ ïðè $\varepsilon \to 0$, ïðîòèâîðå÷èò íåëèíåéíîé ôóíêöèè çàìêíóòî, à f íå ÿâëÿåòñÿ ëèíåéíîé ôóíêöèåé. Òàêèì îáðàçîì, â ïðåäïîëîæåíèÿõ ëåììû ñóùåñòâóåò íåëèíåéíàÿ ôóíêöèÿ $g \in P \cap C^{\infty}(\mathbf{R})$, êîòîðóþ ìîæíî âûáðàòü â âèäå $g=J_\varepsilon f$

Лемма 2. Пусть в условиях теоремы 2 существует дважды непрерывно дифференцируемая функция $g \in P$, не являющаяся линейной. Тогда функция $q(x)=x^2$ принадлежит P.

Äîêàçàòåëüñòâî. Ñóùåñòâóåò òî÷êà x_0, äëÿ êîòîðîé $g''(x_0)\neq 0$. Îáîçíà÷èì $r(x)=2(g(x+x_0)-g(x_0)-xg'(x_0))/g''(x_0)$. Î÷åâèäíî, ÷òî $r\in P$, $r(0)=0$, $r'(0)=0$, $r''(0)=2$, $r(x)=x^2+o(x^2)$. Ïîýòîìó

$$r(\varepsilon x)/\varepsilon^2 \to x^2 \text{ ïðè } \varepsilon \to 0.$$

Поскольку P замкнуто, получаем: функция $q(x)=x^2$ принадлежит P.

Лемма 3. Пусть в условиях теоремы 2 функция $q(x)=x^2$ принадлежит P. Тогда P является кольцом – для любых $f,g\in P$ их произведение $fg\in P$.

Доказательство. Действительно, $fg=\frac{1}{2}\left[(f+g)^2-f^2-g^2\right]$и, так как P замкнуто относительно суперпозиции и линейных операций, то $fg\in P$.

Доказательство теоремы 2 заканчивается обращением к классической теореме Вейерштрасса о приближении функций многочленами: иç ëåìì 1-3 ñëåäóåò, ÷òî â óñëîâèÿõ òåîðåìû 2 P ÿâëÿåòñÿ êîëüöîì è, â ÷àñòíîñòè, ñîäåðæèò âñå ìíîãî÷ëåíû (которые получаются из 1 и id с помощью умножения и линейных операций). Ïî òåîðåìå Âåéåðøòðàññà îòñþäà ñëåäóåò, ÷òî $P=C(\mathbf{R})$.

Теоремы 1,2 можно трактовать как утверждения о универсальных аппроксимационных свойствах любой нелинейности: с помощью линейных операций и каскадного соединения можно из произвольных нелинейных элементов получить любой требуемый результат с любой наперед заданной точностью.

6. Точное представление многочленов от многих переменных с помощью одного произвольного многочлена от одного переменного, линейных операций и суперпозиции

В этом разделе исследуются полугруппы полиномов от одного переменного относительно суперпозиции. Показано, что если такая полугруппа содержит все многочлены первой степени и хотя бы один – более высокой, то она включает

все многочлены. На основании этого факта доказано, что всегда возможно представить многочлен от многих переменных суперпозициями произвольного нелинейного многочлена от одного переменного и линейных функций.

Вернемся к классическому вопросу о представлении функций многих переменных с помощью функций меньшего числа переменных. Следует еще раз заметить, что классических вопроса существует не один, а два:

1. Можно ли получить *точное* представление функции многих переменных с помощью суперпозиции функций меньшего числа переменных?

2. Можно ли получить *сколь угодно точную аппроксимацию* функции многих переменных с помощью некоторых более простых функций и операций?

В рамках первого вопроса особый интерес представляют конструкции, в которых для точного представления всех функций многих переменных используется один и тот же набор функций одного переменного.

Традиционно считается, что эти функции должны иметь весьма специальный и довольно экзотический вид, например, как в обсуждавшейся выше теореме Колмогорова, где использовались существенно негладкие функции.

Напротив, свобода в выборе функций одного переменного для решения второго вопроса при том же самоограничении (один набор функций одного переменного - для приближенного представления всех функций многих переменных) очень велика. Для этого, как показано в предыдущем разделе, можно использовать практически любую нелинейную функцию и достаточно всего одной.

Далее доказываются теоремы, относящиеся к первому вопросу (точное представление). В частности, показано, что можно точно представить любой многочлен от многих переменных с помощью суперпозиций произвольного нелинейного многочлена от одного переменного и линейных функций. Следовательно особенной пропасти между 1-м и 2-м вопросом не существует. Именно это обстоятельство побудило нас включить в книгу данный раздел.

Пусть $\mathbf{R}[X]$ - кольцо многочленов от одного переменного над полем $\mathbf{R}$, $E \subset \mathbf{R}[X]$ - линейное пространство многочленов над $\mathbf{R}$.

Предложение 1. Если E замкнуто относительно суперпозиции многочленов, содержит все многочлены первой степени и хотя бы один многочлен $p(x)$ степени $m>1$, то $E=\mathbf{R}[X]$.

Доказательство. Заметим, что степень многочлена $p'(x)=p(x+1)-p(x)$ равна $m-1$, и $p'(x) \in E$, так как E содержит многочлены первой степени (поэтому $x+1 \in E$), замкнуто относительно суперпозиции (поэтому $p(x+1) \in E$) и линейных операций (поэтому $p'(x) \in E$).

Если $m>2$, то понижаем степень с помощью конечных разностей (переходим к p', p'' и т.д.), пока не получим многочлен второй степени. Вычитая из него линейную часть и умножая на константу, получаем: $x^2 \in E$. Поэтому для любого $f \in E$ имеем $f^2 \in E$ (т.к. E - полугруппа). Дальнейшее очевидно: как неоднократно отмечалось выше, отсюда следует, что для любых $f,g \in E$ их произведение $fg \in E$ а с помощью умножения и сложения многочленов первой степени порождается все кольцо $\mathbf{R}[X]$.

Перейдем к задаче представления многочленов от многих переменных. Обозначим $\mathbf{R}[X_1, ..., X_n]$ кольцо многочленов от n переменных над полем $\mathbf{R}$.

Для каждого многочлена от одного переменного введем множество тех многочленов, которые можно выразить с его помощью, используя суперпозиции и линейные функции. Пусть p – многочлен от одного переменного, $E_p[X_1, ..., X_n]$ – множество многочленов от n переменных, которое можно получить из p и многочленов первой степени, принадлежащих $\mathbf{R}[X_1, ..., X_n]$, с помощью операций суперпозиции, сложения и умножения на число.

Следующие два предложения дают удобную для дальнейшего характеризацию $E_p[X_1, ..., X_n]$ и следуют непосредственно из определений.

Предложение 2. Множество $E_p[X_1, ..., X_n]$ является линейным пространством над $\mathbf{R}$ и для любого многочлена $g(x_1, ... ,x_n)$ из $E_p[X_1, ..., X_n]$ $p(g(x_1, ... ,x_n)) \in E_p[X_1, ..., X_n]$.

Предложение 3.Для данного p семейство линейных подпространств $L \subseteq \mathbf{R}[X_1, ..., X_n]$, содержащих все многочлены первой степени и удовлетворяющих условию

$$\text{если } g(x_1, ... ,x_n) \in L, \text{ то } p(g(x_1, ... ,x_n)) \in L,$$

замкнуто относительно пересечений. Минимальным по включению элементом этого семейства является $E_p[X_1, ..., X_n]$.

Для любого линейного подпространства $E \subseteq \mathbf{R}[X_1, ..., X_n]$ рассмотрим множество алгебраических унарных операций, которые переводят элементы E в элементы E:

$$P_E = \{p \in \mathbf{R}[X] \mid p(g(x_1, ... ,x_n)) \in E \text{ для любого } g(x_1, ... ,x_n) \in E\}.$$

Предложение 4. Для любого линейного подпространства $E \subseteq \mathbf{R}[X_1, ... ,X_n]$ множество полиномов P_E является линейным пространством над $\mathbf{R}$, замкнуто относительно суперпозиции и содержит все однородные многочлены первой степени.

Если линейное пространство E содержит 1, а P_E включает хотя бы один многочлен, степени $m>1$ (т.е. нелинейный), то $P_E = \mathbf{R}[X]$.

Доказательство. Замкнутость P_E относительно суперпозиции следует из определения, все однородные полиномы первой степени входят в P_E, поскольку E является линейным пространством, отсюда также следует, что P_E является линейным пространством. Наконец, если $1 \in E$ и P_E содержит многочлен степени $m>1$, то $1 \in P_E$, тогда $P_E = \mathbf{R}[X]$ по предложению 1.

Теорема 3. Пусть $E \subseteq \mathbf{R}[X_1, ..., X_n]$ – линейное подпространство, P_E содержит хотя бы один многочлен степени $m>1$ и $1 \in E$, тогда E является кольцом (с единицей).

Доказательство. По предложению 4 в условиях теоремы $P_E = \mathbf{R}[X]$. В частности, $x^2 \in P_E$. Это означает, что для любого $f \in E$ также и $f^2 \in E$. Поэтому для любых $f,g \in$ E получаем: $fg \in E$.

Теорема 4. Для любого многочлена p степени $m>1$

$$E_p[X_1, ..., X_n] = \mathbf{R}[X_1, ..., X_n]$$

Доказательство. Заметим, что $E=E_p[X_1, ..., X_n]$ – линейное подпространство в $\mathbf{R}[X_1, ..., X_n]$, P_E содержит хотя бы один многочлен (p) степени $m>1$ и E содержит все многочлены первой степени (и поэтому также 1). В силу теоремы 3, E является кольцом, а так как оно содержит все многочлены первой степени, то совпадает с $\mathbf{R}[X_1, ..., X_n]$, поскольку, используя умножение и сложение можно из этих многочленов получить любой.

Таким образом, из p и многочленов первой степени с помощью операций суперпозиции, сложения и умножения на число можно получить все элементы $\mathbf{R}[X_1, ..., X_n]$.

7. Нейронные сети – универсальные аппроксимирующие устройства и могут с любой точностью имитировать любой непрерывный автомат

Класс функций, вычислимый с помощью нейронных сетей, замкнут относительно линейных операций. Действительно, пусть есть нейронные сети S_1, S_2, ..., S_k , которые вычисляют функции F_1, F_2, ..., F_k от вектора входных сигналов x. Линейная комбинация $\alpha_0+\alpha_1F_1+\alpha_2F_2+...+\alpha_kF_k$ вычисляется сумматором (рис. 2) с весами α_0, α_1, α_2, ...,α_k, на вход которого подаются выходные сигналы сетей S_1, S_2, ..., S_k. Разница в числе тактов функционирования этих сетей до получения ответа легко компенсируется «линиями задержки», составленными из связей (рис. 6) с единичным весом.

Кроме того, класс функций, вычислимый с помощью нейронных сетей, замкнут относительно унарной операции, осуществляемой нелинейным преобразователем сигнала, входящим в состав нейрона (см. рис. 3, 5): если сеть S вычисляет функцию F, то, подавая выход этой сети на вход нелинейного преобразователя (рис. 3), получим на его выходе функцию $\varphi(F)$.

Тем самым, по теореме 1 множество функций, вычислимых нейронными сетями с заданной непрерывной нелинейной характеристической функцией, плотно в пространстве непрерывных функций от входных сигналов.

Теперь - об имитации гладких автоматов с помощью нейронных сетей. Если есть возможность с любой точностью приблизить любую непрерывную функцию

и нет ограничений на способы соединения устройств, то можно сколь угодно точно имитировать работу любого непрерывного автомата. Покажем это.

Каждый автомат имеет несколько входов (n), несколько выходов (p) и конечный набор (s) параметров состояния. Он вычисляет $s+p$ функций от $n+s$ переменных. Аргументы этих функций - входные сигналы (их n) и текущие параметры состояния (их s). Значения функций – выходные сигналы (их p) и параметры состояния на следующем шаге (их s). Каждый такой автомат можно представить как систему из $s+p$ более простых автоматов (рис. 9). Эти простые автоматы вычисляют по одной функции от $n+s$ переменных. Смена состояний достигается за счет того, что часть значений этих функций на следующем шаге становится аргументами – так соединены автоматы (см. рис. 9).

Таким образом, без потери общности можно рассматривать сеть автоматов как набор устройств, каждое из которых вычисляет функцию нескольких переменных $f(x_1, \ldots, x_n)$. Этот простой, но фундаментальный факт позволяет использовать предыдущие результаты. Нейронные сети позволяют с любой точностью вычислять произвольную непрерывную функцию $f(x_1, \ldots, x_n)$. Следовательно, с их помощью можно сколь угодно точно аппроксимировать функционирование любого непрерывного автомата.

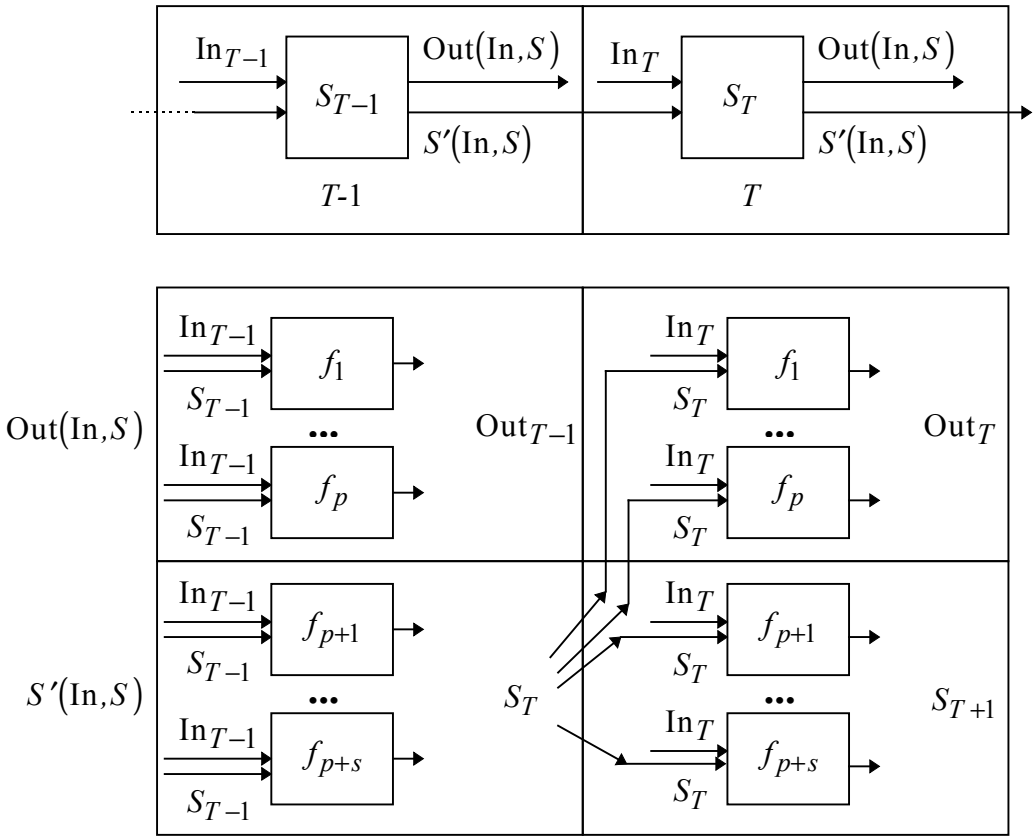


Рис. 9. Представление общего автомата с помощью модулей, вычисляющих функции многих переменных от входных сигналов: на верхней схеме представлено функционирование автомата, на нижней он разложен на отдельные модули. Приняты обозначения:
In - входные сигналы, Out - выходные сигналы, S - параметры состояния, T - дискретное время, Out(In,S) - зависимость выходных сигналов от значений входных и параметров состояния, S'(In,S) - зависимость состояния в следующий момент дискретного времени от входных сигналов и текущего состояния, f_1-f_p - функции переменных (In,S) - компоненты вектора Out(In,S), f_{p+1}-f_{p+s} - функции переменных (In,S) - компоненты вектора S'(In,S), индексами T, $T\pm1$ обозначены соответствующие моменты времени.

Главный вопрос этой главы: что могут нейронные сети. Ответ получен: нейронные сети могут все. Остается открытым другой вопрос - как их этому научить?

Работа над главой была поддержана Красноярским краевым фондом науки, грант 6F0124.

Литература

1. Колмогоров А.Н. О представлении непрерывных функций нескольких переменных суперпозициями непрерывных функций меньшего числа переменных. Докл. АН СССР, 1956. Т. 108, No. 2. С.179-182.

2. Арнольд В.И. О функциях трех переменных. Докл. АН СССР, 1957. Т. 114, No. 4. С. 679-681.

3. Колмогоров А.Н. О представлении непрерывных функций нескольких переменных в виде суперпозиции непрерывных функций одного переменного. Докл. АН СССР, 1957. Т. 114, No. 5. С. 953-956.

4. Витушкин А.Г. О многомерных вариациях. М.: Физматгиз, 1955.

5. Арнольд В.И. О представлении функций нескольких переменных в виде суперпозиции функций меньшего числа переменных // Математическое просвещение, 19 № с. 41-61.

6. Stone M.N. The generalized Weierstrass approximation theorem. Math. Mag., 1948. V.21. PP. 167-183, 237-254.

7. Шефер Х. Топологические векторные пространства. М.: Мир, 1971. 360 с.

8. Cybenko G. Approximation by superposition of a sigmoidal function. Mathematics of Control, Signals, and Systems, 1989. Vol. 2. PP. 303 - 314.

9. Hornik K., Stinchcombe M., White H. Multilayer feedforward networks are universal approximators. Neural Networks. 1989. Vol. 2. PP. 359 - 366.

10. Kochenov D.A., Rossiev D.A. Approximations of functions of C[A,B] class by neural-net predictors (architectures and results). AMSE Transaction, Scientific Siberian, A. 1993, Vol. 6. Neurocomputing. PP. 189-203. Tassin, France.

11. Gilev S.E., Gorban A.N. On completness of the class of functions computable by neural networks. Proc. of the World Congress on Neural Networks (WCNN'96). Sept. 15-18, 1996, San Diego, CA, Lawrens Erlbaum Accociates, 1996. PP. 984-991.

12. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. Новосибирск: Наука (Сиб. отделение), 1996. 276 с.

Глава 2.

Решение задач нейронными сетями

А.Н.Горбань

Вычислительный центр СО РАН в г.Красноярске[1]

Нейронные сети могут все. Но точно также «все» могут машины Тьюринга, интерполяционные многочлены, схемы Поста, ряды Фурье, рекурсивные функции, дизъюнктивные нормальные формы, сети Петри. В предыдущей главе было показано, что с помощью нейронных сетей можно сколь угодно точно аппроксимировать любую непрерывную функцию и имитировать любой непрерывный автомат. Это и слишком много - никому не нужен столь широкий класс функций, и слишком мало, так как далеко не все важные задачи ставятся как задачи аппроксимации.

Другое дело, что в конечном итоге решение практически любой задачи можно описать, как построение некоторой функции, перерабатывающей исходные данные в результат, но такое очень общее описание не дает никакой информации о способе построения этой функции. Мало иметь универсальные способности - надо для каждого класса задач указать, как применять эти способности. Именно здесь, при рассмотрении методов настройки нейронных сетей для решения задач должны выявиться реальные рамки их применимости. Конечно, такие реальные рамки изменяются со временем из-за открытия новых методов и решений.

В данной главе описано несколько базовых задач для нейронных сетей и основных или исторически первых методов настройки сетей для их решения:

1. Классификация (с учителем) (персептрон Розенблатта [1-3]).
2. Ассоциативная память (сети Хопфилда [4-7]).
3. Решение систем линейных уравнений (сети Хопфилда [8]).
4. Восстановление пробелов в данных (сети Хопфилда).
5. Кластер-анализ и классификация (без учителя) (сети Кохонена [9-12]).

Начнем мы, однако, не с сетей, а с систем, состоящих из одного элемента.

1. Настройка одноэлементных систем для решения задач

Даже системы из одного адаптивного сумматора находят очень широкое применение. Вычисление линейных функций необходимо во многих задачах. Вот неполный перечень «специальностей» адаптивного сумматора:

1. Линейная регрессия и восстановление простейших закономерностей [13-14];
2. Линейная фильтрация и адаптивная обработка сигналов [15];
3. Линейное разделение классов и простейшие задачи распознавания образов [16-18].

Задача линейной регрессии состоит в поиске наилучшего линейного приближения функции, заданной конечным набором значений: дана выборка значений вектора аргументов x^1 , ..., x^m, заданы значения функции F в этих точках: $F(x^i)=f_i$, требуется найти линейную (неоднородную) функцию $\varphi(x)=(\alpha,x)+\alpha_0$, ближайшую к F. Чтобы однозначно поставить задачу, необходимо доопределить, что значит «ближайшую». Наиболее популярен *метод наименьших квадратов*, согласно которому φ ищется из условия

$$\sum_{i=1}^{m}(F(x^i)-\varphi(x^i))^2 \to \min. \qquad (1)$$

Необходимо особенно подчеркнуть, что метод наименьших квадратов не является ни единственным, ни наилучшим во всех отношениях способом доопределения задачи регрессии. Его главное достоинство - квадратичность минимизируемого критерия и линейность получаемых уравнений на коэффициенты φ.

Явные формулы линейной регрессии легко получить, минимизируя квадратичный критерий качества регрессии. Обозначим

$$\Delta_i = (F(x^i)-\varphi(x^i));\ H=\sum_{i=1}^{m}\Delta_i^2=\sum_{i=1}^{m}(F(x^i)-\varphi(x^i))^2=$$

$$=\sum_{i=1}^{m}(f_i-(\alpha,x^i)-\alpha_0)^2.$$

Найдем производные минимизируемой функции H по настраиваемым параметрам:

$$\frac{\partial H}{\partial \alpha_j}=\sum_{i=1}^{m}\Delta_i x_j^i,\ (j=1,\ldots,n);\ \frac{\partial H}{\partial \alpha_0}=\sum_{i=1}^{m}\Delta_i.$$

где x_j^i - j-я координата вектора x^i.

---

[1] 660036, Красноярск-36, ВЦК СО РАН. E-mail: gorban@cc.krascience.rssi.ru

Приравнивая частные производные H нулю, получаем уравнения, из которых легко найти все α_j (j=0,...,n). Решение удобно записать в общем виде, если для всех i=1,...,m обозначить $x_0^i \equiv 1$ и рассматривать n+1-мерные векторы данных x^i и коэффициентов α. Тогда

$$\Delta_i = f_i - (\alpha, x^i),\ \partial H / \partial \alpha_j = \sum_{i=1}^{m} \Delta_i x_j^i \ \ (j = 0,1,\ldots,n).$$

Обозначим p n+1-мерный вектор с координатами $p_j = \frac{1}{m}\sum_{i=1}^{m} f_i x_j^i$; Q - матрицу размером (n+1)×(n+1) с элементами $q_{jk} = \frac{1}{m}\sum_{i=1}^{m} x_j^i x_k^i$.

В новых обозначениях решение задачи линейной регрессии имеет вид:

$$\varphi(x)=(\alpha,x),\ \alpha=Q^{-1}p. \qquad (2)$$

Приведем это решение в традиционных обозначениях *математической статистики*. Обозначим M_j среднее значение j-й координаты векторов исходной выборки:

$$M_j = \frac{1}{m}\sum_{i=1}^{m} x_j^i .$$

Пусть **M** - вектор с координатами M_j. Введем также обозначение s_j для выборочного среднеквадратичного отклонения:

$$s_j = \sqrt{\frac{1}{m}\sum_{i=1}^{m} (x_j^i - M_j)^2}$$

Величины s_j задают естественный масштаб для измерения j-х координат векторов x. Кроме того, нам потребуются величина s_f и коэффициенты корреляции f с j-ми координатами векторов x - r_{fj}:

$$s_f = \sqrt{\frac{1}{m}\sum_{i=1}^{m} (f_i - M_f)^2},\ M_f = \frac{1}{m}\sum_{i=1}^{m} f_i,\ r_{fj} = \frac{\frac{1}{m}\sum_{i=1}^{m} (f_i - M_f)(x_j^i - M_j)}{s_f s_j}.$$

Вернемся к n-мерным векторам данных и коэффициентов. Представим, что векторы сигналов проходят предобработку - центрирование и нормировку и далее мы имеем дело с векторами y^i:

$$y_j^i = \frac{x_j^i - M_j}{s_j}.$$

Это, в частности, означает, что все рассматриваемые координаты вектора x имеют ненулевую дисперсию, т.е. постоянные координаты исключаются из рассмотрения - они не несут полезной информации. Уравнения регрессии будем искать в форме: $\varphi(y)=(\beta,y)+\beta_0$. Получим:

$$\beta_0=M_f,\ \beta=s_f R^{-1} R_f, \qquad (3)$$

где R_f - вектор коэффициентов корреляции f с j-ми координатами векторов x, имеющий координаты r_{fj}, R - матрица коэффициентов корреляции между координатами вектора данных:

$$r_{kj} = \frac{\frac{1}{m}\sum_{i=1}^{m}(x_k^i - M_k)(x_j^i - M_j)}{s_k s_j} = \frac{1}{m}\sum_{i=1}^{m} y_k^i y_j^i.$$

В задачах обработки данных почти всегда возникает вопрос о последовательном уточнении результатов по мере поступления новых данных (*обработка данных «на лету»*). Существует, как минимум, два подхода к ответу на этот вопрос для задачи линейной регрессии. Первый подход состоит в том, что изменения в коэффициентах регрессии при поступлении новых данных рассматриваются как малые и в их вычислении ограничиваются первыми порядками теории возмущений. При втором подходе для каждого нового вектора данных делается шаг изменений коэффициентов, уменьшающий ошибку регрессии Δ^2 на вновь поступившем векторе данных. При этом «предыдущий опыт» фиксируется только в текущих коэффициентах регрессии.

В рамках первого подхода рассмотрим, как будет изменяться α из формулы (2) при добавлении нового вектора данных. В первом порядке теории возмущений найдем изменение вектора коэффициента α при изменении вектора p и матрицы Q:

$$\alpha + \Delta\alpha = (Q + \Delta Q)^{-1}(p + \Delta p);$$
$$(Q + \Delta Q)^{-1} = (Q(1 + Q^{-1}\Delta Q))^{-1} = Q^{-1} - Q^{-1}\Delta Q Q^{-1} + o(\Delta Q);$$
$$\Delta\alpha \cong Q^{-1}(\Delta p - \Delta Q\alpha).$$

Пусть на выборке $\{x^i\}_{i=1}^m$ вычислены p, Q, Q^{-1}. При получении нового вектора данных x^{m+1} и соответствующего значения $F(x^{m+1})=f_{m+1}$ имеем[2]:

$$\begin{aligned}
&\Delta p = \tfrac{1}{m+1}(f_{m+1}x^{m+1} - p);\\
&\Delta q_{jk} = \tfrac{1}{m+1}(x_j^{m+1}x_k^{m+1} - q_{jk}) \text{ (т.е. } \Delta Q = \tfrac{1}{m+1}(x^{m+1}\otimes(x^{m+1})^{\mathrm{T}})\text{);}\\
&\Delta(Q^{-1}) = \tfrac{1}{m+1}(Q^{-1} - (Q^{-1}x^{m+1})\otimes(Q^{-1}x^{m+1})^{\mathrm{T}});\\
&\Delta\alpha = \tfrac{1}{m+1}\Delta_{m+1}^0 Q^{-1}x^{m+1},
\end{aligned} \tag{4}$$

где $\Delta_{m+1}^0 = f_{m+1} - (\alpha, x^{m+1})$ - ошибка на векторе данных x^{m+1} регрессионной зависимости, полученной на основании выборки $\{x^i\}_{i=1}^m$.

Пересчитывая по приведенным формулам p, Q, Q^{-1} и α после каждого получения данных, получаем процесс, в котором последовательно уточняются уравнения линейной регрессии. И требуемый объем памяти, и количество операций имеют порядок n^2 - из-за необходимости накапливать и модифицировать матрицу Q^{-1}. Конечно, это меньше, чем потребуется на обычное обращение матрицы Q на каждом шаге, однако следующий простой алгоритм еще экономнее. Он вовсе не обращается к матрицам Q, Q^{-1} и основан на уменьшении на каждом шаге величины $(\Delta_{m+1}^0)^2 = (f_{m+1} - (\alpha, x^{m+1}))^2$ - квадрата ошибки на векторе данных x^{m+1} регрессионной зависимости, полученной на основании выборки $\{x^i\}_{i=1}^m$.

Вновь обратимся к формуле (2) и будем рассматривать n+1-мерные векторы данных и коэффициентов. Обозначим $x = x^{m+1}$; $\Delta = \Delta_{m+1}^0 = f_{m+1} - (\alpha, x^{m+1})$. Тогда

$$grad_\alpha \Delta = -\Delta \times x\,. \tag{5}$$

Последняя элементарная формула столь важна в теории адаптивных сумматоров, что носит «именное название» - формула Уидроу. «Обучение» адаптивного сумматора методом наискорейшего спуска состоит в изменении вектора коэффициентов α в направлении антиградиента Δ^2: на каждом шаге к α добавляется $h\times\Delta\times x$, где h - величина шага.

Если при каждом поступлении нового вектора данных x изменять α указанным образом, то получим последовательную процедуру построения линейной

---

[2] Напомним, что для векторов x, y матрица $x\otimes y^{\mathrm{T}}$ имеет своими элементами $x_i y_k$.

аппроксимации функции $F(x)$. Такой алгоритм обучения легко реализуется аппаратными средствами (изменение веса связи α_j есть произведение прошедшего по ней сигнала x_j на ошибку Δ и на величину шага). Возникает, однако, проблема сходимости: если h слишком мало, то сходимость будет медленной, если же слишком велико, то произойдет потеря устойчивости и сходимости не будет вовсе. Детальному изложению этого подхода и его приложений посвящен учебник [15].

Задача четкого разделения двух классов по обучающей выборке ставится так: имеется два набора векторов x^1 , ..., x^m и y^1,...,y^m . Заранее известно, что x^i относится к первому классу, а y^i - ко второму. Требуется построить решающее правило, то есть определить такую функцию $f(x)$, что при $f(x)>0$ вектор x относится к первому классу, а при $f(x)<0$ - ко второму.

Координаты классифицируемых векторов представляют собой значения некоторых признаков (свойств) исследуемых объектов.

Эта задача возникает во многих случаях: при диагностике болезней и определении неисправностей машин по косвенным признакам, при распознавании изображений и сигналов и т.п.

Строго говоря, классифицируются не векторы свойств, а объекты, которые обладают этими свойствами. Это замечание становится важным в тех случаях, когда возникают затруднения с построением решающего правила - например тогда, когда встречаются принадлежащие к разным классам объекты, имеющие одинаковые признаки. В этих случаях возможно несколько решений:

1) искать дополнительные признаки, позволяющие разделить классы;

2) примириться с неизбежностью ошибок, назначить за каждый тип ошибок свой штраф (c_{12} - штраф за то, что объект первого класса отнесен ко второму, c_{21} - за то, что объект второго класса отнесен к первому) и строить разделяющее правило так, чтобы минимизировать математическое ожидание штрафа;

3) перейти к нечеткому разделению классов - строить так называемые "функции принадлежности" $f_1(x)$ и $f_2(x)$ - $f_i(x)$ оценивает степень уверенности при отнесении объекта к i-му классу (i=1,2), для одного и того же x может быть так, что и $f_1(x)>0$, и $f_2(x)>0$.

Линейное разделение классов состоит в построении линейного решающего правила - то есть такого вектора α и числа α_0 (называемого порогом), что при $(x,\alpha)>\alpha_0$ x относится к первому классу, а при $(x, \alpha)<\alpha_0$ - ко второму.

Поиск такого решающего правила можно рассматривать как разделение классов в проекции на прямую. Вектор α задает прямую, на которую ортогонально проектируются все точки, а число α_0 - точку на этой прямой, отделяющую первый класс от второго.

Простейший и подчас очень удобный выбор состоит в проектировании на прямую, соединяющую центры масс выборок. Центр масс вычисляется в предположении, что массы всех точек одинаковы и равны 1. Это соответствует заданию α в виде

$$\alpha= (y^1 + y^2 +... + y^m)/m - (x^1 + x^2 +... + x^n)/n. \qquad (6)$$

Во многих случаях удобнее иметь дело с векторами единичной длины. Нормируя α, получаем:

$$\alpha= ((y^1 + y^2 +... + y^m)/m - (x^1 + x^2 +... + x^n)/n)/\|(y^1 + y^2 +... + y^m)/m - (x^1 + x^2 +... + x^n)/n\|.$$

Выбор α_0 может производиться из различных соображений. Простейший вариант - посередине между центрами масс выборок:

$$\alpha_0=(((y^1 + y^2 +... + y^m)/m,\alpha)+((x^1 + x^2 +... + x^n)/n,\alpha))/2.$$

Более тонкие способы построения границы раздела классов α_0 учитывают различные вероятности появления объектов разных классов, и оценки плотности распределения точек классов на прямой. Чем меньше вероятность появления данного класса, тем более граница раздела приближается к центру тяжести соответствующей выборки.

Можно для каждого класса построить приближенную плотность вероятностей распределения проекций его точек на прямую (это намного проще, чем для многомерного распределения) и выбирать α_0, минимизируя вероятность ошибки. Пусть решающее правило имеет вид: при $(x, \alpha)>\alpha_0$ x относится к первому классу, а при $(x, \alpha)<\alpha_0$ - ко второму. В таком случае вероятность ошибки будет равна

$$P = p_1 \int_{-\infty}^{\alpha_0} \rho_1(\chi) d\chi + p_2 \int_{\alpha_0}^{\infty} \rho_2(\chi) d\chi$$

где p_1, p_2 - априорные вероятности принадлежности объекта соответствующему классу,

$\rho_1(\chi)$, $\rho_2(\chi)$ - плотности вероятности для распределения проекций χ точек x в каждом классе.

Приравняв нулю производную вероятности ошибки по α_0, получим: число α_0, доставляющее минимум вероятности ошибки, является корнем уравнения:

$$p_1\rho_1(\chi)=p_2\rho_2(\chi), \qquad (7)$$

либо (если у этого уравнения нет решений) оптимальным является правило, относящее все объекты к одному из классов.

Если принять гипотезу о нормальности распределений:

$$\rho(y)=\frac{1}{\sqrt{2\pi}\sigma}e^{-(y-a)^2/2\sigma^2},$$

то для определения α_0 получим:

$$\frac{p_1}{\sqrt{2\pi}\sigma_1}e^{-(y-a_1)^2/2\sigma_1^2}=\frac{p_2}{\sqrt{2\pi}\sigma_2}e^{-(y-a_2)^2/2\sigma_2^2},$$

$$\ln(p_1/p_2)-\ln(\sigma_1/\sigma_2)-(y-a_1)^2/2\sigma_1^2+(y-a_2)^2/2\sigma_2^2=0$$

Если это уравнение имеет два корня $y=\alpha_1$, α_2, ($\alpha_1<\alpha_2$) то наилучшим решающим правилом будет: при $\alpha_1<(x, \alpha)<\alpha_2$ объект принадлежит одному классу, а при $\alpha_1>(x, \alpha)$ или $(x, \alpha)>\alpha_2$ - другому (какому именно, определяется тем, которое из произведений $p_i\rho_i(\chi)$ больше). Если корней нет, то оптимальным является отнесение к одному из классов. Случай единственного корня представляет интерес только тогда, когда $\sigma_1=\sigma_2$. При этом уравнение превращается в линейное и мы приходим к исходному варианту - единственной разделяющей точке α_0.

Таким образом, разделяющее правило с единственной разделяющей точкой α_0 не является наилучшим для нормальных распределений и надо искать две разделяющие точки.

Если сразу ставить задачу об оптимальном разделении многомерных нормальных распределений, то получим, что наилучшей разделяющей поверхностью является квадрика (на прямой типичная «квадрика» - две точки). Предполагая, что ковариационные матрицы классов совпадают (в одномерном случае это предположение о том, что $\sigma_1=\sigma_2$), получаем линейную разделяющую поверхность. Она ортогональна прямой, соединяющей центры выборок не в обычном скалярном произведении, а в

специальном: $\langle x, y\rangle = \left(x, \Sigma^{-1} y\right)$, где Σ - общая ковариационная матрица классов. За деталями отсылаем к прекрасно написанной книге [17], см. также [11,12,16].

Важная возможность усовершенствовать разделяющее правило состоит с использовании оценки не просто вероятности ошибки, а среднего риска: каждой ошибке приписывается «цена» c_i и минимизируется сумма $c_1 p_1 \rho_1(\chi)+c_2 p_2 \rho_2(\chi)$. Ответ получается практически тем же (всюду p_i заменяются на $c_i p_i$), но такая добавка важна для многих приложений.

Требование безошибочности разделяющего правила на обучающей выборке принципиально отличается от обсуждавшихся критериев оптимальности. На основе этого требования строится **персептрон Розенблатта** - «дедушка» современных нейронных сетей [1].

Возьмем за основу при построении гиперплоскости, разделяющей классы, отсутствие ошибок на обучающей выборке. Чтобы удовлетворить этому условию, придется решать систему линейных неравенств:

$$(x^i, \alpha)>\alpha_0 \ (i=1,...,n)$$

$$(y^j, \alpha)<\alpha_0 \ (j=1,...,m).$$

Здесь x^i (i=1,..,n) - векторы из обучающей выборки, относящиеся к первому классу, а y^j (j=1,..,n) - ко второму.

Удобно переформулировать задачу. Увеличим размерности всех векторов на единицу, добавив еще одну координату $-\alpha_0$ к α, x_0 =1 - ко всем x и y_0 =1 - ко всем y . Сохраним для новых векторов прежние обозначения - это не приведет к путанице.

Наконец, положим $z^i = x^i$ (i=1,...,n), $z^j = -y^j$ (j=1,...,m).

Тогда получим систему n+m неравенств

$$(z^i,\alpha)>0 \ (i=1,...,n+m),$$

которую будем решать относительно α. Если множество решений непусто, то любой его элемент α порождает решающее правило, безошибочное на обучающей выборке.

Итерационный алгоритм решения этой системы чрезвычайно прост. Он основан на том, что для любого вектора x его скалярный квадрат (x,x) больше нуля. Пусть α - некоторый вектор, претендующий на роль решения неравенств $(z^i,\alpha)>0$ (i=1,...,n+m), однако часть из них не выполняется. Прибавим те z^i, для которых неравенства имеют

неверный знак, к вектору α и вновь проверим все неравенства $(z^i,\alpha)>0$ и т.д. Если они совместны, то процесс сходится за конечное число шагов. Более того, добавление z^i к α можно производить сразу после того, как ошибка ($(z^i,\alpha)<0$) обнаружена, не дожидаясь проверки всех неравенств - и этот вариант алгоритма тоже сходится [2].

2. Сети Хопфилда

Перейдем от одноэлементных систем к нейронным сетям. Пусть α_{ij} - вес связи, ведущей от *j*-го нейрона к *i*-му (полезно обратить внимание на порядок индексов). Для полносвязных сетей определены значения α_{ij} при всех *i*,*j*, для других архитектур связи, ведущие от *j*-го нейрона к *i*-му для некоторых *i*,*j* не определены. В этом случае положим $\alpha_{ij}=0$.

В данном разделе речь пойдет в основном о полносвязных сетях. Пусть на выходах всех нейронов получены сигналы x_j (*j*-номер нейрона). Обозначим *x* вектор этих выходных сигналов. Прохождение вектора сигналов *x* через сеть связей сводится к умножению матрицы (α_{ij}) на вектор сигналов *x*. В результате получаем вектор входных сигналов нелинейных элементов нейронов: $y_i = \sum_j \alpha_{ij} x_j$.

Это соответствие «*прохождение сети ≡ умножение матрицы связей на вектор сигналов*» является основой для перевода обычных численных методов на нейросетевой язык и обратно. Практически всюду, где основной операцией является умножение матрицы на вектор, применимы нейронные сети. С другой стороны, любое устройство, позволяющее быстро осуществлять такое умножение, может использоваться для реализации нейронных сетей.

В частности, *вычисление градиента квадратичной формы* $H = \frac{1}{2}(x, Qx)$ может осуществляться полносвязной сетью с *симметричной* матрицей связей: $\mathrm{grad}H = Qx$ ($\alpha_{ij} = q_{ij} = q_{ji}$). Именно это наблюдение лежит в основе данного раздела.

Что можно сделать, если мы умеем вычислять градиент квадратичной формы?

В первую очередь, можно методом наискорейшего спуска **искать точку минимума многочлена второго порядка**. Пусть задан такой многочлен: $P(x) = \frac{1}{2}(x, Qx) + (b, x)$. Его градиент равен $\mathrm{grad}P = Qx + b$. Этот вектор может быть

получен при прохождении вектора x через сеть с весами связей $\alpha_{ij} = q_{ij} = q_{ji}$ при условии, что на входной сумматор каждого нейрона по дополнительной связи веса b подается стандартный единичный сигнал.

Зададим теперь функционирование сети формулой

$$x' = x - h\mathrm{grad}P = x - h(Qx + b) \qquad (8)$$

Нелинейных элементов вовсе не нужно! Каждый (j-й) нейрон имеет входные веса $\alpha_{ij} = -hq_{ij}$ для связей с другими нейронами ($i \neq j$), вес $-b_j$ для постоянного единичного входного сигнала и вес $\alpha_{jj} = 1 - hq_{jj}$ для связи нейрона с самим собой (передачи на него его же сигнала с предыдущего шага). Выбор шага h>0 может вызвать затруднение (он зависит от коэффициентов минимизируемого многочлена). Есть, однако, простое решение: в каждый момент дискретного времени T выбирается свое значение h_T. Достаточно, чтобы шаг стремился со временем к нулю, а сумма шагов - к бесконечности (например, $h_T = \frac{1}{T}$, или $h_T = \frac{1}{\sqrt{T}}$).

Итак, простая симметричная полносвязная сеть без нелинейных элементов может методом наискорейшего спуска искать точку минимума квадратичного многочлена.

Решение системы линейных уравнений $Ax = b$ сводится к минимизации многочлена

$$P = \frac{1}{2}((Ax - b),(Ax - b)) = \frac{1}{2}(x, A^{\mathrm{T}}Ax) - (A^{\mathrm{T}}b, x) - \frac{1}{2}(b, b).$$

Поэтому решение системы может производиться нейронной сетью. Простейшая сеть, вычисляющая градиент этого многочлена, не полносвязна, а состоит из двух слоев: первый с матрицей связей A, второй - с транспонированной матрицей A^{T}. Постоянный единичный сигнал подается на связи с весами b_j на первом слое. Минимизация этого многочлена, а значит и решение системы линейных уравнений, может проводиться так же, как и в общем случае, в соответствии с формулой $x' = x - h\mathrm{grad}P$. Усовершенствованные варианты алгоритма решения можно найти в работах Сударикова [8].

Небольшая модификация позволяет вместо безусловного минимума многочлена второго порядка P искать точку условного минимума с условиями $x_i = c_i$ для $i = i_1, \ldots, i_k$, то есть точку минимума P в ограничении на аффинное

многообразие, параллельное некоторым координатным плоскостям. Для этого вместо формулы

$$x' = x - h\mathrm{grad}P = x - h(Qx + b)$$

следует использовать:

$$x'_i = c_i \text{ при } i = i_1, \ldots, i_k;$$

$$x'_i = x_i - h\frac{\partial P}{\partial x_i} = x_i - h\left(\sum_j q_{ij}x_j + b_i\right) \text{ при } i \neq i_1, \ldots, i_k. \qquad (9)$$

Устройство, способное *находить точку условного минимума многочлена второго порядка* при условиях вида $x_i = c_i$ для $i = i_1, \ldots, i_k$ позволяет решать важную задачу - **заполнять пробелы в данных** (и, в частности, строить линейную регрессию).

Предположим, что получаемые в ходе испытаний векторы данных подчиняются многомерному нормальному распределению:

$$\rho(x) = Ce^{-\frac{1}{2}((x-\mathbf{M}x),Q(x-\mathbf{M}x))},$$

где $\mathbf{M}x$ - вектор математических ожиданий координат, $Q = \Sigma^{-1}$, Σ - ковариационная матрица, n - размерность пространства данных,

$$C = \frac{1}{(2\pi)^{n/2}\sqrt{\det\Sigma}}.$$

Напомним определение матрицы Σ:

$$(\Sigma)_{ij} = \mathbf{M}((x_i - \mathbf{M}x_i)(x_j - \mathbf{M}x_j)),$$

где $\mathbf{M}$ - символ математического ожидания, нижний индекс соответствует номеру координаты.

В частности, простейшая оценка ковариационной матрицы по выборке дает:

$$S = \frac{1}{m}\sum_j (x^j - \mathbf{M}x) \otimes (\mathbf{M}x^j - \mathbf{M}x)^{\mathrm{T}},$$

где m - число элементов в выборке, верхний индекс j - номер вектора данных в выборке, верхний индекс Т означает транспонирование, а $\otimes$ - произведение вектора-столбца на вектор-строку (тензорное произведение).

Пусть у вектора данных x известно несколько координат: $x_i = c_i$ для $i = i_1, \ldots, i_k$. Наиболее вероятные значения неизвестных координат должны доставлять условный максимум показателю нормального распределения - многочлену второго порядка

$((x-\mathbf{M}x),Q(x-\mathbf{M}x))$ (при условии $x_i = c_i$ для $i = i_1,\ldots,i_k$). Эти же значения будут условными математическими ожиданиями неизвестных координат при заданных условиях.

Таким образом, чтобы построить сеть, заполняющую пробелы в данных, достаточно сконструировать сеть для поиска точек условного минимума многочлена $((x-\mathbf{M}x),Q(x-\mathbf{M}x))$ при условиях следующего вида: $x_i = c_i$ для $i = i_1,\ldots,i_k$. Матрица связей Q выбирается из условия $Q = \Sigma^{-1}$, где Σ - ковариационная матрица (ее оценка по выборке).

На первый взгляд, пошаговое накопление $Q = \Sigma^{-1}$ по мере поступления данных требует слишком много операций - получив новый вектор данных требуется пересчитать оценку Σ, а потом вычислить $Q = \Sigma^{-1}$. Можно поступать и по-другому, воспользовавшись формулой приближенного обращения матриц первого порядка точности:

$$(\Sigma + \varepsilon\Delta)^{-1} = \Sigma^{-1} - \varepsilon\Sigma^{-1}\Delta\Sigma^{-1} + o(\varepsilon).$$

Если же добавка Δ имеет вид $\Delta = y \otimes y^{\mathrm{T}}$, то

$$(\Sigma + \varepsilon\Delta)^{-1} = \Sigma^{-1} - \varepsilon(\Sigma^{-1}y) \otimes (\Sigma^{-1}y)^{\mathrm{T}} + o(\varepsilon). \qquad (10)$$

Заметим, что решение задачи (точка условного минимума многочлена) не меняется при умножении Q на число. Поэтому полагаем:

$$Q_0 = 1,\ Q_{k+1} = Q_k + \varepsilon(Q_k(x^{k+1} - (\mathbf{M}x)^{k+1})) \otimes (Q_k(x^{k+1} - (\mathbf{M}x)^{k+1}))^{\mathrm{T}},$$

где 1 - единичная матрица, $\varepsilon>0$ - достаточно малое число, x^{k+1} - k+1-й вектор данных, $(\mathbf{M}x)^{k+1}$ - среднее значение вектора данных, уточненное с учетом x^{k+1}:

$$(\mathbf{M}x)^{k+1} = \frac{1}{k+1}(k(\mathbf{M}x)^k + x^{k+1})$$

В формуле для пошагового накопления матрицы Q ее изменение ΔQ при появлении новых данных получается с помощью вектора $y = x^{k+1} - (\mathbf{M}x)^{k+1}$, пропущенного через сеть: $(\Delta Q)_{ij} = \varepsilon z_i z_j$, где $z = Qy$. Параметр ε выбирается достаточно малым для того, чтобы обеспечить положительную определенность получаемых матриц (и, по возможности, их близость к истинным значениям Q).

Описанный процесс формирования сети можно назвать обучением. Вообще говоря, можно проводить формальное различение между формированием сети по *явным*

формулам и по алгоритмам, не использующим явных формул для весов связей (*неявным*). Тогда термин «обучение» предполагает неявные алгоритмы, а для явных остается название «формирование». Здесь мы такого различия проводить не будем.

Если при обучении сети поступают *некомплектные данные* x^{k+1} с отсутствием значений некоторых координат, то сначала эти значения восстанавливаются с помощью имеющейся сети, а потом используются в ее дальнейшем обучении.

Во всех задачах оптимизации существенную роль играет вопрос о *правилах остановки:* когда следует прекратить циклическое функционирование сети, остановиться и считать полученный результат ответом? Простейший выбор - остановка по малости изменений: если изменения сигналов сети за цикл меньше некоторого фиксированного малого δ (при использовании переменного шага δ может быть его функцией), то оптимизация заканчивается.

До сих пор речь шла о минимизации положительно определенных квадратичных форм и многочленов второго порядка. Однако самое знаменитое приложение полносвязных сетей связано с увеличением значений положительно определенных квадратичных форм. Речь идет о системах **ассоциативной памяти** [4-7,9,10,12].

Предположим, что задано несколько эталонных векторов данных $x^1,\ldots,x^m$ и при обработке поступившего на вход системы вектора x требуется получить на выходе ближайший к нему эталонный вектор. Мерой сходства в простейшем случае будем считать косинус угла между векторами - для векторов фиксированной длины это просто скалярное произведение. Можно ожидать, что изменение вектора x по закону

$$x' = x + h\sum_k x^k (x^k, x), \qquad (11)$$

где h - малый шаг, приведет к увеличению проекции x на те эталоны, скалярное произведение на которые (x^k, x) больше.

Ограничимся рассмотрением эталонов, и ожидаемых результатов обработки с координатами ± 1. Развивая изложенную идею, приходим к дифференциальному уравнению

$$\frac{dx}{dt} = -\text{grad}H,$$

$$H = H_0 + \theta H_1,\ H_0(x) = -\frac{1}{2}\sum_k (x^k, x)^2,\ H_1(x) = \frac{1}{2}\sum_i (x_i^2 - 1)^2, \quad (12)$$

$$\text{grad}H_0 = -\sum_k x^k (x^k, x),\ (\text{grad}H_1)_j = (x_j^2 - 1)x_j,$$

где верхними индексами обозначаются номера векторов-эталонов, нижними - координаты векторов.

Функция H называется «*энергией*» сети, она минимизируется в ходе функционирования. Слагаемое H_0 вводится для того, чтобы со временем возрастала проекция вектора x на те эталоны, которые к нему ближе, слагаемое H_1 обеспечивает стремление координат вектора x к ± 1. Параметр θ определяет соотношение между интенсивностями этих двух процессов. Целесообразно постепенно менять θ со временем, начиная с малых θ<1, и приходя в конце концов к θ>1.

Подробнее системы ассоциативной памяти рассмотрены в отдельной главе. Здесь же мы ограничимся обсуждением получающихся весов связей. Матрица связей построенной сети определяется функцией H_0, так как $(\text{grad}H_1)_j = (x_j^2 - 1)x_j$ вычисляется непосредственно при j-м нейроне без участия сети. Вес связи между i-м и j-м нейронами не зависит от направления связи и равен

$$\alpha_{ij} = \sum_k x_i^k x_j^k\ . \quad (13)$$

Эта простая формула имеет чрезвычайно важное значение для развития теории нейронных сетей. Вклад k-го эталона в связь между i-м и j-м нейронами ($x_i^k x_j^k$) равен +1, если i-я и j-я координаты этого эталона имеют одинаковый знак, и равен -1, если они имеют разный знак.

В результате возбуждение i-го нейрона передается j-му (и симметрично, от j-го к i-му), если у большинства эталонов знак i-й и j-й координат совпадают. В противном случае эти нейроны тормозят друг друга: возбуждение i-го ведет к торможению j-го, торможение i-го - к возбуждению j-го (воздействие j-го на i-й симметрично). Это правило образования ассоциативных связей (правило Хебба) сыграло огромную роль в теории нейронных сетей.

3. Сети Кохонена для кластер-анализа и классификации без учителя

Построение отношений на множестве объектов - одна из самых загадочных и открытых для творчества областей применения искусственного интеллекта. Первым и наиболее распространенным примером этой задачи является классификация без учителя. Задан набор объектов, каждому объекту сопоставлен вектор значений признаков (строка таблицы). Требуется разбить эти объекты на классы эквивалентности.

Естественно, прежде, чем приступать к решению этой задачи, нужно ответить на один вопрос: зачем производится это разбиение и что мы будем делать с его результатом? Ответ на него позволит приступить к формальной постановке задачи, которая всегда требует компромисса между сложностью решения и точностью формализации: буквальное следование содержательному смыслу задачи нередко порождает сложную вычислительную проблему, а следование за простыми и элегантными алгоритмами может привести к противоречию со здравым смыслом.

Итак, зачем нужно строить отношения эквивалентности между объектами? В первую очередь - для фиксации знаний. Люди накапливают знания о классах объектов - это практика многих тысячелетий, зафиксированная в языке: знание относится к имени класса (пример стандартной древней формы: "люди смертны", "люди" - имя класса). В результате классификации как бы появляются новые имена и правила их присвоения.

Для каждого нового объекта мы должны сделать два дела:

1) найти класс, к которому он принадлежит;

2) использовать новую информацию, полученную об этом объекте, для исправления (коррекции) правил классификации.

Какую форму могут иметь правила отнесения к классу? Веками освящена традиция представлять класс его "типичным", "средним", "идеальным" и т.п. элементом. Этот типичный объект является идеальной конструкцией, олицетворяющей класс.

Отнесение объекта к классу проводится путем его сравнения с типичными элементами разных классов и выбора ближайшего. Правила, использующие типичные объекты и меры близости для их сравнения с другими, очень популярны и сейчас.

Простейшая мера близости объектов - квадрат евклидового расстояния между векторами значений их признаков (чем меньше расстояние - расстояние - тем ближе объекты). Соответствующее определение признаков типичного объекта - среднее арифметическое значение признаков по выборке, представляющей класс.

Мы не оговариваем специально существование априорных ограничений, налагаемых на новые объекты - естественно, что "вселенная" задачи много 'уже и гораздо определеннее Вселенной.

Другая мера близости, естественно возникающая при обработке сигналов, изображений и т.п. - квадрат коэффициента корреляции (чем он больше, тем ближе объекты). Возможны и иные варианты - все зависит от задачи.

Если число классов m заранее определено, то задачу классификации без учителя можно поставить следующим образом.

Пусть $\{x^p\}$ - векторы значений признаков для рассматриваемых объектов и в пространстве таких векторов определена мера их близости $\rho\{x,y\}$. Для определенности примем, что чем ближе объекты, тем меньше ρ. С каждым классом будем связывать его типичный объект. Далее называем его ядром класса. Требуется определить набор из m ядер $y^1, y^2, \ldots y^m$ и разбиение $\{x^p\}$ на классы:

$\{x^p\} = Y_1 \cup Y_2 \cup \ldots \cup Y_m$

минимизирующее следующий критерий

$$Q = \sum_{i=1}^{m} D_i \to \min, \tag{14}$$

где для каждого (i-го) класса D_i - сумма расстояний от принадлежащих ему точек выборки до ядра класса:

$$D_i = \sum_{x^p \in Y_i} \rho(x^p, y^i). \tag{15}$$

Минимум Q берется по всем возможным положениям ядер y^i и всем разбиениям $\{x^p\}$ на m классов Y_i.

Если число классов заранее не определено, то полезен критерий слияния классов: классы Y_i и Y_j сливаются, если их ядра ближе, чем среднее расстояние от элемента класса до ядра в одном из них. (Возможны варианты: использование среднего расстояния по обоим классам, использование порогового коэффициента, показывающего, во сколько раз должно расстояние между ядрами превосходить среднее расстояние от элемента до ядра и др.)

Использовать критерий слияния классов можно так: сначала принимаем гипотезу о достаточном числе классов, строим их, минимизируя Q, затем некоторые Y_i объединяем, повторяем минимизацию Q с новым числом классов и т.д.

Существует много эвристических алгоритмов классификации без учителя, основанных на использовании мер близости между объектами. Каждый из них имеет свою область применения, а наиболее распространенным недостатком является отсутствие четкой формализации задачи: совершается переход от идеи кластеризации прямо к алгоритму, в результате неизвестно, что ищется (но что-то в любом случае находится, иногда - неплохо).

Сетевые алгоритмы классификации без учителя строятся на основе итерационного метода динамических ядер. Опишем его сначала в наиболее общей абстрактной форме. Пусть задана выборка предобработанных векторов данных $\{x^p\}$. Пространство векторов данных обозначим E. Каждому классу будет соответствовать некоторое ядро a. Пространство ядер будем обозначать A. Для каждых $x \in E$ и $a \in A$ определяется мера близости $d(x,a)$. Для каждого набора из k ядер $a_1,...,a_k$ и любого разбиения $\{x^p\}$ на k классов $\{x^p\}=P_1 \cup P_2 \cup ... \cup P_k$ определим критерий качества:

$$D = D(a_1, a_2, ..., a_k, P_1, P_2, ... P_k) = \sum_{i=1}^{k} \sum_{x \in P_i} d(x, a_i) \qquad (16)$$

Требуется найти набор $a_1,...,a_k$ и разбиение $\{x^p\}=P_1 \cup P_2 \cup ... \cup P_k$, минимизирующие D.

Шаг алгоритма разбивается на два этапа:

1-й этап - для фиксированного набора ядер $a_1,...,a_k$ ищем минимизирующее критерий качества D разбиение $\{x^p\}=P_1 \cup P_2 \cup ... \cup P_k$; оно дается решающим правилом: $x \in P_i$, если $d(x, a_i) < d(x, a_j)$ при $i \neq j$, в том случае, когда для x минимум $d(x,a)$ достигается при нескольких значениях i, выбор между ними может быть сделан произвольно;

2-й этап - для каждого P_i (i=1,...,k), полученного на первом этапе, ищется $a_i \in A$, минимизирующее критерий качества (т.е. слагаемое в D для данного i - $D_i = \sum_{x \in P_i} d(x, a_i)$

Начальные значения $a_1,...,a_k$, $\{x^p\}=P_1 \cup P_2 \cup ... \cup P_k$ выбираются произвольно, либо по какому-нибудь эвристическому правилу.

На каждом шаге и этапе алгоритма уменьшается критерий качества D, отсюда следует сходимость алгоритма - после конечного числа шагов разбиение $\{x^p\}=P_1\cup P_2\cup...\cup P_k$ уже не меняется.

Если ядру a_i сопоставляется элемент сети, вычисляющий по входному сигналу x функцию $d(x,a_i)$, то решающее правило для классификации дается интерпретатором "победитель забирает все": элемент x принадлежит классу P_i, если выходной сигнал i-го элемента $d(x,a_i)$ меньше всех остальных

Единственная вычислительная сложность в алгоритме может состоять в поиске ядра по классу на втором этапе алгоритма, т.е. в поиске $a\in A$, минимизирующего $D_i = \sum_{x\in P_i} d(x,a)$

В связи с этим, в большинстве конкретных реализаций метода мера близости d выбирается такой, чтобы легко можно было найти a, минимизирующее D для данного P .

В простейшем случае пространство ядер A совпадает с пространством векторов x, а мера близости $d(x,a)$ - положительно определенная квадратичная форма от x-a, например, квадрат евклидового расстояния или другая положительно определенная квадратичная форма. Тогда ядро a_i, минимизирующее D_i, есть центр тяжести класса P_i :

$$a_i = \frac{1}{|P_i|}\sum_{x\in P_i} x\,, \qquad (17)$$

где $|P_i|$ - число элементов в P_i.

В этом случае также упрощается и решающее правило, разделяющее классы. Обозначим $d(x,a)=(x-a,x-a)$, где (.,.) - билинейная форма (если d - квадрат евклидового расстояния между x и a, то (.,.) - обычное скалярное произведение). В силу билинейности

$d(x,a)=(x-a,x-a)=(x,x)-2(x,a)+(a,a)$.

Чтобы сравнить $d(x,a_i)$ для разных i и найти среди них минимальное, достаточно вычислить линейную неоднородную функцию от x:

$d_1(x,a_i) = (a_i,a_i)-2(x,a_i)$.

Минимальное значение $d(x,a_i)$ достигается при том же i, что и минимум $d_1(x,a_i)$, поэтому решающее правило реализуется с помощью k сумматоров, вычисляющих $d(x,a)$

и интерпретатора, выбирающего сумматор с минимальным выходным сигналом. Номер этого сумматора и есть номер класса, к которому относится *x*.

Пусть теперь мера близости - коэффициент корреляции между вектором данных и ядром класса:

$$d(x,a) = r(x,a) = \sum_j \frac{(x_j - M_x)(a_j - M_a)}{\sigma_x \sigma_a}$$

где x_j, a_j - координаты векторов, $M_x = \frac{1}{n}\sum_j x_j$ (и аналогично M_a), n - размерность пространства данных, $\sigma_x = \sqrt{\frac{1}{n}\sum_j (x_j - M_x)^2}$ (и аналогично σ_a).

Предполагается, что данные предварительно обрабатываются (нормируются и центрируются) по правилу:

$$x \to \frac{x_j - M_x}{\sigma_x}.$$

Точно также нормированы и центрированы векторы ядер *a*. Поэтому все обрабатываемые векторы и ядра принадлежат сечению единичной евклидовой сферы (||*x*||=1) гиперплоскостью ($\sum_i x_i = 0$). В таком случае $d(x,a) = (x,a)$.

Задача поиска ядра для данного класса *P* имеет своим решением

$$a_P = \sum_{x \in P} x \Big/ \left\| \sum_{x \in P} x \right\|. \qquad (18)$$

В описанных простейших случаях, когда ядро класса точно определяется как среднее арифметическое (или нормированное среднее арифметическое) элементов класса, а решающее правило основано на сравнении выходных сигналов линейных адаптивных сумматоров, нейронную сеть, реализующую метод динамических ядер, называют сетью Кохонена. В определении ядер *a* для сетей Кохонена входят суммы $\sum_{x \in P} x$. Это позволяет накапливать новые динамические ядра, обрабатывая по одному примеру и пересчитывая a_i после появления в P_i нового примера. Сходимость при такой модификации, однако, ухудшается.

Закончим раздел рассмотрением различных способов использования полученных классификаторов.

1. *Базовый способ*: для вектора данных x^i и каждого ядра a_i вычисляется $y_i=d(x,a_i)$ (условимся считать, что правильному ядру отвечает максимум d, изменяя, если надо, знак d); по правилу «победитель забирает все» строка ответов y_i преобразуется в строку, где только один элемент, соответствующий максимальному y_i, равен 1, остальные - нули. Эта строка и является результатом функционирования сети. По ней может быть определен номер класса (номер места, на котором стоит 1) и другие показатели.

2. *Метод аккредитации*: за слоем элементов базового метода, выдающих сигналы 0 или 1 по правилу "победитель забирает все" (далее называем его слоем базового интерпретатора), надстраивается еще один слой выходных сумматоров. С каждым (i-м) классом ассоциируется q-мерный выходной вектор z^i с координатами z^i_j . Он может формироваться по-разному: от двоичного представления номера класса до вектора ядра класса. Вес связи, ведущей от i-го элемента слоя базового интерпретатора к j-му выходному сумматору определяется в точности как z^i_j . Если на этом i-м элементе базового интерпретатора получен сигнал 1, а на остальных - 0, то на выходных сумматорах будут получены числа z^i_j.

3. *Нечеткая классификация*. Пусть для вектор данных x обработан слоем элементов, вычисляющих $y_i=d(x,a_i)$. Идея дальнейшей обработки состоит в том, чтобы выбрать из этого набора $\{y_i\}$ несколько самых больших чисел и после нормировки объявить их значениями функций принадлежности к соответствующим классам. Предполагается, что к остальным классам объект наверняка не принадлежит. Для выбора семейства G наибольших y_i определим следующие числа:

$$y_{\max} = \max\{y_i\},\ M_y = \frac{1}{k}\sum_i y_i,\ s = (1-\alpha)M_y + \alpha y_{\max}$$

где число α характеризует отклонение "уровня среза" s от среднего значения M_y $\alpha\in$[-1,1], по умолчанию обычно принимается α=0.

Множество $J=\{i|y_i\in \mathrm{G}\}$ трактуется как совокупность номеров тех классов, к которым может принадлежать объект, а нормированные на единичную сумму неотрицательные величины

$$f_i = \frac{y_i - s}{\sum_{j\in J}(y_i - s)}$$ (при $i\in J$ и $f = 0$ в противном случае)

интерпретируются как значения функций принадлежности этим классам.

4. *Метод интерполяции* надстраивается над нечеткой классификацией аналогично тому, как метод аккредитации связан с базовым способом. С каждым классом связывается q-мерный выходной вектор z^i. Строится слой из q выходных сумматоров, каждый из которых должен выдавать свою компоненту выходного вектора. Весовые коэффициенты связей, ведущих от того элемента нечеткого классификатора, который вычисляет f_i, к j-му выходному сумматору определяются как z^i_j. В итоге вектор выходных сигналов сети есть

$$z = \sum_i f_i z^i$$

В отдельных случаях по смыслу задачи требуется нормировка f_i на единичную сумму квадратов или модулей.

Выбор одного из описанных четырех вариантов использования сети (или какого-нибудь другого) определяется нуждами пользователя. Предлагаемые четыре способа покрывают большую часть потребностей.

За пределами этой главы остался наиболее универсальный способ обучения нейронных сетей методами гладкой оптимизации - минимизации функции оценки. Ему посвящена следующая глава.

Работа над главой была поддержана Красноярским краевым фондом науки, грант 6*F*0124.

Литература

1. Розенблатт Ф. Принципы нейродинамики. Перцептрон и теория механизмов мозга. М.: Мир, 1965. 480 с.

2. Минский М., Пайперт С. Персептроны. - М.: Мир, 1971.

3. Ивахненко А.Г. Персептроны. - Киев: Наукова думка, 1974.

4. Hopfield J.J. Neural Networks and Physical systems with emergent collective computational abilities//Proc. Nat. Sci. USA. 1982. V.79. P. 2554-2558.

5. Уоссермен Ф. Нейрокомпьютерная техника.- М.: Мир, 1992.

6. Итоги науки и техники. Сер. "Физ. и Матем. модели нейронных сетей" / Под ред. А.А.Веденова. - М.: Изд-во ВИНИТИ, 1990-92 - Т. 1-5.

7. Фролов А.А., Муравьев И.П. Нейронные модели ассоциативной памяти.- М.: Наука, 1987.- 160 с.

8. Судариков В.А. Исследование адаптивных нейросетевых алгоритмов решения задач линейной алгебры // Нейрокомпьютер, 1992. № 3,4. С. 13-20.

9. Кохонен Т. Ассоциативная память. - М.: Мир, 1980.

10. Кохонен Т. Ассоциативные запоминающие устройства. - М.: Мир, 1982.

11. Фор А. Восприятие и распознавание образов.- М.: Машиностроение, 1989.- 272 с.

12. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. Новосибирск: Наука (Сиб. отделение), 1996. 276 с.

13. Кендалл М., Стьюарт А. Статистические выводы и связи.- М.: Наука, 1973.- 900 с.

14. Мостеллер Ф., Тьюки Дж. Анализ данных и регрессия.- М.: Финансы и статистика, 1982.- 239 с.

15. Уидроу Б., Стирнз С. Адаптивная обработка сигналов. М.: Мир, 1989. 440 с.

16. Айвазян С.А., Бежаева З.И., Староверов О.В. Классификация многомерных наблюдений.- М.: Статистика, 1974.- 240 с.

17. Дуда Р., Харт П. Распознавание образов и анализ сцен.- М.: Мир, 1976.- 512 с.

18. Ивахненко А.Г. Самообучающиеся системы распознавания и автоматического регулирования.- Киев: Техника, 1969.- 392 с.

19. Искусственный интеллект: В 3-х кн. Кн. 2. Модели и методы: Справочник / под ред. Д.А. Поспелова.- М.: Радио и связь, 1990.- 304 с.

20. Zurada J. M. Introduction to artificial neural systems. PWS Publishing Company, 1992. 785 P.

21. Горбань А.Н. Обучение нейронных сетей. М.: изд. СССР-США СП "ПараГраф", 1990. 160 с.

Глава 3.

Быстрое дифференцирование, двойственность и обратное распространение ошибки

А.Н.Горбань

Вычислительный центр СО РАН в г.Красноярске[1]

В этой главе излагается материал, значение которого для вычислительной математики и всего того, что по-английски именуют «computer sciences», выходит далеко за пределы нейроинформатики. Со временем он безусловно будет включен в программы университетских курсов математического анализа.

Обсудим одну "очевидную" догму, без разрушения которой было бы невозможно эффективное обучение нейронных сетей. Пусть вычислительные затраты (оцениваемые временем, затраченным некоторым универсальным вычислительным устройством) на вычисление одного значения функции n переменных $H(x_1,...,x_n)$ примерно равны T. Сколько времени потребуется тому же устройству на вычисление grad$H$ (при разумном составлении программы)? Большинство математиков с университетским дипломом ответит:

$$T_{\mathrm{grad}H} \sim nT_H. \qquad (1)$$

Это неверно! Правильный ответ:

$$T_{\mathrm{grad}H} \sim CT_H. \qquad (2)$$

где C - константа, не зависящая от размерности n (в большинстве случаев $C \sim 3$).

Для всех функций многих переменных, встречающихся на практике, необходимые вычислительные затраты на поиск их градиента всего лишь в два-три раза превосходят затраты на вычисление одного значения функции. Это удивительно - ведь координатами вектора градиента служат n частных производных, а затраты на вычисление одной такой производной в общем случае примерно такие же, как и на вычисление значения функции. Почему же вычисление всех их вместе дешевле, чем по-отдельности?

«Чудо» объясняется довольно просто: нужно рационально организовать вычисление производных сложной функции многих переменных, избегая

[1] 660036, Красноярск-36, ВЦК СО РАН. E-mail: gorban@cc.krascience.rssi.ru

дублирования. Для этого необходимо подробно представить само вычисление функции, чтобы потом иметь дело не с «черным ящиком», преобразующим вектор аргументов в значение функции, а с детально описанным графом вычислений.

Поиск grad$H$ удобно представить как некоторый двойственный процесс над структурой вычисления H. Промежуточные результаты, появляющиеся при вычислении градиента, являются ни чем иным, как множителями Лагранжа. Оказывается, что если представить H как сложную функцию, являющуюся суперпозицией функций малого числа переменных (а по-другому вычислять функции многих переменных мы не умеем), и аккуратно воспользоваться правилом дифференцирования сложной функции, не производя по дороге лишних вычислений и сохраняя полезные промежуточные результаты, то вычисление всей совокупности $\partial H/\partial x_i$ (i=1,...,$\underline{n}$) немногим сложнее, чем одной из этих функций - они все собраны из одинаковых блоков.

Я не знаю, кто все это придумал первым. В нейроинформатике споры о приоритете ведутся до сих пор. Конец переоткрытиям положили две работы 1986 г.: Румельхарта, Хинтона и Вильямса [1,2] и Барцева и Охонина [3]. Однако первые публикации относятся к 70-м и даже 60-м годам нашего столетия. По мнению В.А.Охонина, Лагранж и Лежандр также вправе претендовать на авторство метода.

1. Обучение нейронных сетей как минимизация функции ошибки

Построение обучения как оптимизации дает нам универсальный метод создания нейронных сетей для решения задач. Если сформулировать требования к нейронной сети, как задачу минимизации некоторой функции - оценки, зависящей от части сигналов (входных, выходных, ...) и от параметров сети, то обучение можно рассматривать как оптимизацию и строить соответствующие алгоритмы, программное обеспечение и, наконец, устройства (hardware). Функция оценки обычно довольно просто (явно) зависит от части сигналов - входных и выходных. Ее зависимость от настраиваемых параметров сети может быть сложнее и включать как явные компоненты (слагаемые, сомножители,...), так и неявные - через сигналы (сигналы, очевидно, зависят от параметров, а функция оценки - от сигналов).

За пределами задач, в которых нейронные сети формируются по явным правилам (сети Хопфилда, проективные сети, минимизация аналитически заданных функций и т.п.) нам неизвестны случаи, когда требования к нейронной сети нельзя было бы представить в форме минимизации функции оценки. Не следует путать такую

постановку задачи и ее весьма частный случай - "обучение с учителем". Уже метод динамических ядер, описанный в предыдущей главе, показывает, каким образом обучение без учителя в задачах классификации может быть описано как минимизация целевой функции, оценивающей качество разбиения на классы.

Если для решения задачи не удается явным образом сформировать сеть, то проблему обучения можно, как правило, сформулировать как задачу минимизации оценки. Осторожность предыдущей фразы ("как правило") связана с тем, что на самом деле нам неизвестны и никогда не будут известны все возможные задачи для нейронных сетей, и, быть может, где-то в неизвестности есть задачи, которые несводимы к минимизации оценки.

Минимизация оценки - сложная проблема: параметров астрономически много (для стандартных примеров, реализуемых на PC - от 100 до 1000000), адаптивный рельеф (график оценки как функции от подстраиваемых параметров) сложен, может содержать много локальных минимумов, извилистых оврагов и т.п.

Наконец, даже для того, чтобы воспользоваться простейшими методами гладкой оптимизации, нужно вычислять градиент функции оценки. Здесь мы сталкиваемся с одной "очевидной" догмой, без разрушения которой было бы невозможно эффективное обучение нейронных сетей.

2. Граф вычисления сложной функции

Сложная функция задается с помощью суперпозиции некоторого набора «простых». Простые функции принадлежат исходно задаваемому конечному множеству *F*. Формально они выделены только принадлежностью к множеству *F* - никаких обязательных иных отличий этих функций от всех прочих в общем случае не предполагается.

В этом разделе мы изучим способы задания сложных функций, то есть формулы и соответствующие им графы. Свойства самих функций нас сейчас не интересуют (будем рассматривать ноты, а не слушать музыку).

Теория выражений, определяющих сложные функции, является простейшим разделом математической логики, в которой сами эти выражения называются *термами* [4].

Термы - это правильно построенные выражения в некотором формальном языке. Чтобы задать такой язык, необходимо определить его *алфавит*. Он состоит из трех множеств символов:

1) C - множество символов, обозначающих константы;

2) V - множество символов, обозначающих переменные;

3) F - множество функциональных символов, $F = \bigcup_{k=1}^{\infty} F_k$, где F_k - множество символов для обозначения функций k переменных.

Вводя различные множества символов, мы постоянно обращаемся к их *интерпретации* («... символы, обозначающие...»). Строго говоря, это не нужно - можно (а с чисто формальной точки зрения - и должно) описать все правила действия с символами, не прибегая к их интерпретации, однако ее использование позволяет сократить изложение формальных правил за счет обращения к имеющемуся содержательному опыту.

Термы определяются индуктивно:

1) любой символ из $C \cup V$ есть терм;

2) если $t_1,\ldots,t_k$ - термы и $f \in F_k$, то $ft_1\ldots t_k$ - терм.

Множество термов T представляет собой объединение:

$$T = \bigcup_{i=0}^{\infty} T_i ,$$

где $T_0 = C \cup V$,

$$T_{i+1} = \bigcup_{k=1}^{\infty} \bigcup_{f \in F_k} \{ft_1\ldots t_k | t_1,\ldots,t_k \in \bigcup_{j=0}^{i} T_j\} \quad (T_1 \subseteq T_2 \subseteq \ldots \subseteq T_i \subseteq T_{i+1} \subseteq \ldots).$$

Удобно разбить T на непересекающиеся множества - слои $S_{i+1} = T_{i+1} \setminus T_i$ ($S_0 = T_0$). Элементы S_i будем называть термами глубины i или i-слойными термами. Множеству S_i принадлежат выражения, обозначающие те функции от термов предыдущих слоев $S_0,\ldots,S_{i-1}$, которые сами им не принадлежат[2].

Для оперирования с термами очень полезны две теоремы [4].

---

[2] Трудно удержаться от вольности речи - обращения к формально еще не введенной, но совершенно очевидной интерпретации («... обозначающие...»).

Теорема 1 (о построении термов). Каждый терм t единственным образом представляется в виде $ft_1 \ldots t_k$, где f - первый символ в t, $f \in F$, число k определяется по f ($f \in F_k$), а $t_1, \ldots, t_k$ - термы.

Эта теорема является точной формулировкой эквивалентности используемой бесскобочной и обычной записи.

Пусть u и v - выражения, то есть последовательности символов алфавита. Скажем, что u входит в v, если существуют такие выражения p и q (возможно, пустые), что v совпадает с puq.

Теорема 2 (о вхождении терма в терм). Пусть $f \in F_k$, $t_1, \ldots, t_k$ - термы, t представляется в виде $ft_1 \ldots t_k$, τ - терм и τ входит в t. Тогда или τ совпадает с t, или τ входит в одно из t_i $(i = 1, \ldots, k)$.

Доказываются эти теоремы элементарной индукцией по длине термов [4]. В доказательстве теоремы 2 выделяется лемма, представляющая и самостоятельный интерес.

Лемма 1. Каждое вхождение любого символа в терм τ начинает вхождение некоторого терма в τ.

Определим отношение между термами $t_1 \le t_2$ индуктивным образом «сверху вниз» - по глубине вхождения:

1) $t \le t$;

2) если t совпадает с $ft_1 \ldots t_k$, $f \in F_k$ и $t_1, \ldots, t_k$ - термы, то $t_1, \ldots, t_k \le t$;

3) если $t_1 \le t$ и $t \le t_2$, то $t_1 \le t_2$.

Согласно теореме 2, $t_1 \le t_2$ тогда и только тогда, когда t_1 входит в t_2.

Для каждого терма t определим множество входящих в него термов $S^t = \{\tau | \tau \le t\}$. Если $t \in S_i$, то при $0 \le k \le i$ непусты множества $S_k^t = S^t \cap S_k$. При этом множество S_i^t состоит из одного элемента - исходного терма t.

Свяжем с термом t ориентированный граф G_0^t с вершинами, взаимнооднозначно соответствующими термам из S^t. Будем одинаково обозначать вершины и соответствующие им термы. Пара вершин (τ_1, τ_2) образует ориентированное от τ_1 к τ_2

ребро графа G_0^t, если терм τ_2 имеет вид $ft_1\ldots t_k$, $f \in F_k$, $t_1,\ldots,t_k$ - термы и один из них t_i $(i = 1,\ldots,k)$ совпадает с τ_1. Вершины графа G_0^t удобно располагать по слоям S_i^t.

Для произвольного графа G будем обозначать v(G) множество вершин, а e(G) - множество ребер G.

Возьмем для примера выражение для сложной функции

$$\varphi(x_1, x_2, x_3) = f_5(f_3(x_1, f_1(x_1, x_2)), f_4(f_1(x_1, x_2), f_2(x_2, x_3))). \qquad (3)$$

В принятой выше бесскобочной префиксной записи оно имеет вид

$$f_5 f_3 x_1 f_1 x_1 x_2 f_4 f_1 x_1 x_2 f_2 x_2 x_3, \qquad (3')$$

где все функциональные символы принадлежат F_2.

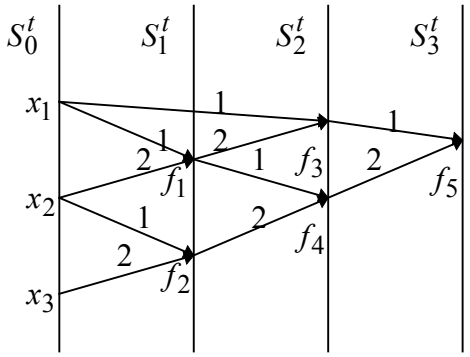


Рис. 1. Пример графа G_0^t. Около вершин и над ребрами указаны соответствующие метки

Граф G_0^t для этого терма изображен на рис. 1.

Для того, чтобы терм однозначно восстанавливался по графу, необходимы еще два дополнения.

1. Сопоставим каждой вершине $\tau \in$ v(G_0^t) метку $p(\tau)$ - символ алфавита. Если вершина принадлежит нулевому слою S_0^t, то ей соответствует терм, совпадающий с символом из $C \cup V$. Этот символ и сопоставляется вершине в качестве метки. Если вершина принадлежит S_i^t (i>0), то меткой служит функциональный символ: вершине τ сопоставляется $f \in F$, если τ имеет вид $ft_1\ldots t_k$, где $f \in F_k$, а $t_1,\ldots,t_k$ - термы.

2. Каждому ребру $(\tau',\tau) \in$ e(G_0^t), приходящему в вершину τ, сопоставим метку $P(\tau',\tau)$ - конечное множество натуральных чисел (номеров): пусть терм τ имеет вид $ft_1\ldots t_k$, где $f \in F_k$, а $t_1,\ldots,t_k$ - термы, тогда ребру (τ',τ) сопоставляется множество тех i ($1 \le i \le k$), для которых τ' совпадает с t_i. На практике в большинстве случаев эта метка состоит из одного номера, но возможны и другие варианты - так же, как функции вида $f(x,x)$. Для графических иллюстраций удобно ребра (τ',τ), имеющие в своей метке $P(\tau',\tau)$ больше одного номера, рисовать как пучок ребер, идущих от вершины τ' к вершине τ - по

одному такому ребру для каждого номера из $P(\tau',\tau)$; этот номер и будет меткой соответствующего ребра из пучка.

Граф G_0^t вместе со всеми метками будем обозначать G^t. На рис. 1 указаны соответствующие метки для разобранного примера.

Итак, для всякого терма t построен ориентированный граф G_0^t и две функции: первая сопоставляет каждой вершине $\tau \in \mathrm{v}(G_0^t)$ символ алфавита $p(\tau) \in C \cup V \cup F$, вторая (обозначим ее P) - каждому ребру $(\tau',\tau) \in \mathrm{e}(G_0^t)$ - конечное множество натуральных чисел $P(\tau',\tau)$. Отмеченный граф - набор (G_0^t, p, P) обозначаем G^t. Функции p и P удовлетворяют следующему ограничению:

А) если для данного $\tau \in S^t$ множество входящих ребер (τ',τ) непусто, то $p(\tau) = f^\tau \in F_k$ (является k-местным функциональным символом при некотором k) и семейство множеств

$$\{P(\tau',\tau) | (\tau',\tau) \in \mathrm{e}(G_0^t)\}$$

при фиксированном τ образует разбиение множества номеров $\{1,...,k\}$, то есть

$$P(\tau',\tau) \cap P(\tau'',\tau) = \varnothing \text{ при } \tau' \neq \tau'',$$

$$\forall \tau \qquad \bigcup_{\tau' :\, (\tau',\tau) \in \mathrm{e}(G_0^t)} P(\tau',\tau) = \{1,\ldots,k\}.$$

На этом завершается изложение основных формальных конструкций. Прежде, чем переходить к интерпретации, сформулируем теорему об эквивалентности графического и формульного представления термов.

Пусть G - конечный ориентированный граф, не имеющий ориентированных циклов, и в G существует и единственна такая вершина τ^*, к которой от любой вершины ведет ориентированный путь. Пусть, далее, заданы две функции: p - на множестве вершин G со значениями в множестве символов алфавита и P - на множестве ребер G со значениями в множестве конечных наборов натуральных чисел и выполнено условие А.

Теорема 3. Существует и единственен терм t, для которого $G^t = (G,p,P)$.

Доказательство проводится в два этапа. Сначала в *G* устанавливается послойная структура: строится разбиение множества вершин *G*: $s_0 \cup s_1 \cup ... \cup s_k$. Множество s_0 состоит из тех вершин, к которым не ведет ни одного ребра - из-за отсутствия ориентированных циклов такие вершины существуют. Множество s_{i+1} состоит из тех вершин, к которым ведут ребра только из элементов $s_0 \cup s_1 \cup ... \cup s_i$. Последний непустой элемент в последовательности $s_0, s_1, ..., s_k$ состоит из одной вершины τ^*, все предшествующие элементы этой последовательности непусты, а объединение всех s_i содержит все вершины *G*.

Доказательство основного утверждения теоремы проводится индукцией по числу слоев *k*.

Интерпретация сопоставляет терму сложную функцию. Она строится так. Задается некоторое множество *D* - область интерпретации. Каждой константе *c*, входящей в интерпретируемый терм *t*, сопоставляется элемент из *D* ($c \in D$), каждому *k*-местному функциональному символу *f*, входящему в *t*, сопоставляется функция *k* переменных $f: D^k \to D$ (мы сохраняем одинаковое обозначение для символов и их интерпретации, это вполне соответствует интуиции и не должно приводить к путанице). Каждой переменной, входящей в интерпретируемый терм *t*, сопоставляется переменная, пробегающая *D*. В результате терму *t* сопоставляется функция *n* переменных $F^t: D^n \to D$, где *n* - число различных символов переменных, входящих в *t*. Эта «сложная» функция получается суперпозицией «простых» функций, соответствующих функциональным символам.

Если $t \in S_0$, то есть терм является константой или переменной, то вместо сложной функции F^t получаем константу или тождественную функцию id, переводящую значение переменной в него же. Если $t \in S_1$, то соответствующая функция является «простой». Истинные сложные функции появляются, начиная со слоя S_2.

Заметим, что заданная интерпретация терма *t* одновременно определяет интерпретацию каждого терма, входящего в *t*.

Представим процесс вычисления сложной функции F^t с помощью отмеченного графа G^t. Строится два представления: статическое (все результаты промежуточных вычислений располагаются на графе) и динамическое (шаг за шагом, слой за слоем).

Пусть задана интерпретация терма t и определены значения всех переменных, входящих в t. Тогда для любого терма τ, входящего в t, также задана интерпретация и определены значения всех функций F^τ $(\tau \le t)$. Каждой вершине τ, входящей в граф G_0^t, сопоставляется значение функции F^τ - элемент D. При этом вершинам нулевого слоя соответствуют значения переменных и констант, а единственной вершине последнего (выходного) слоя - значение F^t. Будем называть элементы D, соответствующие вершинам, *значениями* этих вершин и обозначать их $Z(\tau)$.

Для каждой вершины τ, принадлежащей ненулевому слою, можно выписать *уравнение функционирования*, связывающее значения вершин. Пусть $p(\tau)= f^\tau$, $f^\tau \in F_k$ и $\mathrm{in}(\tau)$ - совокупность входящих в τ ребер. Напомним, что совокупность меток ребер, оканчивающихся в τ,

$$\{P(\tau',\tau)|(\tau',\tau)\in \mathrm{in}(\tau)\}$$

образует разбиение множества $\{1,2,...,k\}$.

Уравнение функционирования для вершины τ, принадлежащей ненулевому слою, имеет вид

$$Z(\tau) = f^\tau(z_1,...,z_k),\ z_i = Z(\tau') \text{ при } i \in P(\tau',\tau),\ (\tau',\tau)\in \mathrm{in}(\tau). \qquad (4)$$

В силу уравнения функционирования (4), если для входящих в τ ребер $(\tau',\tau)\in \mathrm{in}(\tau)$ известны значения $Z(\tau')$ и задана интерпретация символа $p(\tau)= f^\tau$ - метки вершины, то можно найти значение вершины $Z(\tau)$. На основании этого (очевидного) замечания строится динамическое представление вычисления сложной функции.

С каждой вершиной графа τ, принадлежащей ненулевому слою, ассоциируется автомат, вычисляющий функцию $f^\tau(z_1,...,z_k)$, где $f^\tau \in F_k$ - метка вершины τ. Автоматы срабатывают по слоям в дискретные моменты времени (такты) - автоматы i-го слоя в i-й момент времени. В начальный момент сформированы значения вершин нулевого слоя - известны значения переменных и констант. Они поступают на входы автоматов первого слоя в соответствии с нумерацией аргументов. После i-го такта функционирования определены значения вершин, принадлежащих слоям $S_0,...,S_i$. На

i+1-м такте автоматы i+1-го слоя вычисляют значения вершин i+1-го слоя, получая входные сигналы с предыдущих слоев по правилу (4) - в соответствии с метками входящих ребер.

3. Вычисления на ориентированном графе

Для дальнейшего полезно обобщение изложенных конструкций, предназначенное для описания одновременного вычисления нескольких сложных функций.

Пусть G - связный (но не обязательно ориентированно связный) ориентированный граф без ориентированных циклов. Как и выше, множество вершин G обозначаем $\mathrm{v}(G)$, множество ребер - $\mathrm{e}(G)$. Пусть, далее, каждой вершине $\tau \in \mathrm{v}(G)$ сопоставлена метка - символ алфавита $p(\tau)$, а каждому ребру $(\tau', \tau) \in \mathrm{e}(G)$ сопоставляется метка - конечное множество натуральных чисел $P(\tau', \tau)$ и выполнено условие согласования А: если для данного $\tau \in \mathrm{v}(G)$ множество входящих ребер (τ', τ) непусто, то $p(\tau) = f^{\tau} \in F_k$ (является k-местным функциональным символом при некотором k) и семейство множеств $\{P(\tau', \tau) | (\tau', \tau) \in \mathrm{e}(G)\}$ при фиксированном τ образует разбиение множества номеров $\{1,...,k\}$.

С помощью транзитивного замыкания G устанавливаем на множестве его вершин $\mathrm{v}(G)$ отношения частичного порядка: $\tau \geq \tau'$, если существует ориентированный путь, ведущий от τ' к τ. Из-за отсутствия ориентированных циклов это отношение антисимметрично: если $\tau \geq \tau'$ и $\tau' \geq \tau$, то τ совпадает с τ'. Минимальные вершины (к которым не ведет ни одного ребра) называем *входными*, а максимальные (от которых не идет ни одного ребра) - *выходными*. Обозначим множество входных вершин v_{in}, а выходных - $\mathrm{v}_{\mathrm{out}}$. Метки входных вершин являются символами переменных или констант, метки остальных вершин - функциональные символы.

Определим послойную структуру: $\mathrm{v}(G) = s_0 \cup s_1 \cup ... \cup s_k$. Нулевой слой состоит из минимальных вершин, первый слой из минимальных вершин графа, получаемого удалением из G нулевого слоя и выходящих из него ребер и т.д. - i+1-й слой состоит из минимальных вершин графа, получаемого удалением из G всех вершин объединения $s_0 \cup s_1 \cup ... \cup s_i$ и содержащих их ребер. Последний слой состоит только из выходных вершин. Предыдущие слои также могут содержать выходные вершины.

С каждой вершиной графа $\tau \in \mathrm{v}(G)$ однозначно связан терм (который мы, следуя традиции предыдущего раздела, также будем обозначать τ). Для его построения удалим

из G все вершины, кроме тех, от которых к τ можно пройти по ориентированному пути, а также связанные с ними ребра. Полученный граф обозначим G^{τ}: $v(G^{\tau})=\{\tau' \in v(G) | \tau \geq \tau'\}$. Граф G^{τ} удовлетворяет условиям теоремы 3, поэтому по нему единственным образом восстанавливается терм (и соответствующая сложная функция).

Пусть задано множество D - область интерпретации и указана интерпретация всех символов, отмечающих вершины графа, а также значения переменных, отвечающих входным вершинам. Тогда по уравнениям функционирования (4) (они полностью сохраняются и для рассматриваемых графов) можно определить значения $Z(\tau)$ для всех вершин графа. В результате определяются значения всех сложных функций, формулы для которых являются термами, соответствующими вершинам графа G.

Описанные общие графы могут появляться в различных ситуациях. Для дальнейшего важно то, что такие графы возникают, если из графа вычисления сложной функции удалить один или несколько последних слоев.

4. Двойственное функционирование, дифференциальные операторы и градиент сложной функции

А. Производная сложной функции одного переменного

Основную идею двойственного функционирования можно понять уже на простейшем примере. Рассмотрим вычисление производной сложной функции одного переменного. Пусть заданы функции одного переменного $f_1(A)$,$f_2(A)$,...,$f_n(A)$. Образуем из них сложную функцию

$F(x)=f_n(f_{n-1}(...(f_1(x))...))$. (1)

Можно представить вычисление $F(x)$ как результат работы n автоматов, каждый из которых имеет один вход и выдает на выходе значение $f_i(A)$, где A - входной сигнал (рис.2, а). Чтобы построить систему автоматов, вычисляющую $F'(x)$, надо дополнить исходные автоматы такими, которые вычисляют функции $f_i'(A)$, где A - входной сигнал (важно различать производную f_i по входному сигналу, то есть по аргументу функции f_i, и производную сложной функции $f_i(A(x))$ по x; $f_i'(A)$ - производные по A).

Для вычисления $F'(x)$ потребуется еще цепочка из n-1 одинаковых автоматов, имеющих по два входа, по одному выходу и подающих на выход произведение входов. Тогда формулу производной сложной функции

$$\frac{dF}{dx} = f_n'(f_{n-1}(\ldots(f_1(x)\ldots)) \times f_{n-1}'(\ldots(f_1(x)\ldots)\times\ldots\times f_1'(x)$$

можно реализовать с помощью сети автоматов, изображенной на рис. 2, б. Сначала по этой схеме вычисления идут слева направо: на входы f_1 и f_1' подаются значения x, после вычислений $f_1(x)$ это число подается на входы f_2 и f_2' и т.д. В конце цепочки оказываются вычисленными все f_i (f_{i-1} (...)) и f_i'(f_{i-1} (...)).

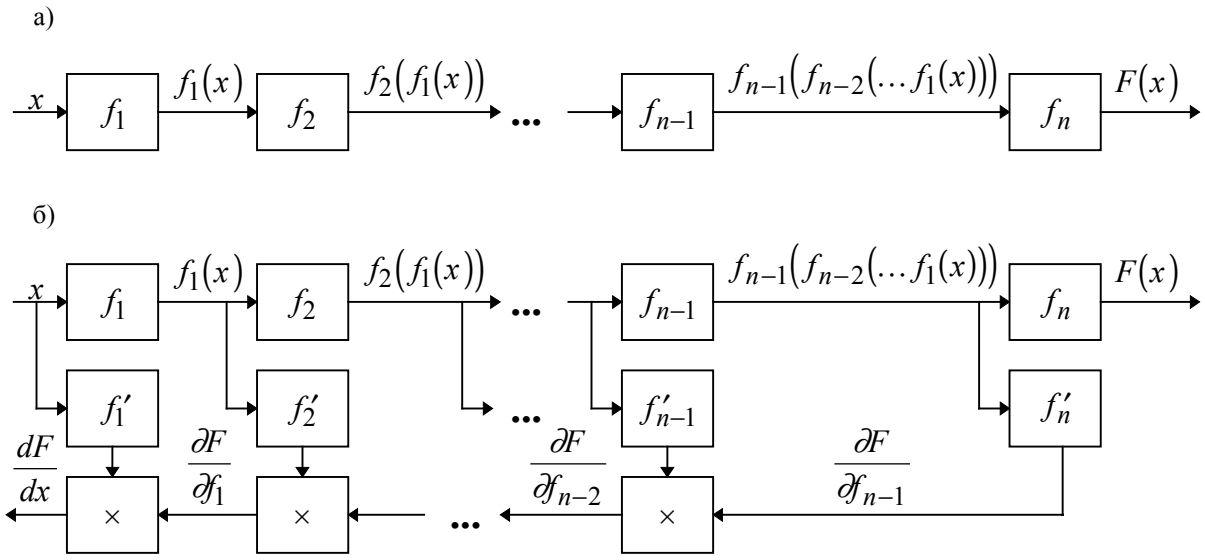


Рис.2. Схематическое представление вычисления сложной функции одного переменного и ее производных.

Тем самым, для каждого автомата нижней строки, кроме самого правого, оказывается сформированным по одному значению входа, а у самого правого - оба, поэтому он может сработать и начать передачу сигнала в обратном направлении - справа налево. Это обратное движение есть последовательное вычисление попарных произведений и передача их налево. В конце получаем *dF*/*dx*.

Б. Двойственное функционирование и быстрое дифференцирование

Вычисление градиента сложной функции многих переменных также будет представлено как некоторый вычислительный процесс, в ходе которого сигналы перемещаются в обратном направлении - от выходных элементов графа к входным. И так же, как при вычислении сложной функции, будет явно представлено прохождение каждого узла (подобно тому, как это сделано в уравнении функционирования (4)), а весь процесс в целом будет складываться из таких элементарных фрагментов за счет

структуры связей между узлами. Сама эта структура - та же самая, что и для прямого процесса.

Всюду в этом разделе область интерпретации - действительная прямая, а все функции дифференцируемы.

Основной инструмент при построении двойственного функционирования - формула для дифференцирования «двухслойной» сложной функции нескольких переменных

$$\frac{\partial f(A_1(x_1,\ldots,x_n),A_2(x_1,\ldots,x_n),\ldots,A_k(x_1,\ldots,x_n))}{\partial x_i}=\sum_{j=1}^{k}\frac{\partial f}{\partial A_j}\times\frac{\partial A_j}{\partial x_i}. \qquad (5)$$

При проведении индукции по числу слоев нам придется использовать графы с несколькими выходными вершинами, описанные в предыдущем разделе, поэтому сразу будем рассматривать этот более общий случай.

Итак, пусть G - связный (но не обязательно ориентированно связный) ориентированный граф без ориентированных циклов. Как и выше, v(G) - множество вершин G, e(G) - множество ребер. Пусть, далее, каждой вершине $\tau\in$ v(G) сопоставлена метка - символ алфавита $p(\tau)$, а каждому ребру $(\tau',\tau)\in$ e(G) сопоставляется метка - конечное множество натуральных чисел $P(\tau',\tau)$ и выполнено условие согласования A: если для данного $\tau\in$ v(G) множество входящих ребер (τ',τ) непусто, то $p(\tau)=f^{\tau}\in F_k$ (является k-местным функциональным символом при некотором k) и семейство множеств $\{P(\tau',\tau)|(\tau',\tau)\in e(G)\}$ при фиксированном τ образует разбиение множества номеров $\{1,...,k\}$. Для каждой вершины $\tau\in$ v(G) обозначим множество входящих в нее ребер in(τ), а выходящих из нее - out(τ).

Пусть указана интерпретация всех символов, отмечающих вершины графа, значения переменных, отвечающих входным вершинам и уравнениям функционирования (4) определены значения $Z(\tau)$ для всех вершин графа. В результате определены значения всех сложных функций, формулы для которых являются термами, соответствующими вершинам графа G. Процесс вычисления $Z(\tau)$ будем называть *прямым* в противовес *обратному* или *двойственному*, к описанию которого мы переходим.

При заданных $Z(\tau)$ для каждой вершины и каждого ребра строятся *переменные двойственного функционирования* (или, более кратко, *двойственные переменные*). Будем обозначать их $\mu(\tau)$ для вершин и $\mu(\tau',\tau)$ - для ребер.

Для выходных вершин $\mu(\tau)$ являются независимыми переменными. Пусть их значения заданы. Для вершины τ, не являющейся выходной, значение $\mu(\tau)$ есть сумма значений двойственных переменных, соответствующих выходящим из τ ребрам:

$$\mu(\tau) = \sum_{(\tau,\tau')\in \mathrm{out}(\tau)} \mu(\tau,\tau') . \tag{6}$$

Для ребра (τ',τ) значение $\mu(\tau',\tau)$ определяется согласно формуле (5):

$$\mu(\tau',\tau) = \mu(\tau) \sum_{i\in P(\tau',\tau)} \frac{\partial f^{\tau}(t_1,\ldots,t_k)}{\partial t_i} . \tag{7}$$

В формуле (7) $f^{\tau} = p(\tau) \in F_k$ - метка ребра τ, $t_1,\ldots,t_k$ - аргументы «простой» функции f^{τ}, а производные f^{τ} берутся при условиях, определяемых прямым процессом:

$$t_j = Z(\theta) \text{ при } \theta \in \mathrm{in}(\tau),\ j \in P(\theta,\tau) . \tag{8}$$

Для каждого $j\in\{1,\ldots,k\}$ такое θ существует и единственно в силу того, что метки входящих в вершину τ ребер образуют разбиение множества номеров $\{1,\ldots,k\}$.

В самом распространенном случае все метки ребер $P(\tau',\tau)$ содержат по одному элементу. В этом случае формула (7) приобретает особенно простой вид

$$\mu(\tau',\tau) = \mu(\tau) \frac{\partial f^{\tau}(t_1,\ldots,t_k)}{\partial t_{i(\tau',\tau)}} \text{ где } P(\tau',\tau) = \{i(\tau',\tau)\} . \tag{7'}$$

Используя (6)-(8), можно записать правила вычисления двойственных переменных для вершин, не использующие двойственных переменных для ребер:

$$\mu(\tau)\left(= \sum_{(\tau,\theta)\in \mathrm{out}(\tau)} \mu(\tau,\theta)\right) = \sum_{(\tau,\theta)\in \mathrm{out}(\tau)} \mu(\theta) \sum_{i\in P(\tau,\theta)} \frac{\partial f^{\theta}(t_1,\ldots,t_k)}{\partial t_i}, \tag{9}$$

где производные $f^{\theta} = p(\theta) \in F_k$ берутся при условиях

$$t_j = Z(\eta) \text{ при } \eta \in \mathrm{in}(\theta),\ j \in P(\eta,\theta) . \tag{10}$$

Греческими буквами τ, θ, η здесь обозначены вершины графа.

Опять же, в распространенном случае, когда все $P(\tau',\tau)$ одноэлементны, применяем (7′) вместо (7) и получаем

$$\mu(\tau)\left(= \sum_{(\tau,\theta)\in \mathrm{out}(\tau)} \mu(\tau,\theta)\right) = \sum_{(\tau,\theta)\in \mathrm{out}(\tau)} \mu(\theta) \frac{\partial f^{\theta}(t_1,\ldots,t_k)}{\partial t_{i(\tau,\theta)}} \text{ где } P(\tau,\theta) = \{i(\tau,\theta)\} \tag{9'}$$

Напомним, что каждой вершине τ, принадлежащей ненулевому слою графа G, соответствуют терм τ и сложная функция F^τ от независимых переменных и констант, отмечающих вершины нулевого слоя G.

Процесс вычисления двойственных переменных организуется послойно от выходных вершин к входным и часто называется процессом *обратного распространения* (back propagation) или просто обратным процессом.

Теорема 5. Пусть задана интерпретация всех символов, отмечающих вершины графа G, определены значения независимых переменных (а также констант), соответствующих вершинам входного слоя v_{in} и значения независимых переменных двойственного функционирования $\mu(\tau)$ для вершин выходного слоя $\mathrm{v}_{\mathrm{out}}$. Тогда для любого $\theta \in \mathrm{v}_{\mathrm{in}}$

$$\mu(\theta) = \sum_{\tau \in \mathrm{v}_{\mathrm{out}}} \mu(\tau) \frac{\partial F^\tau}{\partial x_\theta}. \qquad (11)$$

где F^τ - соответствующая вершине τ функция от независимых переменных и констант, отмечающих вершины нулевого слоя G, x_θ - соответствующая вершине $\theta \in \mathrm{v}_{\mathrm{in}}$ переменная или константа, а производные в (11) берутся при фиксированных значениях прочих переменных и констант.

Доказательство проводится индукцией по числу слоев. Для графов из двух слоев - нулевого и первого - теорема очевидна. Спуск от i+1-слойных графов к i-слойным производится с помощью формулы 5.

Прохождение вершины графа и прилегающих к ней ребер при прямом и обратном процессах проиллюстрировано на рис. 3.

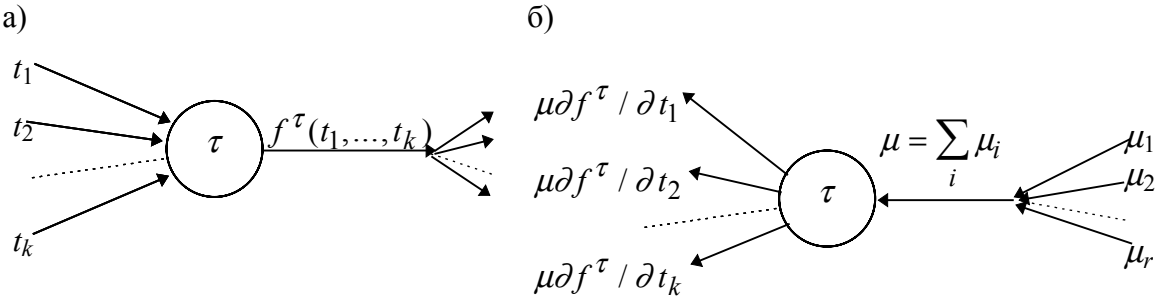


Рис. 3. Прохождение вершины τ в прямом (а) и обратном (б) направлении.

5. Сложность вычисления функции и ее градиента

Подсчитаем теперь число операций, необходимых для вычисления всех двойственных переменных $\mu(\tau)$ для вершин и $\mu(\tau',\tau)$ - для ребер.

Во-первых, нужно вычислить все частные производные

$$\frac{\partial f^{\theta}(t_1,\ldots,t_k)}{\partial t_i}$$

для всех вершин θ и всех k аргументов «простой» функции, соответствующей каждой вершине. Число таких производных равно сумме числа аргументов для всех функциональных символов, соответствующих вершинам графа, то есть следующей величине E:

$$E = \sum_{(\tau,\theta)\in e(G)} P(\tau,\theta).$$

Договоримся в этом разделе отображать ребра (τ',τ), имеющие в своих метках $P(\tau',\tau)$ больше одного номера, как пучки ребер, идущих от вершины τ' к вершине τ - по одному такому ребру для каждого номера из $P(\tau',\tau)$. Число E просто равно числу ребер в графе. Число необходимых умножений и число сложений также не превосходят E.

Количество вычислений «простых» функций при вычислении сложной равно числу вершин графа. Обозначим его V. Отношение E/V дает представление об отношении вычислительных затрат на вычисление градиента к затратам на вычисление функции. Чтобы последовательно использовать эту оценку, а также искать те функции, для которых вычисление градиента еще проще, необходимо зафиксировать исходные предположения. Будем обозначать T_f затраты на вычисление f.

1. Для каждой вершины графа, соответствующей ей функции f, и любого аргумента этой функции x справедлива оценка $T_f \cong T_{\partial f/\partial x}$;
2. Для функций $f = f^{\tau}(z_1,\ldots,z_k)$, соответствующих вершинам графа, $T_f \geq T_{\Sigma}$, $\Sigma = \sum_{i=1}^{k} z_i$, то есть сумма переменных - простейшая функция;
3. Умножение и сложение имеют примерно одинаковую сложность.

В предположениях 1-3 зафиксирован тот уровень точности, с которым будут делаться оценки. При формальном использовании отношения «a примерно равно b» неизбежны парадоксы типа «куча»: один камень не куча, если n камней - не куча, то и n+1 - тоже, следовательно... . Чтобы избежать этого и сделать рассуждения более наглядными, поступим следующим образом. Сложность «простой» функции k

переменных и любой ее частной производной оценим как ck, где c - некоторая константа, $c \geq 1$; сложность суммы k слагаемых (и произведения k сомножителей) определим как k-1. Последнее вычитание единицы имеет смысл при рассмотрении большого числа сумм, содержащих мало слагаемых.

Пусть, как и выше, E- число ребер графа, V- число вершин, V_{out} - число выходных вершин (им не соответствует ни одной суммы в (6)). Сложность прямого прохождения графа (вычисления функций) оценивается как cE.

Обратное прохождение графа при вычислении градиентов складывается из трех слагаемых:

1. Вычисление всех частных производных простых функций, соответствующих вершинам графа. Необходимо вычислить E таких производных. Сложность вычисления одной такой производной для вершины τ, имеющей $|\text{in}(\tau)|$ входящих ребер, оценивается как $c|\text{in}(\tau)|$. Оценка сложности вычисления всех этих производных дается следующей суммой $T_{\text{Dif}} = c\sum_{\tau}|\text{in}(\tau)|^2$ $(\leq cE^2)$.
2. Вычисление всех произведений (7) на ребрах - E произведений (в связи с тем, что мы в данном разделе передачу сигнала на разные входы автомата, вычисляющего $f^{\tau}(z_1,\ldots,z_k)$, обозначаем различными ребрами, уравнения (7), сохраняя прежнюю внешнюю форму, преобразуются так, что в суммах остается по одному слагаемому, остальное суммирование переносится к вершинам (6)).
3. Вычисление всех сумм (6) - сложность равна E-(V- V_{out}).

Итого, полная сложность обратного прохождения сигналов оценивается как

$$T = T_{\text{Dif}} + 2E - (V - V_{\text{out}}) = c\sum_{\tau}|\text{in}(\tau)|^2 + 2E - (V - V_{\text{out}}).$$

Основную роль в последней сумме играет первое слагаемое - именно им определяется сложность обратного распространения по графу (если все $|\text{in}(\tau)|=1$, то $T_{\text{Dif}} = cE$, а уже в случае, когда $|\text{in}(\tau)|=2$, мы получаем $T_{\text{Dif}} = 2cE$). Поэтому можно положить $T \approx T_{\text{Dif}}$ (строго говоря, $3T_{\text{Dif}} \geq T \geq T_{\text{Dif}}$, однако различия в два-три раза для нас сейчас несущественны).

Если характерное число переменных у простых функций, соответствующих вершинам графа, равно m (то есть $|\text{in}(\tau)|=m$), то для вычисляемой по графу сложной

функции F можно оценить отношение затрат на вычисление F и grad$F$ следующим образом:

$$T_{\mathrm{grad}F} \cong c\frac{E}{k}k^2 = ckE,\ T_F \cong cE,\ \frac{T_{\mathrm{grad}F}}{T_F} \cong k\,.$$

Эта оценка, конечно, лучше, чем полное число переменных n, но все же искомое отношение оценивается примерно как отношение числа ребер к числу вершин E/V (с точностью до менее значимых слагаемых). Если нас интересуют графы с большим числом связей, то для сохранения эффективности вычисления градиентов нужно ограничиваться специальными классами простых функций. Из исходных предположений 1-3 наиболее существенно первое ($T_f \cong T_{\partial f/\partial x}$). Зададимся вопросом: для каких функций f вычисление частных производных вообще не требует вычислений?

Ответ очевиден: общий вид таких функций

$$f(z_1,\ldots z_n) = \sum_{i\in P_1} z_i + \sum_{j\in P_2, k\in P_3} z_j z_k\,, \qquad (12)$$

где множества индексов P_1, P_2, P_3 не пересекаются и, кроме того, P_2, P_3 имеют одинаковое число элементов. Значения всех частных производных таких функций уже известны, а источники (адреса) этих значений указываются связями графа. Какие-то значения z_k во второй сумме могут быть константами (значения нулевого слоя), какие-то - независимыми переменными (также нулевой слой) или значениями, найденными при промежуточных вычислениях.

В общем случае функции (12) билинейны. Их частный случай - линейные функции: если индексам из P_2 в (12) соответствуют константы, то функция f - линейная.

И функции вида (12), и соответствующие им вершины будем называть *квазилинейными*.

Пусть интерпретация функциональных символов такова, что в графе вычислений присутствуют только вершины двух сортов: квазилинейные или с одной входной связью (соответствующие простым функциям одного переменного). Обозначим количества этих вершин V_q и V_1 соответственно. Оценка сложности вычисления градиента для таких графов принципиально меняется. Обратное функционирование в этом случае требует следующих затрат:

1) Поиск всех производных функций одного переменного, соответствующих вершинам графа (число таких производных равно V_1), сложность поиска одной производной оценивается как c.
2) Вычисление всех произведений (7) на ребрах - E произведений.
3) Вычисление всех сумм (6) - сложность равна E-(V- V_{out}).

Итого, суммарная сложность обратного прохождения сигналов оценивается как

$$T_{\text{grad}F} = cV_1 + 2E - (V - V_{\text{out}}).$$

Оценим сложность вычислений при прямом распространении:

1) Для любой квазилинейной вершины τ сложность вычисления функции оценивается как $|\text{in}(\tau)|$-1,
2) Для любой вершины с одной входной связью сложность оценивается как c.

Итак, для прямого распространения сложность оценивается как

$$T_F = cV_1 + (E - V_1 - V_q).$$

С ростом числа связей $\dfrac{T_{\text{grad}F}}{T_F} \to 2$!!!

Графы вычислений (с заданной интерпретацией функциональных символов), в которых присутствуют только вершины двух сортов: квазилинейные или с одной входной связью (соответствующие простым функциям одного переменного) играют особую роль. Будем называть их ***существенно квазилинейными***. Для функций, вычисляемых с помощью таких графов, затраты на вычисление вектора градиента примерно вдвое больше, чем затраты на вычисление значения функции (всего!). При этом число связей и отношение E/V могут быть сколь угодно большими. Это достоинство делает использование существенно квазилинейных графов весьма притягательным во всех задачах оптимизации. Их частным случаем являются нейронные сети, для которых роль квазилинейных вершин играют адаптивные линейные сумматоры.

Таким образом, обратное прохождение адаптивного сумматора требует таких же затрат, как и прямое. Для других элементов стандартного нейрона обратное распространение строится столь же просто: точка ветвления при обратном процессе превращается в простой сумматор, а нелинейный преобразователь $x \mapsto f(x)$ в линейную связь $\mu \mapsto f'(x)\mu$ с коэффициентом связи $f'(x)$. Поэтому для нейронных

сетей обратное распространение выглядит особенно просто. Детали можно найти в различных руководствах, например [5,6]

Отличие общего случая от более привычных нейронных сетей состоит в том, что можно использовать билинейные комбинации (12) произвольных уже вычисленных функций, а не только линейные функции от них с постоянными коэффициентами-весами.

В определенном смысле квазилинейные функции (12) вычисляются линейными сумматорами с весами 1 и z_j ($j \in P_2$) и аргументами z_i ($i \in P_1$) и z_k ($k \in P_3$), только веса не обязательно являются константами, а могут вычисляться на любом слое.

Естественно возникает вопрос о вычислительных возможностях сетей, все вершины которых являются квазилинейными. Множество функций, вычисляемых такими сетями, содержит все координатные функции и замкнуто относительно сложения, умножения на константу и умножения функций. Следовательно оно содержит и все многочлены от любого количества переменных. Любая непрерывная функция на замкнутом ограниченном множестве в конечномерном пространстве может быть сколь угодно точно приближена с их помощью.

6. Двойственность по Лежандру и смысл двойственных переменных

Просматривая текст предыдущих разделов, я убедился, что подробное изложение и элементы стиля математической логики могут даже очевидное сделать сложным. В данном разделе описывается связь двойственного функционирования сетей автоматов с преобразованием Лежандра и неопределенными множителями Лагранжа. Это изложение ориентировано на читателя, знакомившегося с лагранжевым и гамильтоновым формализмами и их связью (в классической механике или вариационном исчислении). Фактически на двух страницах будет заново изложен весь предыдущий материал главы.

Переменные обратного функционирования μ появляются как вспомогательные при вычислении производных сложной функции. Переменные такого типа появляются не случайно. Они постоянно возникают в задачах оптимизации и являются множителями Лагранжа.

Мы вводим μ, исходя из правил дифференцирования сложной функции. Возможен другой путь, связанный с переходом от функции Лагранжа к функции

Гамильтона. Изложим его и параллельно получим ряд дальнейших обобщений. Для всех сетей автоматов, встречавшихся нам в предыдущих разделах и главах, можно выделить три группы переменных:

внешние входные сигналы $x_{...}$,

переменные функционирования - значения на выходах всех элементов сети $f_{...}$,

переменные обучения $a_{...}$

(многоточиями заменяются различные наборы индексов).

Объединим их в две группы - вычисляемые величины $y_{...}$ - значения $f_{...}$ и задаваемые - $b_{...}$ (включая $a_{...}$ и $x_{...}$). Упростим индексацию, перенумеровав f и b натуральными числами: $f_1,...,f_N$; $b_1,...,b_M$.

Пусть функционирование системы задается набором из N уравнений

$$\psi_i(y_1,...,y_N,b_1,...,b_M)=0 \ (i=1,...,N). \qquad (13)$$

Для послойного вычисления сложных функций вычисляемые переменные - это значения вершин для всех слоев, кроме нулевого, задаваемые переменные - это значения вершин первого слоя (константы и значения переменных), а уравнения функционирования имеют простейший вид (4), для которого

$$\psi_i = Z(\tau) - f(z_1,...,z_k),$$
$$\text{где } z_i = Z(\tau') \text{ при } i \in P(\tau',\tau).$$

Предполагается, что система уравнений (13) задает способ вычисления y_i .

Пусть имеется функция (лагранжиан) $H(y_1,...,y_N,b_1,...,b_M)$. Эта функция зависит от b и явно, и неявно - через переменные функционирования y. Если представить, что уравнения (13) разрешены относительно всех y ($y=y(b)$), то H можно представить как функцию от b:

$$H=H_1(b)=H(y_1(b),...,y_N(b),b). \qquad (14)$$

где b - вектор с компонентами b_i .

Для задачи обучения требуется найти производные $D_i=\partial H_1(b)/\partial b_i$. Непосредственно и явно это сделать трудно.

Поступим по-другому. Введем новые переменные $\mu_1,...,\mu_N$ (множители Лагранжа) и производящую функцию W:

$$W(y,b,\mu) = H(y,b) + \sum_{i=1}^{N\mu} \mu_i \psi_i(y,b).$$

В функции W аргументы y, b и μ - независимые переменные.

Уравнения (13) можно записать как

$$\frac{\partial W}{\mu_i} = 0,\ (i = 1,\ldots,N). \tag{15}$$

Заметим, что для тех y, b, которые удовлетворяют уравнениям (13), при любых μ

$$W(y,b,\mu) \equiv H(y,b). \tag{16}$$

Это означает, что для истинных значений переменных функционирования y при данных b функция $W(y,b,\mu)$ совпадает с исследуемой функцией H.

Попытаемся подобрать такую зависимость $\mu_i(b)$, чтобы, используя (16), получить для $D_i = \partial H_1(b)/\partial b_i$ наиболее простые выражения . На многообразии решений (15)

$$D_i = \frac{\partial H(y(b),b)}{\partial b_i} = \frac{\partial W(y(b),b,\mu(b))}{\partial b_i}.$$

Поэтому

$$\begin{aligned} D_i &= \frac{\partial H(y,b)}{\partial b_i} + \sum_{j=1}^{N} \frac{\partial H(y,b)}{\partial y_j} \times \frac{\partial y_j(b)}{\partial b_i} = \\ &= \frac{\partial W(y,b,\mu)}{\partial b_i} + \sum_{j=1}^{N} \frac{\partial W(y,b,\mu)}{\partial y_j} \times \frac{\partial y_j(b)}{\partial b_i} + \sum_{j=1}^{N} \frac{\partial W(y,b,\mu)}{\partial \mu_j} \times \frac{\partial \mu_j(b)}{\partial b_i} = \\ &= \sum_{j=1}^{N} \left(\frac{\partial H(y,b)}{\partial y_j} + \sum_{k=1}^{N} \mu_k \frac{\partial \psi_k(y,b)}{\partial y_j} \right) \times \frac{\partial y_j(b)}{\partial b_i} + \sum_{j=1}^{N} \mu_j \frac{\partial \psi_j(y,b)}{\partial b_i} + \frac{\partial H(y,b)}{\partial b_i}\ . \end{aligned} \tag{17}$$

Мы всюду различаем функцию $H(y,b)$, где y и b - независимые переменные, и функцию только от переменных b $H(y(b),b)$, где $y(b)$ определены из уравнений (13). Аналогичное различение принимается для функций $W(y,b,\mu)$ и $W(y(b),b,\mu(b))$.

Произвол в определении $\mu(b)$ надо использовать наилучшим образом - все равно от него придется избавляться, доопределяя зависимости. Если выбрать такие μ, что слагаемые в первой сумме последней строки выражения (17) обратятся в нуль, то формула для D_i резко упростится. Положим поэтому

$$\frac{\partial H(y,b)}{\partial y_j} + \sum_{k=1}^{N} \mu_k \frac{\partial \psi_k(y,b)}{\partial y_j} = 0\,. \tag{18}$$

Это - система уравнений для определения μ_k (k=1,...,N). Если μ определены согласно (18), то

$$D_i = \frac{\partial H(y,b)}{\partial b_i} + \sum_{j=1}^{N} \mu_j \frac{\partial \psi_j(y,b)}{\partial b_i}$$

это - в точности те выражения, которые использовались при поиске производных сложных функций. В наших вычислениях мы пользовались явным описанием функционирования. Метод множителей Лагранжа допускает и менее явные описания сетей.

В математическом фольклоре бытует такой тезис: сложная задача оптимизации может быть решена только при эффективном использовании двойственности. Методы быстрого дифференцирования являются яркой иллюстрацией этого тезиса.

7. Оптимизационное обучение нейронных сетей

Когда можно представить обучение нейронных сетей как задачу оптимизации? В тех случаях, когда удается *оценить* работу сети. Это означает, что можно указать, хорошо или плохо сеть решает поставленные ей задачи и оценить это «хорошо или плохо» количественно. Строится *функция оценки*. Она, как правило, явно зависит от выходных сигналов сети и неявно (через функционирование) - от всех ее параметров. Простейший и самый распространенный пример оценки - сумма квадратов расстояний от выходных сигналов сети до их требуемых значений:

$$H = \frac{1}{2} \sum_{\tau \in \mathrm{v}_{\mathrm{out}}} (Z(\tau) - Z^*(\tau))^2 ,$$

где $Z^*(\tau)$ - требуемое значение выходного сигнала.

Другой пример оценки- качество классификации в сетях Кохонена. В этом случае ответы заранее неизвестны, но качество работы сети оценить можно.

Устройство, вычисляющее оценку, надстраивается над нейронной сетью и градиент оценки может быть вычислен с использованием описанного принципа двойственности.

В тех случаях, когда оценка является суммой квадратов ошибок, значения независимых переменных двойственного функционирования $\mu(\tau)$ для вершин выходного слоя $\mathrm{v}_{\mathrm{out}}$ при вычислении градиента H устанавливаются равными

$$\mu(\tau) = \frac{\partial H}{\partial Z(\tau)} = (Z(\tau) - Z^*(\tau)) \text{ -} \qquad (19)$$

на вход при обратном функционировании поступают ошибки выходных сигналов! Это обстоятельство настолько впечатлило исследователей, что они назвали метод

вычисления градиента оценки методом обратного распространения ошибок. Впрочем, после формулы Уидроу, описанной в главе 2, формула (19) должна быть очевидной.

Для обучения используется оценка, усредненная по примерам с известным ответом.

Предлагается рассматривать обучение нейронных сетей как задачу оптимизации. Это означает, что весь мощный арсенал методов оптимизации может быть испытан для обучения. Так и видится: нейрокомпьютеры наискорейшего спуска, нейрокомпьютеры Ньютона, Флетчера и т.п. - по названию метода нелинейной оптимизации.

Существует, однако, ряд специфических ограничений. Они связаны с огромной размерностью задачи обучения. Число параметров может достигать 10^8 - и даже более. Уже в простейших программных имитаторах на персональных компьютерах подбирается 10^3 - 10^4 параметров.

Из-за высокой размерности возникает два требования к алгоритму:

1. Ограничение по памяти. Пусть n - число параметров. Если алгоритм требует затрат памяти порядка n^2 ,то он вряд ли применим для обучения. Вообще говоря, желательно иметь алгоритмы, которые требуют затрат памяти порядка Kn, K=const.

2. Возможность параллельного выполнения наиболее трудоемких этапов алгоритма и желательно - нейронной сетью. Если какой-либо особо привлекательный алгоритм требует память порядка n^2 , то его все же можно использовать, если с помощью анализа чувствительности и, возможно, контрастирования сократить число обучаемых параметров до разумных пределов.

Еще два обстоятельства связаны с нейрокомпьютерной спецификой.

1. Обученный нейрокомпьютер должен с приемлемой точностью решать все тестовые задачи (или, быть может, почти все с очень малой частью исключений). Поэтому задача обучения становится по существу многокритериальной задачей оптимизации: надо найти точку общего минимума большого числа функций. Обучение нейрокомпьютера исходит из гипотезы о существовании такой точки. Основания гипотезы - очень большое число переменных и сходство между функциями. Само понятие "сходство" здесь трудно формализовать, но опыт показывает что предположение о существовании общего минимума или, точнее, точек, где значения всех оценок мало отличаются от минимальных, часто оправдывается.

2. Обученный нейрокомпьютер должен иметь возможность приобретать новые навыки без утраты старых. Возможно более слабое требование: новые навыки могут сопровождаться потерей точности в старых, но эта потеря не должна быть особо существенной, а качественные изменения должны быть исключены. Это означает, что в достаточно большой окрестности найденной точки общего минимума оценок значения этих функций незначительно отличаются от минимальных. Мало того, что должна быть найдена точка общего минимума, так она еще должна лежать в достаточно широкой низменности, где значения всех минимизируемых функций близки к минимуму. Для решения этой задачи нужны специальные средства.

Итак, имеем четыре специфических ограничения, выделяющих обучение нейрокомпьютера из общих задач оптимизации: астрономическое число параметров, необходимость высокого параллелизма при обучении, многокритериальность решаемых задач, необходимость найти достаточно широкую область, в которой значения всех минимизируемых функций близки к минимальным. В остальном - это просто задача оптимизации и многие классические и современные методы достаточно естественно ложатся на структуру нейронной сети.

Заметим, кстати, что если вести оптимизацию (минимизацию ошибки), меняя параметры сети, то в результате получим решение задачи аппроксимации. Если же ведется минимизация целевой некоторой функции и ищутся соответствующие значения переменных, то в результате решаем задачу оптимизации (хотя формально это одна и та же математическая задача и разделение на параметры и переменные определяется логикой предметной области, а с формальной точки зрения разница практически отсутствует).

Значительное число публикаций по методам обучения нейронных сетей посвящено переносу классических алгоритмов оптимизации (см., например, [7,8]) на нейронные сети или поиску новых редакций этих методов, более соответствующих описанным ограничениям - таких, например, как метод виртуальных частиц [5,6]. Существуют обширные обзоры и курсы, посвященные обучению нейронных сетей (например, [9,10]). Не будем здесь останавливаться на обзоре этих работ - если найден градиент, то остальное приложится.

Работа над главой была поддержана Красноярским краевым фондом науки, грант 6*F*0124.

Литература

1.Rumelhart D.E., Hinton G.E., Williams R.J. Learning internal representations by error propagation. // Parallel Distributed Processing: Exploration in the Microstructure of Cognition, D.E.Rumelhart and J.L.McClelland (Eds.), vol. 1, Cambridge, MA: MIT Press, 1986. PP. 318 - 362.

2.Rummelhart D.E., Hinton G.E., Williams R.J. Learning representations by back-propagating errors // Nature, 1986. V. 323. P. 533-536.

3.Áàðöåâ Ñ.È., Îõîíèí Â.À. Àäàïòèâíûå ñåòè îáðàáîòêè èíôîðìàöèè. Ïðåïðèíò ÈÔ ÑÎ ÀÍ ÑÑÑÐ, Êðàñíîÿðñê, 1986, ¹59Á, 20 c.

4.Шенфилд Дж. Математическая логика. М.: Наука, 1975. 528 с.

5.Горбань А.Н. Обучение нейронных сетей. М.": изд. СССР-США СП "ПараГраф", 1990. 160 с. (English Translation: AMSE Transaction, Scientific Siberian, A, 1993, Vol. 6. Neurocomputing, PP. 1-134).

6.Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. Новосибирск: Наука (Сиб. отделение), 1996. 276 с.

7.Химмельблау Д. Прикладное нелинейное программирование. М.: Мир, 1975. 534 с.

8.Гилл Ф., Мюррей У., Райт М. Практическая оптимизация. М.: Мир,1985. 509 с.

9.Zurada J. M. Introduction to artificial neural systems. PWS Publishing Company, 1992. 785 pp.

10.Haykin S. Neural networks. A comprehensive foundations. McMillan College Publ. Co. N.Y., 1994. 696 pp.

Глава 4.

Нейросетевые информационные модели сложных инженерных систем

С.А. Терехов

Лаборатория Искусственных Нейронных Сетей, Российский Федеральный Ядерный Центр - Всероссийский НИИ Технической Физики

В данной главе обсуждаются нейросетевые методы построения моделей сложных систем, основанные на экспериментальных данных. Подробно рассмотрены постановки типовых задач информационного моделирования (прямых, обратных и смешанных). Изложение сопровождается модельными иллюстрациями и примерами реальных практических применений.

Сложные системы

Рассмотрим систему, состоящую из некоторого числа компонент. Для определенности будем иметь в виду, скажем, терминал крупного океанского порта, обслуживающий разгрузку судов портовыми кранами, и отправку грузов автомобильным и железнодорожным транспортом. Нашей конечной целью будет построение *модели* системы, описывающей ее поведение, и обладающей предсказательными свойствами. Модель способна во многих приложениях заменить собой исследуемую систему.

Адрес для электронной переписки: S.A.Terekhoff@vniitf.ru; Факс 7-(35172)-329-19;
Интернет:WWW:http://www.ch70.chel.su

Каждая из компонент системы имеет свои свойства и характер поведения в зависимости от собственного состояния и внешних условий. Если все возможные проявления системы сводятся к сумме проявлений ее компонент, то такая система является простой, несмотря на то, что число ее компонент может быть велико. Для описания простых систем традиционно применяются методы *анализа*, состоящие в последовательном расчленении системы на компоненты и построении моделей все более простых элементов. Таковым в своей основе является метод математического моделирования[1], в котором модели описываются в форме уравнений, а предсказание поведения системы основывается на их решении.

Современные технические системы (например, упомянутый выше порт, инженерные сооружения, приборные комплексы, транспортные средства и др.) приближаются к такому уровню сложности, когда их наблюдаемое поведение и свойства не сводятся к простой сумме свойств отдельных компонент. При объединении компонент в систему возникают *качественно* новые свойства, которые не могут быть установлены посредством *анализа* свойств компонент.

В случае терминала порта небольшие отклонения в производительности работы кранов, малые изменения или сбои графика движения железнодорожных составов, отклонения в степени загрузки и в графике прибытия судов могут вызвать качественно новый режим поведения порта, как системы, а именно затор. Образование затора вызывает обратное воздействие на режимы работы компонент, что может привести к серьезным авариям и т.д. Состояние затора не может быть в полной мере получено на основе отдельного анализа, например, свойств одного крана. Однако в рамках системы обычный режим работы этого крана может приводить к состоянию затора.

Такие системы, в которых при вычленении компонент могут быть потеряны принципиальные свойства, а при добавлении компонент возникают качественно новые свойства, будем называть *сложными*. Модель сложной системы, основанная на принципах анализа, будет неустранимо неадекватной изучаемой

системе, поскольку при разбиении системы на составляющие ее компоненты теряются ее качественные особенности.

Принципы информационного кибернетического моделирования

Возможным выходом из положения является построение модели на основе *синтеза* компонент. Синтетические модели являются практически единственной альтернативой в социологии, долгосрочных прогнозах погоды, в макроэкономике, медицине. В последнее время синтетические информационные модели широко используются и при изучении технических и инженерных систем. В ряде приложений информационные и математические компоненты могут составлять единую модель (например, внешние условия описываются решениями уравнений математической физики, а отклик системы - информационной моделью).

Основным принципом информационного моделирования является принцип "*черного ящика*". В противоположность аналитическому подходу, при котором моделируется внутренняя *структура* системы, в синтетическом методе "черного ящика" моделируется внешнее *функционирование* системы. С точки зрения пользователя модели структура системы спрятана в черном ящике, который имитирует поведенческие особенности системы.

Кибернетический принцип "черного ящика" был предложен [2] в рамках теории идентификации систем, в которой для построения модели системы предлагается широкий параметрический класс базисных функций или уравнений, а сама модель *синтезируется* путем выбора параметров из условия наилучшего, при заданной функции ценности, соответствия решений уравнений поведению системы. При этом структура системы никак не отражается в структуре уравнений модели.

Функционирование системы в рамках синтетической модели описывается чисто *информационно*, на основе данных экспериментов или наблюдений над реальной системой. Как правило, информационные модели проигрывают

формальным математическим моделям и экспертным системам[1] по степени "объяснимости" выдаваемых результатов, однако отсутствие ограничений на сложность моделируемых систем определяет их важную практическую значимость.

Типы информационных моделей

Можно выделить несколько типов[2] информационных моделей, отличающихся по характеру запросов к ним. Перечислим лишь некоторые из них:

- Моделирование отклика системы на внешнее воздействие
- Классификация внутренних состояний системы
- Прогноз динамики изменения системы
- Оценка полноты описания системы и сравнительная информационная значимость параметров системы
- Оптимизация параметров системы по отношению к заданной функции ценности
- Адаптивное управление системой

В этом разделе изложение будет основываться на моделях первого из указанных типов.

Пусть X - вектор, компоненты которого соответствуют количественным свойствам системы, X' - вектор количественных свойств внешних воздействий. Отклик системы может быть описан некоторой (неизвестной) вектор-функцией F: $Y = F(X,X')$, где Y - вектор отклика. Задачей моделирования является *идентификация системы*, состоящая в нахождении функционального отношения, алгоритма или системы правил в общей форме $Z=G(X,X')$, ассоциирующей каждую пару векторов (X,X') с вектором Z таким образом, что Z

[1] Информационные модели, основанные на *логически прозрачных* нейронных сетях, предложенные в [4], в некоторой степени отражают причинно-следственные взаимоотношения между параметрами модели

и Y близки в некоторой метрике, отражающей цели моделирования. Отношение $Z=G(X,X')$, воспроизводящее в указанном смысле функционирование системы F, будем называть *информационной моделью* системы F.

Нейронные сети в информационном моделировании

Искусственные нейронные сети (ИНС) являются удобным и естественным базисом для представления информационных моделей. Нейросеть может быть достаточно формально определена [3], как совокупность простых процессорных элементов (часто называемых *нейронами*), обладающих полностью локальным функционированием, и объединенных однонаправленными связями (называемыми *синапсами*). Сеть принимает некоторый *входной* сигнал[3] из внешнего мира, и пропускает его сквозь себя с преобразованиями в каждом процессорном элементе. Таким образом, в процессе прохождения сигнала по связям сети происходит его обработка, результатом которой является определенный *выходной* сигнал. В укрупненном виде ИНС выполняет функциональное соответствие между входом и выходом, и может служить информационной моделью G системы F.

Определяемая нейросетью функция может быть *произвольной* при легко выполнимых требованиях к структурной сложности сети и наличии нелинейности в переходных функциях нейронов [4]. Возможность представления любой системной функции F с наперед заданной точностью определяет нейросеть, как компьютер *общего назначения*. Этот компьютер, в сравнении с машиной фон Неймана, имеет принципиально другой способ организации вычислительного процесса - он *не программируется* с использованием явных правил и кодов в соответствии с заданным алгоритмом, *а обучается* посредством целевой адаптации синаптических связей (и, реже, их

---

[2] Границы между типами моделей являются весьма условными

[3] Под сигналом в широком смысле может понимается вектор состояний входов нейросети.

структурной модификацией и изменением переходных функций нейронов) для представления требуемой функции.

В гипотетической ситуации, когда функция системы *F* известна или известен алгоритм ее вычисления при произвольных значениях аргументов, машина фон Неймана наилучшим средством для моделирования (состоящего в вычислении *F*), и необходимость в информационных моделях отпадает.

При моделировании реальных сложных технических систем значения системной функции *F* получаются на основе экспериментов или наблюдений, которые проводятся лишь для конечного параметров *X*. При этом значения как *Y* так и *X* измеряются приближенно, и подвержены ошибкам различной природы (см. ниже). Целью моделирования является получение значений системных откликов при произвольном изменении *X*. В этой ситуации может быть успешно применена информационная (статистическая) модель *G* исследуемой системы *F*.

Информационные модели могут строиться на основе традиционных методов *непараметрической статистики* Данная наука позволяет строить обоснованные модели систем в случае большого набора экспериментальных данных (достаточного для доказательства статистических гипотез о характере распределения) и при относительно равномерном их распределении в пространстве параметров. Однако при высокой стоимости экспериментальных данных, или невозможности получения достаточного их количества (как, например, при построении моделей тяжелых производственных аварий, пожаров и т.п.), их высокой зашумленности, неполноте и противоречивости, нейронные модели оказываются более предпочтительными. Нейронная сеть оказывается избирательно чувствительной в областях скопления данных, и дает гладкую интерполяцию в остальных областях.

Эта особенность нейросетевых моделей основывается на более общем принципе - адаптивной кластеризации данных. Одной из первых сетей, обладающих свойствами адаптивной кластеризации была карта самоорганизации Т. Кохонена [5,6]. Задачей нейросети Кохонена является автоматизированное построение отображения набора входных векторов высокой размерности в карту

кластеров меньшей размерности, причем, таким образом что близким кластерам на карте отвечают близкие друг к другу входные вектора в исходном пространстве. Таким образом, при значительном уменьшении размерности пространства сохраняется топологический порядок расположения данных. При замене всех векторов каждого кластера его центроидом достигается высокая степень сжатия информации при сохранении ее структуры в целом[4].

Карты Кохонена применяются в основном, для двух целей. Первая из них - наглядное упорядочивание многопараметрической информации. На практике обычно используются одномерные и двумерные карты. Кластеры, задаваемые узлами карты, содержат группы в некотором смысле похожих наблюдений, которым может быть приписан групповой семантический смысл. Одним из новых эффективных применений сети Кохонена является построение тематической карты электронных сообщений в глобальных компьютерных сетях. При помощи такой карты пользователь получает возможность свободной навигации в бесконечном потоке сообщений, в соответствии с индивидуальным кругом интересов[5]. В применении к моделированию технических систем, карты Кохонена могут использоваться для выявления различий в режимах поведения системы, при этом могут выявляться аномальные режимы. Важно, что при этом могут быть обнаружены неожиданные скопления близких данных, последующая интерпретация которых пользователем может привести к получению *нового знания* об исследуемой системе.

Вторая группа технических применений связана с предобработкой данных. Карта Кохонена группирует близкие входные сигналы X, а требуемая функция $Y=G(X)$ строится на основе применения обычной нейросети прямого распространения (например, многослойного персептрона или линейной звезды Гроссберга) к выходам нейронов Кохонена. Такая гибридная архитектура была

---

[4] При этом, конечно, часть информации теряется. Однако, это не очень принципиально, например, при сжатии изображений.

[5] Электронный адрес интерактивной карты Кохонена *WEBSOM* с сообщениями по тематике нейронных сетей, разработанной в университете в Хельсинки группой самого Т.Кохонена, в сети *Интернет*: http://websom.hut.fi

предложена Р. Хехт-Нильсеном [7,8], она получила название сети *встречного распространения*. Нейроны слоя Кохонена обучаются без учителя, на основе самоорганизации, а нейроны распознающих слоев адаптируются с учителем итерационными методами. При использовании линейных выходных нейронов значения их весов могут быть получены безитерационно, непосредственным вычислением псевдо-обратной матрицы по Муру-Пенроузу.

Сеть встречного распространения дает кусочно-постоянное представление модели $Y=G(X)$, поскольку при вариации вектора X в пределах одного кластера на слое соревнующихся нейронов Кохонена возбуждается один и тот же нейрон-победитель. В случае сильно зашумленных данных, такое представление обладает хорошими регуляризирующими свойствами. При этом процедура обучения сети встречного распространения заметно быстрее, чем, например, обучение многослойного персептрона стандартным методом обратного распространения ошибок [9].

Другой альтернативой традиционным многослойным моделям является переход к нейросетям простой структуры, но с усложненными процессорными элементами. В частности, можно рассмотреть нейроны высоких порядков, активирующим сигналом для которых является взвешенная сумма входов, их попарных произведений, произведений троек и т.д., вплоть до порядка k.

$$y = f\left(W^{(0)} + \sum_i W_i^{(1)} x_i + \sum_{i<j} W_{ij}^{(2)} + \ldots + \sum_{i1<i2<..<ik} W_{i1..ik}^{(k)} x_1 x_2 .. x_k \right)$$

Каждый процессорный элемент k-го порядка способен выполнить не только линейное разделение областей в пространстве входов, но также и произвольное разделение, задаваемое поли-линейной функцией нескольких аргументов. Семейство решающих правил, определяемых нелинейным нейроном значительно богаче, чем множество линейно разделимых функций. На Рис. 1 приведен пример решающего правила, задаваемого *одним* нейроном второго порядка, для классической линейно неразделимой задачи "*исключающее ИЛИ*".

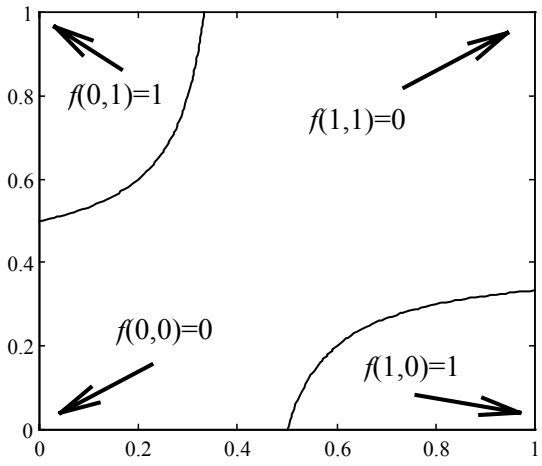


Рис.1 Решающее правило для задачи "исключающее ИЛИ".

Важным достоинством нейронов высокого порядка является возможность строить нейросетевые модели без скрытых слоев, воспроизводящие широкий класс функций[6]. Такие нейроархитектуры не требуют длительного итерационного обучения, оптимальные веса даются решением уравнений регрессии. Другой отличительной чертой является возможность эффективной аппаратной (электронной или оптической) реализации корреляций высокого порядка. Так, например, существуют нелинейные среды, оптические свойства которых определяются полиномиальной зависимостью от амплитуды электрического поля световой волны. Потенциально, устройства, основанные на таких средах, могут обеспечить высокие скорости вычислений со свойственной оптическим компьютерам супер-параллельностью вычислений.

В этой главе описанные и другие нейроархитектуры будут применены к модельным и реалистичным задачам информационного моделирования сложных инженерных систем.

---

[6] Нужно заметить, что класс нейросетей без скрытых слоев не является *полным* в смысле возможности приближения произвольной функции. Так, для представления решающих правил для двух переменных все многообразие функций сводится лишь к Паде (1,1)-приближениям (гиперболам).

Характер приближений в информационных моделях

Специфичность информационных моделей проявляется не только в способах их синтеза, но и характере делаемых приближений (и связанных с ними ошибок). Отличия в поведении системы и ее информационной модели возникают вследствие свойств экспериментальных данных.

- Информационные модели *ab initio* являются неполными. Пространства входных и выходных переменных *не могут*, в общем случае, содержать *все* параметры, существенные для описания поведения системы. Это связано как с техническими ограничениями, так и с ограниченностью наших представлений о моделируемой системе. Кроме того, при увеличении числа переменных ужесточаются требования на объем необходимых экспериментальных данных для построения модели (об этом см. ниже). Эффект опущенных (скрытых) входных параметров может нарушать однозначность моделируемой системной функции F.
- База экспериментальных данных, на которых основывается модель G рассматривается, как внешняя данность. При этом, в данных всегда присутствуют ошибки разной природы, шум, а также противоречия отдельных измерений друг другу. За исключением простых случаев, искажения в данных не могут быть устранены полностью.
- Экспериментальные данные, как правило, имеют произвольное распределение в пространстве переменных задачи. Как следствие, получаемые модели будут обладать неодинаковой достоверностью и точностью в различных областях изменения параметров.
- Экспериментальные данные могут содержать пропущенные значения (например, вследствие потери информации, отказа измеряющих датчиков, невозможности проведения полного набора анализов и т.п.). Произвольность в интерпретации этих значений, опять-таки, ухудшает свойства модели.

Такие особенности в данных и в постановке задач требуют особого отношения к ошибкам информационных моделей.

Ошибка обучения и ошибка обобщения

Итак, при информационном подходе требуемая модель G системы F не может быть полностью основана на явных правилах и формальных законах. Процесс получения G из имеющихся отрывочных экспериментальных сведений о системе F может рассматриваться, как *обучение* модели G поведению F в соответствии с заданным критерием, настолько близко, насколько возможно. Алгоритмически, обучение означает подстройку внутренних параметров модели (весов синаптических связей в случае нейронной сети) с целью минимизации *ошибки модели* $E = \|G - F\|$.

Прямое измерение указанной ошибки модели на практике не достижимо, поскольку системная функция F при произвольных значениях аргумента не известна. Однако возможно получение ее оценки:

$$E_L = \sum_{X \in \mathbf{X}} \|G(X) - Y\|,$$

где суммирование по X проводится по некоторому конечному набору параметров $\boldsymbol{X}$, называемому *обучающим множеством*. При использовании базы данных наблюдений за системой, для обучения может отводиться некоторая ее часть, называемая в этом случае *обучающей выборкой*. Для обучающих примеров X отклики системы Y известны[7]. Норма невязки модельной функции G и системной функции Y на множестве $\boldsymbol{X}$ играет важную роль в информационном моделировании и называется *ошибкой обучения* модели.

Для случая точных измерений (например, в некоторых задачах классификации, когда отношение образца к классу не вызывает сомнений) однозначность системной функции для достаточно широкого класса G моделей гарантирует возможность достижения произвольно малого значения ошибки

[7] С учетом описанных выше особенностей экспериментальных данных.

обучения E_L. Нарушение однозначности системной функции в присутствии экспериментальных ошибок и неполноты признаковых пространств приводит в общем случае к ненулевым ошибкам обучения. В этом случае предельная достижимая ошибка обучения может служить мерой корректности постановки задачи и качества класса моделей G.

В приложениях пользователя обычно интересуют предсказательные свойства модели. При этом главным является вопрос, каковым будет отклик системы на *новое воздействие*, пример которого отсутствует в базе данных наблюдений. Наиболее общий ответ на этот вопрос дает (по-прежнему недоступная) ошибка модели E. Неизвестная ошибка, допускаемая моделью G на данных, не использовавшихся при обучении, называется *ошибкой обобщения* модели E_G.

Основной целью при построении информационной модели является уменьшение именно ошибки обобщения, поскольку малая ошибка обучения гарантирует адекватность модели лишь в заранее выбранных точках (а в них значения отклика системы известны и без всякой модели!). Проводя аналогии с обучением в биологии, можно сказать, что малая ошибка обучения соответствует *прямому запоминанию* обучающей информации, а малая ошибка обобщения - *формированию понятий и навыков*, позволяющих распространить ограниченный опыт обучения на новые условия. Последнее значительно более ценно при проектировании нейросетевых систем, так как для непосредственного запоминания информации лучше приспособлены не нейронные устройства компьютерной памяти.

Важно отметить, что малость ошибки обучения *не* гарантирует малость ошибки обобщения. Классическим примером является построение модели функции (аппроксимация функции) по нескольким заданным точкам полиномом высокого порядка. Значения полинома (модели) при достаточно высокой его степени являются *точными* в обучающих точках, т.е. ошибка обучения равна нулю. Однако значения в промежуточных точках могут значительно отличаться от аппроксимируемой функции, следовательно ошибка обобщения такой модели может быть неприемлемо большой.

Поскольку истинное значение ошибки обобщения не доступно, в практике используется ее оценка. Для ее получения анализируется часть примеров из имеющейся базы данных, для которых известны отклики системы, но которые *не использовались* при обучении. Эта выборка примеров называется *тестовой выборкой*. Ошибка обобщения оценивается, как норма уклонения модели на множестве примеров из тестовой выборки.

Оценка ошибки обобщения является принципиальным моментом при построении информационной модели. На первый взгляд может показаться, что сознательное не использование части примеров при обучении может только ухудшить итоговую модель. Однако без этапа тестирования единственной оценкой качества модели будет лишь ошибка обучения, которая, как уже отмечалось, мало связана с предсказательными способностями модели. В профессиональных исследованиях могут использоваться несколько независимых тестовых выборок, этапы обучения и тестирования повторяются многократно с вариацией начального распределения весов нейросети, ее топологии и параметров обучения. Окончательный выбор "наилучшей" нейросети выполняется с учетом имеющегося объема и качества данных, специфики задачи, с целью минимизации риска большой ошибки обобщения при эксплуатации модели.

Прямые, обратные и комбинированные задачи информационного моделирования

При формулировании постановки информационной задачи предсказания реакции исследуемой системы при ее известном состоянии на заданные внешние воздействия, т.е. получения величин Y при заданных X исследователь имеет дело с *прямой* задачей. Прямая задача является типичной при моделировании поведения системы, если запросы к информационной модели носят характер *что-если*.

Другим важным классом информационных задач являются *обратные* задачи. Целью обратной задачи выступает получение входных величин X, соответствующих наблюдаемым значениям выходов Y. При моделировании сложных систем соответствующий запрос к модели формулируется, как поиск внешних условий, которые привели к реализовавшемуся отклику системы.

Для большинства приложений чисто обратные задачи встречаются относительно редко, так как обычно имеются дополнительные сведения о системе. Например, кроме измеренного отклика, могут быть известны переменные состояния системы и часть параметров воздействия. В этом случае задача относится к классу *комбинированных* задач: по известным значениям части компонент входного X и выходного Y векторов восстановить оставшиеся неизвестные компоненты.

В общем случае моделируемая системная функция может быть представлена в виде $(X,Y)=\boldsymbol{F}(X, Y)$. В этом случае комбинированный вектор (X,Y) рассматривается одновременно, как входной и выходной. В этом смысле, произвольная задача допускает комбинированную постановку.

Некорректность обратной задачи

Отличительная особенность обратных и комбинированных задач состоит в том, что они обычно являются *некорректно поставленными*[10], и поэтому требуют специализированных методов поиска приближенных решений. Согласно Ж.Адамару, для корректности постановки задачи необходимо:

- *существование* решения при всех допустимых исходных данных;
- *единственность* данного решения;
- *устойчивость* решения к изменениям (малым) исходных данных.

Рассмотрим характер возможных нарушений данных условий при решении модельной обратной задачи.

Пусть имеется три исследуемых систем, описываемых кусочно-линейными функциями одной переменной $y=F(x)$ на отрезке [0..1]. Системы отличаются друг от друга величиной скачка h системной функции (см Рис.2). Прямая задача состоит в построении приближения G к функции F, с использованием пар значений $\{x_i, y_i=s(x_i)\}$, где x_i - конечный набор N_α случайных равномерно распределенных на [0..1] точек. Обратная задача заключается в нахождении функции, аппроксимирующей соотношения $x_i(y_i)$. В зависимости от величины скачка моделируемой функции можно выделить три варианта.

Система А (h=0). Модель является линейной: $y=x$. Для прямой задачи легко получить исчезающую ошибку обучения $E_L \approx 0$, и малую[8] ошибку обобщения E_G. Для обратной задачи получаются такие же результаты, так она при *точных* значения $\{x_i, y_i\}$ не содержит некорректности. Задачи с решениями, корректными на всей области определения и множестве значений, будем называть *безусловно корректными*. Корректность постановки обратной задачи для системы А определяется существованием однозначной и непрерывной функции F^{-1}.

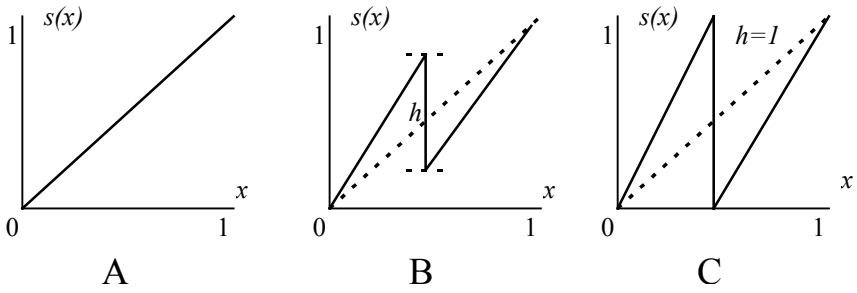


Рис 2. Модельные системы с различными величинами скачка системной функции.

Система В ($0<h<1$). Прямая задача в этом случае также хорошо определена, и при использовании достаточно богатого множества базисных функций можно произвольно уменьшить ошибку обучения ($E_L \approx 0$) при хорошем обобщении. Обратная задача характеризуется наличием на множестве значений областей с однозначной ($y > 0.5+0.5h$; $y < 0.5-0.5h$) и неоднозначной ($y \in [0.5-0.5h,$

[8] Это достигается, например, использованием нейронной сети без скрытых слоев с произвольной переходной функцией нейронов, имеющей близкий к линейному

0.5+0.5h]) обратной функцией. В областях однозначности функции могут быть получены произвольно точные результаты для обратной задачи. Однако в отрезке нарушения однозначности ошибка обучения (и ошибка обобщения) останется конечной, поскольку противоречие в данных, полученных из разных ветвей обратной функции, не устранимо. Значение ошибки обобщения пропорционально длине отрезка неоднозначности h. Такие задачи, корректное (единственное и устойчивое) решение которых может быть получено только для некоторой подобласти множества значений, будем называть *условно* (или *частично*) *корректными*[9].

Система С (h=1). Прямая задача по-прежнему корректно поставлена, требуемое обучение и обобщение может быть достигнуто ($E_L \approx 0$). Однако ситуация качественно меняется для случая обратной задачи. Обратная функция двузначна на всем множестве значений, информация о ее значении минимальна. Обратная задача полностью некорректно поставлена.

Что общего между всеми этими примерами? В каждом из них ошибка обобщения при решении обратной задачи не может быть меньше значения, определяемого размером области неоднозначности h, который, таким образом, может рассматриваться, как мера некорректности задачи. В случае, если для решения обратной задачи используется метод со стабилизирующими свойствами (например, с малым числом свободных параметров по сравнению с числом обучающих примеров), будет получено гладкое решение с ненулевой ошибкой обучения, определяемой параметром h.

Заметим, что прямая задача является безусловно корректной только при полном отсутствии шума в обучающих данных. При наличии случайных компонент в значениях X имеется целое "облако" решений прямой задачи,

---

участок изменения.

[9] В книге [1], стр. 563, условно-корректными названы задачи, в постановку которых добавлено априорное предположение о существовании решения на некотором компакте. Для данного решения должна быть доказана теорема единственнности. В нашем рассмотрении в качестве такого компактного множества выступают отрезки, на которых обратная функция однозначна.

причем размер облака пропорционален величине шума. Таким образом, нарушается единственность решения прямой задачи, и она становится некорректно поставленной.

Регуляризация в нейросетевых моделях

Классическим методом решения некорректных задач является метод регуляризации А.Н.Тихонова [10]. Суть метода состоит в использовании дополнительных *априорных* предположений о характере решения. Обычно в качестве таковых используются требования максимальной гладкости функции, представляющей решение задачи. Данный принцип полностью соответствует идее *бритвы Оккама*, согласно которой следует предпочесть простейшее из возможных решений, если нет указаний на необходимость использования более сложного варианта.

В приложении к нейросетевые моделям, регуляризирующие методы сводятся к оптимизации функционала ошибки (в простейшем случае - суммы квадратов уклонений модели от экспериментальных значений) с аддитивной добавкой, исчезающей по мере улучшения свойств гладкости функции:

$$E[G] = \sum_{\alpha=1}^{N_\alpha} \left(G(\vec{x}^{\,\alpha}) - \vec{y}^{\,(\alpha)} \right)^2 + \lambda \phi[G].$$

Здесь φ - регуляризирующий функционал, λ - неотрицательная константа регуляризации.

Замечательной особенностью нейросетевых моделей (аппроксимаций системной функции на основе конечного набора наблюдений) являются их внутренние регуляризирующие свойства, позволяющие получать малые ошибки обобщения. Полезность регуляризирующих свойств нейронных сетей проявляется в ситуациях, когда экспериментальные данные о системе содержат внутреннюю избыточность. Избыточность позволяет представить совокупность данных моделью, содержащей меньшее число параметров, чем имеется данных. Таким образом, нейросетевая модель *сжимает* экспериментальную

информацию, устраняя шумовые компоненты и подчеркивая непрерывные, гладкие зависимости.

Следует отметить, что в случае полностью случайных отображений построение модели с малой ошибкой обобщения *не возможно*. Достаточно рассмотреть простой пример, в котором аппроксимируется отображение фамилий абонентов телефонной сети (вектор входов *X*) в номера их телефонов (вектор выходов *Y*). При любой схеме построения обобщающей модели предсказание номера телефона нового абонента по его фамилии представляется абсурдным.

Имеется обширная научная библиография, посвященная обоснованию оптимального выбора нейроархитектур и переходных функций нейронов исходя из различных видов регуляризирующих функционалов φ (см., например [11] и цитируемую там литературу). Практическая направленность данной главы не позволяет изложить математические детали. Одним из продуктивных подходов к построению нейросетей с хорошими обобщающими свойствами является требование убывания высоких гармоник Фурье переходных функций. Различные законы убывания приводят к локальным сплайн-методам и нейросетям с радиальными базисными функциями.

В случае сигмоидальной переходной функции абсолютная величина коэффициентов Фурье[10] асимптотически быстро убывает. Это свойство отчасти объясняет регуляризирующие свойства популярных многослойных сетей с такими переходными функциями.

Рассмотрим особенности регуляризированных решений обратных задач моделирования описанных систем A, B и C. Обучающая выборка в расчетах содержала 200 пар *x*-*y*, в которых величина *x* случайно равномерно распределена на отрезке [0,1], а значение *y* определяется моделируемой функцией. Расчеты проведены для нейросети с обратным распространением ошибки и нейросети встречного распространения. Еще 500 случайных примеров служили для оценки

[10] Имеется в виду интеграл Фурье в смысле главного значения (интеграл от квадрата сигмоидальной функции, очевидно, расходится).

ошибки обобщения. В трех сериях расчетов величины y из обучающей выборки нагружались внешней шумовой компонентой с амплитудой 0%, 10% и 50% соответственно. Обучение проводилось на обратной зависимости $x(y)$, т.е. величины y использовались в качестве входов, а x - выходов нейросети.

Проведенные расчеты преследовали следующие основные цели:

- выяснение возможности получения оценки некорректности задачи из наблюдений за ошибкой обучения и обобщения,
- изучение роли шума и его влияния на точность оценки степени некорректности,

Результаты моделирования приведены на Рис. 3 - 7.

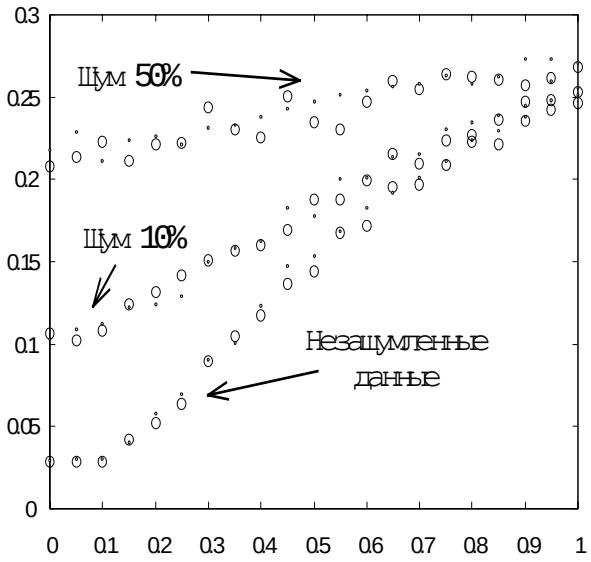


Рис. 3 Зависимость ошибки обучения E_L (кружки) и ошибки обобщения E_G (точки) от степени некорректности h обратной задачи при различных уровнях шума

На Рис. 3 представлено изменение ошибки обучения (и практически совпадающей с ней ошибки обобщения) при росте скачка моделируемой функции. Ошибка при различных уровнях шума прямо пропорциональна величине скачка, определяемого параметром некорректности h. Для сильно

некорректной задачи (*h=1*) результаты полностью не зависят от шума в данных. Теоретически, для неограниченного обучающего набора для моделируемых систем имеется точное (линейное) решение, минимизирующее среднеквадратичное уклонение, которое в предельном случае (*h*=1) дает значение ошибки 0.25. Расчетное значение на Рис.3 в этом наихудшем случае близко к данной теоретической величине.

Таким образом, скейлинг ошибки обучения выявляет степень некорректности задачи независимо от присутствия аддитивного шума в обучающих данных. Данные шум может быть вызван как неточностью измерений, так и эффектом "скрытых" параметров, неучтенных в модели.

На следующем рисунке приведено регуляризованное решение предельно некорректной задачи (*h*=1), даваемое нейронной сетью с обратным распространением, обученной на зашумленных данных.

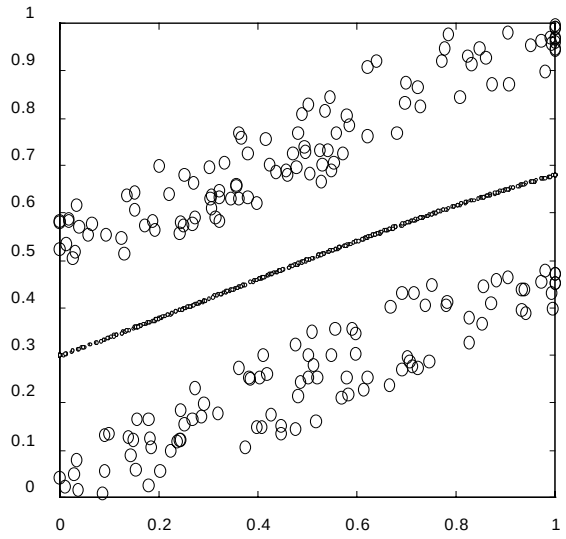


Рис. 4. Регуляризованное решение (точки) предельно некорректной обратной задачи, полученное при помощи нейросети с обратным распространением ошибки на зашумленных данных (кружки).

Решение отвечает минимуму среднеквадратичного уклонения от обучающих данных, что является типичным для сетей с сигмоидальными функциями.

Укажем явно, в чем состоит характер априорных предположений, принимаемых при построении нейросетевых моделей. Единственное предположение (которого оказывается достаточно для регуляризации) состоит в указании базисной архитектуры нейросети с *ограничением ее структурной сложности*. Последнее существенно, т.к., например, при неограниченном увеличении числа нейронов на скрытом слое, сеть способна достаточно точно запомнить дискретный обучающий набор. При этом вместо гладкого решения (Рис.4) будет получено "пилообразное" решение, колеблющееся между двумя ветвями обратной функции, проходя через все обучающие точки.

Дифференцированная оценка степени корректности обратной задачи на основе кластерного анализа сетью Кохонена

Обратная задача может считаться условно корректной, если в признаковом пространстве выходных переменных имеются области, где обратное отображение однозначно (как в случае системы В с промежуточными значениями скачка *h*). Для рассмотренных в предыдущем пункте однопараметрических систем области корректности могут быть выявлены при графическом представлении экспериментальных данных. Отделение областей условной корректности в многомерных пространствах параметров является качественно более сложной задачей. В этом разделе предлагается исследовать возможности нейросетевых алгоритмов адаптивной кластеризации данных для дифференциальных оценок областей условной корректности.

При произвольном распределении точек в многомерном пространстве задача *таксономии* (т.е. разделения всех точек на несколько компактных групп, называемых кластерами) является достаточно сложной, несмотря на то, что имеется целый ряд методов ее решения. Ситуация дополнительно усложняется в важном практическом случае, когда число кластеров заранее не известно.

На классе нейросетевых алгоритмов также предложено несколько подходов [5,6, 12-13]. Классическим является предложенный Т.Кохоненом [5] алгоритм

построения *самоорганизующейся карты*, которая представляет собой отображение многомерного распределения точек на двумерную решетку с регулярным соседством между узлами. При этом близким узлам на карте отвечают близкие вектора в исходном многомерном пространстве, т.е. сохраняется не только структура разбиения точек на кластеры, но и отношения топологической близости между ними.

Если для приложений достаточно только оценки плотности распределения точек по кластерам с сохранением лишь *ближнего порядка* в кластеризации, то такое разбиение может быть выполнено более эффективно на основе модели "*нейронного газа*" [12-13], в которой соседство узлов не фиксировано, а динамически меняется по мере улучшения кластеризации. В относительно недавней модификации метода, получившей название "*расширяющийся нейронный газ*" [13], переменными являются не только отношения соседства, но и число нейронов-кластеров.

В данной главе более подробно рассматриваются приложения более часто используемой карты Кохонена.

Метод дифференциальной оценки степени некорректности задачи

Основная идея предлагаемого метода дифференциальной оценки степени некорректности обратной или комбинированной задачи состоит в реализации следующего плана:

- Построить распределение векторов обучающей выборки по кластерам, содержащим близкие по величине параметров наблюдения. Кластеризация ведется по выходным компонентам Y для чисто обратной задачи, или по совокупности входных и выходных компонент (X,Y) для комбинированного отображения $(X,Y)=F(X,Y)$;
- Провести обучение набора (по числу кластеров) малых нейросетей с обратным распространением на данных каждого кластера, оценить ошибку обучения (и, если в распоряжении имеется достаточно данных, ошибку обобщения). Провести набор статистики по результатам обучения

нескольких вариантов с различными реализациями случайной инициализации весов. Для получения несмещенных оценок следует учесть, что кластеры могут содержать разное число векторов;

- Поставить в соответствие каждому кластеру данных количественную степень некорректности отображения в области данного кластера. В качестве нее может выступать величина, пропорциональная локальной ошибке обучения для данного кластера;
- Выбрать неприемлемый уровень некорректности (в простейшем случае при помощи порогового правила) для построения гибридной системы, аналогичной малым экспертам [4], которая дает регуляризованное решение с локальной оценкой точности в областях с "малой" некорректностью, и предупреждает пользователя о плохой обусловленности задачи, если запрос относится к области "сильной" некорректности.

Важно отметить, что в данном подходе пользователь получает для каждого запроса к нейросетевой модели *адекватную* локальную точность получаемого результата, и *корректный* отказ в выдаче результата в области высокой нерегулярности задачи. Поскольку карта Кохонена дает высокую степень наглядности при изучении распределения экспериментальных данных, то распределение степени некорректности по ней представляет богатый материал для понимания особенностей модели и ее параметров. Неоднородности в "раскраске" карты могут отвечать различным режимам поведения инженерной установки или прибора. При моделировании технических систем это часто может служить указанием на нежелательные (или аварийные!) соотношения параметров при эксплуатации.

Пример выявления области некорректности в модельной задаче

Для иллюстрации предлагаемого метода рассмотрим его применение к уже использовавшимся модельным системам A, B и C. Для простоты рассмотрения (и снижения числа необходимых вычислений) можно применить упрощенный

алгоритм получения оценки некорректности. Для этого вместо использования набора малых экспертов ограничимся одним персептроном (без скрытых слоев), входы которого замкнуты на выходы нейронов карты Кохонена, а число выходов совпадает с размерностью признакового пространства выходов задачи. Такая гибридная нейроархитектура, называемая сетью *встречного распространения*, предложена Р.Хехт-Нильсеном [7-8].

Каждый кластер соревновательного слоя Кохонена в сети встречного распространения включает в себя несколько векторов обучающего множества. Предъявление на вход нейросети некоторого вектора вызывает соревнование в слое Кохонена, при этом в результате остается активным лишь один нейрон, возбуждение которого затормозило все остальные нейроны. Выход победившего нейрона (нормированный на единицу) воспринимается персептроном, в итоге формируется вектор выходов нейросети в целом. Нужно отметить, что все входные вектора в пределах одного кластера неразличимы (т.к. им всем соответствует один и тот же победитель), поэтому выходы сети встречного распространения не изменятся, если при смене входных векторов не произойдет переход от одного кластера к другому. Таким образом, нейронная сеть встречного распространения дает кусочно-постоянное приближение к моделируемой функции.

Уклонение кусочно-постоянной поверхности от значений выходных векторов обучающей выборки, соответствующих входам в пределах заданного кластера принимается за оценку степени некорректности в области этого кластера[11].

[11] Центры кластеров задают разбиение признакового пространства на *многогранники Вороного* (в двумерном случае - на *ячейки Дирихле*). Все точки в пределах одного многогранника ближе к центроиду соответствующего кластера, чем ко всем остальным кластерам.

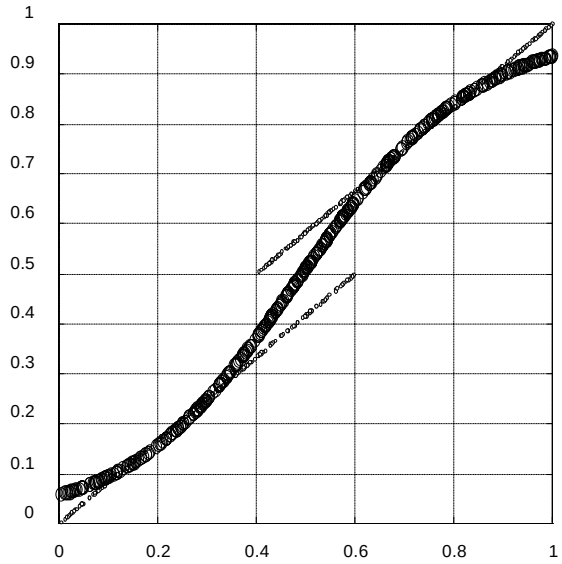


Рис. 5. Гладкое регуляризованное решение (кружки) сетью с обратным распространением ошибки для слабо некорректной задачи двузначного отображения, заданного дискретным набором примеров (точки).

На Рис. 5 и 6 приведено сравнения гладкого регуляризованного решения, определяемого многослойной сетью с обратным распространением, и решения, получаемого при помощи нейросети встречного распространения. Расчеты проведены для системы B для случая относительно слабой некорректности с малым значением величины скачка h.

Легко заметить совершенно различный характер регуляризации, даваемый этими моделями. Уклонение решения от точек обучающего множества в многослойной сети с гладкими переходными функциями охватывает более широкую область, чем собственно область некорректности ($0.4<Y<0.6$). Кривая решения и ошибка гладко распространяются в область, где поведение моделируемой системы регулярно.

В случае сети встречного распространения, напротив, регуляризованное решение содержит минимальные ошибки в области регулярности (разбиение на кластеры заметно только вблизи $Y=0$ и $Y=1$). Решение же в области многозначности функции не является регуляризованным - кластеры со значениями обеих ветвей обратной функции хаотически перепутаны.

Полезность того или иного представления решения может определиться только в контексте конкретного приложения. Для системы, предупреждающей о высокой ошибке решения в области некорректности, по-видимому, следует предпочесть результат сети встречного распространения (Рис. 6), так искажения решения в областях, где это решение имеет смысл, минимальны.

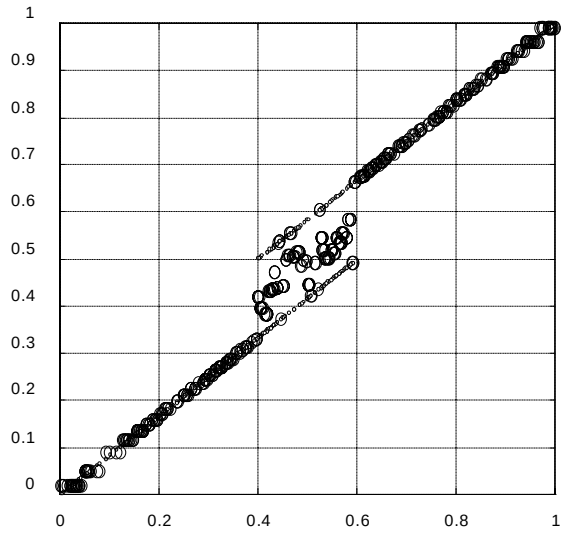


Рис. 6. Кусочно-постоянное в области регулярности решение некорректной обратной задачи, полученное с помощью сети встречного распространения (см. подпись и обозначения на Рис.5).

Обратимся теперь к изучению возможности автоматического выделения области некорректности. В нейронной сети встречного распространения кластеры, расположенные в области некорректности задачи будут содержать близкие вектора, для которых значения моделируемой функции относятся к разным ветвям неоднозначности. Персептрон выходного слоя нейросети в этом случае будет обучаться среднему значению на векторах кластера, поэтому ошибка обучения останется конечной.

В приведенном примере, при h=0.2, теоретическое значение предельной ошибки обучения (среднеквадратичное уклонение) для данных одного кластера равно 0.1. Распределение ошибки по кластерам, наблюдаемое в расчетах,

приведено на Рис. 7. Область некорректности может быть легко автоматически выделена при помощи простого решающего правила.

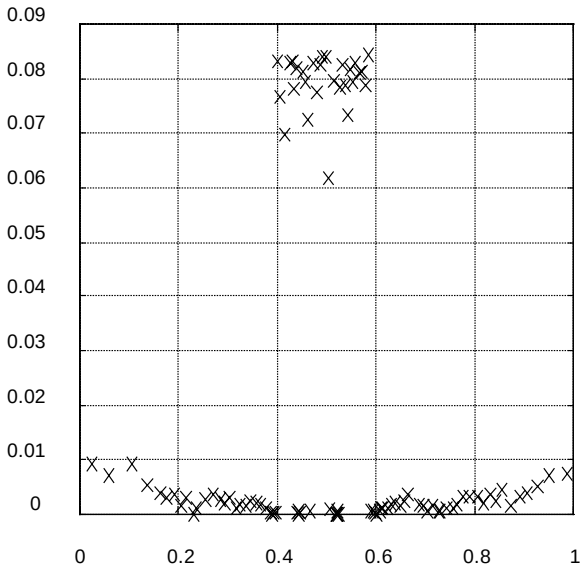


Рис 7. Распределение ошибки обучения по пространственным кластерам Карты самоорганизации Кохонена с легко выделяемой областью некорректности задачи.

Подведем некоторые итоги рассмотрения модельных задач. Можно выделить два основных пути применения нейронных сетей встречного распространения для решения обратных и комбинированных некорректно поставленных задач.

Во-первых, слой самоорганизующихся нейронов карты Кохонена позволяет получить локальную дифференциальную оценку степени некорректности задачи и пространственное распределение ошибки обобщения, делаемой сетью. Кластерное разложение одинаково легко выполняется в признаковых пространствах любой размерности.

Алгоритм кластеризации Кохонена легко обобщается на случай наличия пропусков в данных. Поскольку для отнесения некоторого вектора к кластеру требуется лишь вычислить Евклидово расстояние между этим вектором и текущим приближением к центроиду кластера, и найти кластер с минимальным расстоянием, то при наличии пропущенных компонент в векторе расстояние можно вычислять по имеющимся компонентам. Это эквивалентно поиску

ближайшего кластера в подпространстве известных компонент. Замечательно, что сеть встречного распространения может обучаться даже если в *каждом* обучающем векторе имеются пропущенные компоненты. При этом не требуется заполнения пропусков искусственными значениями.

Второй прикладной аспект состоит в том, что в областях корректности задачи решение, даваемое сетью встречного распространения является весьма точным. Это связано с локальным характером обучения в пределах каждого кластера, и, соответственно отсутствием эффектов равномерного распределения ошибки по кластерам. В этом смысле, регуляризующий эффект сети встречного распространения меньше, нежели у традиционной многослойной сети с обратным распространением.

Прикладное информационное моделирование в задаче оценки риска при эксплуатации сложной инженерной системы

Сложные инженерные устройства при воздействии внешних факторов могут демонстрировать разнообразное нелинейное поведение. С точки зрения эксплуатации таких систем отклик может отвечать нормальному режиму работы, а также аномальным и аварийным режимам. В последнем случае требуется принятие специальных мер для снижения риска последствий инцидента.

Задача информационного моделирования при оценке риска[12] для объекта при внешнем нагружении состоит в необходимости предсказания его поведения в обычных и аномальных условиях.

[12] Проблема оценивания самой величины риска при этом остается за рамками данного рассмотрения. Одним из методов вычисления риска при известных из информационного или математического моделирования режимах работы системы является взвешивание вероятностей реализации различных режимов с экспертными оценками их последствий. Далее управление эксплуатацией системы состоит в минимизации риска.

Описание инженерной системы

Примером задачи оценки риска является эксплуатация контейнера для перевозки или хранения промышленных отходов (например, делящиеся материалы в отработанных твэлах атомных электростанций, или токсичные химические вещества). В качестве аномального внешнего воздействия требуется рассмотрение пожара с различными параметрами. Отклик контейнера (измеряемый оценкой сохранности содержимого и не проникновением его во внешнюю среду) может изменяться в зависимости от текущего состояния системы, например, степенью возможных повреждений в аварийных условиях.

Рассмотрим относительно простую и полезную с практической точки зрения модель, основанную на данных измерений параметров контейнера при различных условиях пожара. Фактически, построенная модель основывалась на результатах численного моделирования, а не на реальных данных. Это, однако, не снижает ценность рассмотрения, поскольку в численных расчетах удается учесть широкий спектр пожаров для одного и того же контейнера. Неопределенность в коэффициентах, описывающих теплофизические свойства материалов, а также численные эффекты, вносят шум в используемые данные, приближая условия моделирования к реальным. В сложившейся практике нейросетевого моделирования данные такого рода называют *реалистичными* (в отличие от *искусственных* и *реальных* данных).

База собранных данных содержит 8 параметров, описывающих контейнер и условия пожара. Признаковое пространство входов состоит из 6 переменных - одной *переменной состояния* контейнера (экспертная оценка степени повреждения контейнера) и пяти *параметров воздействия* (свойств пожара).

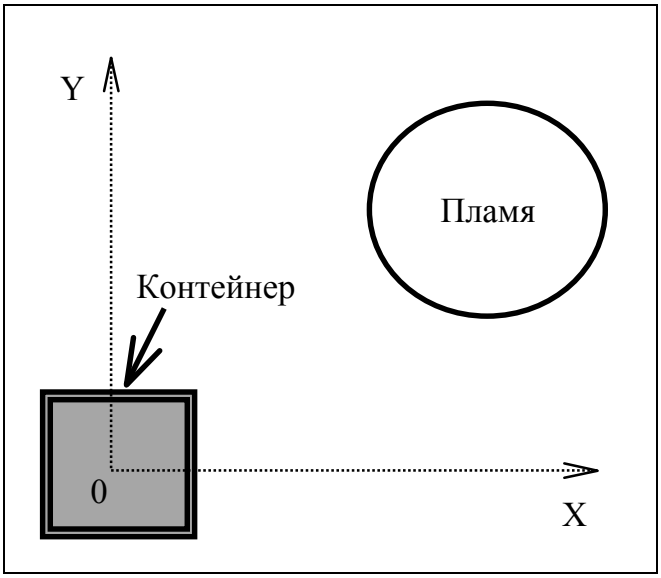


Рис. Схема расположения контейнера и области, охваченной пламенем.

Параметры пожара включают две координаты области пламени, диаметр этой области, температуру пожара и его длительность.

Двумя выходными переменными являются максимальное значение температуры внутри контейнера на протяжении всего пожара, а также отрезок времени, в течении которого температура внутри контейнера превышала некоторое пороговое значение, соответствующее критическому уровню возможного повреждения содержимого контейнера.

Нейросетевая информационная модель системы

Информационная модель отклика контейнера строилась на основе сети встречного распространения и многослойной сети с обучением по методу обратного распространения ошибки. Были рассмотрены прямая, обратная и комбинированная задачи.

Нейронная сеть для прямой задачи содержит 6 входов и 2 оцениваемых выхода. Прямая задача для данного приложения позволят ответить на следующие вопросы:

- Какова будет максимальная температура внутри контейнера при известных параметрах пожара?

- Превысит или нет внутренняя температура заданное критическое значение? Если да, то как долго система будет находиться в критических условиях?
- Что отвечает большему риску повреждения содержимого контейнера: короткий, но высокотемпературный пожар, или длительная умеренная тепловая нагрузка?

Обратная задача соответствует оценке параметров внешнего воздействия по измерениям отклика системы. Тепловой режим внутри контейнера при этом контролируется датчиками температуры. Запросы к обратной модели носят диагностический характер:

- Какова длительность и температура пламени?
- Как далеко от контейнера произошел пожар, и каков был размер пламени?
- Какова фактическая степень повреждения контейнера?

Наиболее интересная комбинированная задача рассматривает часть параметров как известные, а остальные, как неизвестные. При обучении нейросети комбинированной задаче множества переменных, используемых как входные и как выходные, могут частично или полностью перекрываться.

Комбинированная задача отвечает на все запросы прямой и обратной задач, но имеет дополнительные возможности:

- Оценка состояния контейнера по внешним и внутренним измерениям.
- Каковы наитяжелейшие условия пожара, при которых контейнер еще сохраняет содержимое?

Обратную и комбинированную задачи следует рассматривать, как некорректно поставленные.

Область возможных значений физических параметров ограничивалась максимальными температурами пожара (достигаемыми при горении

обогащенного топлива), расстояниями и размерами пламени, при которых теплопередача контейнеру приводит к температуре около 200^{o}C (типичный порог для пожаро-сигнализирующих датчиков). Длительность пожара ограничивалась значением 1 час.

После введения всех ограничений данные из базы данных были линейным преобразованием приведены в "серый" формат [0..1].

Интегральная оценка корректности модели

На первом этапе в предлагаемой технологии исследовалась корректность задачи на всей области значений параметров. С этой целью последовательно выбирались семь параметров из восьми, включенных в модель. Эти параметры считались известными, а оставшийся восьмой параметр - неизвестным. Таким образом, каждый из параметров по очереди тестировался, как неизвестный. При моделировании определялась ошибка обучения многослойной сети с обратным распространением. Все расчеты проводились для нейросетей большого[13] размера, поэтому полученная ошибка связывается только с некорректностью задачи. Был использован следующий вид функции ошибки (называемой процентом среднего квадрата ошибки[14]):

$$E = \frac{100\%}{N_\alpha N_{out}} \sum_{\alpha}^{N_\alpha} \sum_{j}^{N_{out}} \left(Z_j^{(\alpha)} - Y_j^{(\alpha)} \right)^2$$

Численное моделирование показало, что *обе прямые* задачи, когда неизвестными считались выходные переменные задачи - максимальная температура внутри контейнера и длительность периода превышения заданного уровня температуры, являются корректно поставленными. Значение ошибки обучения не превышало 1%. Напротив, все *шесть обратных/комбинированных*

[13] Для определения необходимого числа нейронов на скрытом слое была выполнена серия расчетов с увеличивающимся размером сети. Далее использовалось полученное предельное значение (для нашей базы данных из ~9000 примеров потребовалось 30 нейронов).

[14] Исходный английский термин - *squared error percentage*.

задач оказались некорректными с ошибкой обучения масштаба 25-35%. Данные результат является принципиальным для планирования последующих экспериментов с информационной моделью: попытки оценки решения обратных задач на всей области значений (без дифференциального анализа корректности) будут неудачными.

Прикладная нейросетевая модель для прямой задачи

Для выбора эффективной нейросетевой модели для (корректно поставленной) прямой задачи была изучена зависимость ошибки обучения и обобщения от объема используемых при обучении данных и числа свободных весовых параметров в нейронной сети.

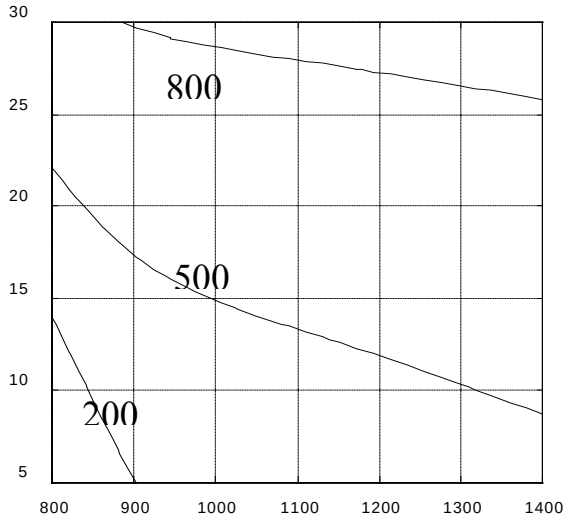


Рис. 9. Области на плоскости "температура пламени" - "длительность пожара", в которых максимальная температура внутри контейнера превышает значения 200, 500 и 800^{o}C.

Для приложений требуется компактная и быстрая нейросетевая модель, легко обучаемая и имеющая невысокую ошибку обобщения. Данные требования в некоторой мере противоречивы, поэтому был суммирован опыт большого числа компьютерных экспериментов. Были обнаружены следующие особенности:

- Степень обобщения улучшается лишь на 50-80% при росте объема базы

обучающих данных (и соответственно, затрат на обучение!) в 3-4 раза. Следовательно, можно избежать больших объемов данных.

- Использование нейронных сетей с число свободных параметров, близким к числу записей в базе данных приводит к ошибке обобщения, в 10 раз большей, чем ошибка обучения. В этом случае предсказательные возможности системы не велики.
- Подходящая нейронная сеть для прямой задачи характеризуется 10-15 нейронами на скрытом слое с масштаба 100 синаптическими связями, обученная на базе данных из 300-500 записей, она показывает ошибку обучения 2-3% при ошибке обобщения до 5%.

На основе выбранной нейросетевой модели было проведено обучение нейросети и исследован ряд информационных запросов к ней.

Первая серия запросов была выполнена для определения области температур и длительностей пожара, при которых содержимое контейнера не перегревается. Рассматривались пожары, происходящие в непосредственной близости к контейнеру и имеющие диаметр до 15 м. Изолинии температур внутри контейнера показаны на Рис.9. Расчеты соответствуют трем различным значениям нагрева содержимого 200, 500 и 800^{o}C. Данные результаты в применении к конкретным образцам контейнеров могут составить основу технических требований к противопожарным службам. Так, например, кривая 200^{o}C показывает параметры, при которых срабатывают типичные температурные датчики. Если критическим для эксплуатации оказывается режим превышения 500^{o}C при температуре пламени 800^{o}C, то, как следует из Рис.9, контейнер способен выдерживать такую нагрузку в течение 22 мин.

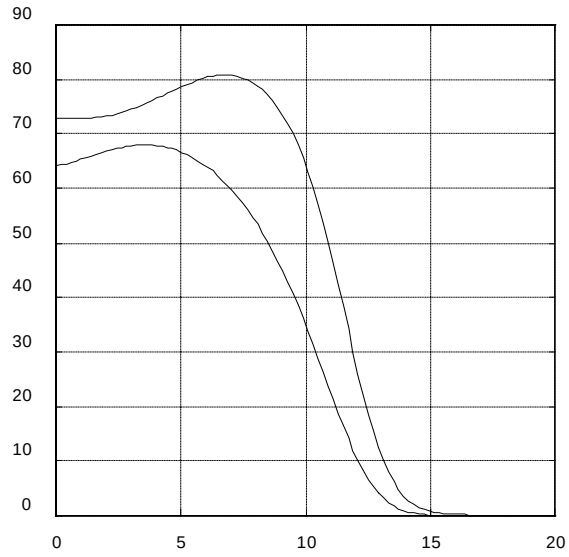


Рис. 10. Зависимость длины промежутка времени (в минутах), в течении которого температура внутри контейнера превышала критический уровень, от расстояния до эпицентра пожара (в метрах) для двух значений диаметра пламени (15 и 20 м).

Вторая рассмотренная задача связана с изучением зависимости тепловых условий внутри системы от расстояния до эпицентра пожара. На Рис. 10 представлена зависимость длительности закритического нагрева (в минутах) от расстояния до области пожара (в метрах). Длительность пожара составляла 1 час. Интересно отметить, что наблюдается некоторое промежуточное значения расстояния до пожара, при котором теплопередача к контейнеру максимальна[15].

На основе данной модели может быть исследовано множество других практических вопросов. Поскольку при анализе запросов нейронная сеть работает только в режиме прямого ненагруженного функционирования, время выполнения запросов минимально.

Возможности регуляризации обратной задачи

Займемся теперь рассмотрением обратной и комбинированной задач. Как мы уже убедились на основе расчетов, эти задачи обладают ярко выраженной

[15] Наличие конечного расстояния, при котором теплопередача максимальна, подтверждается простыми геометрическими соотношениями для однородно изотропно излучающего источника конечных размеров.

некорректностью, связанной с неустранимой неоднозначностью обратной функции, поэтому интерес представляет возможность лишь их частичной регуляризации.

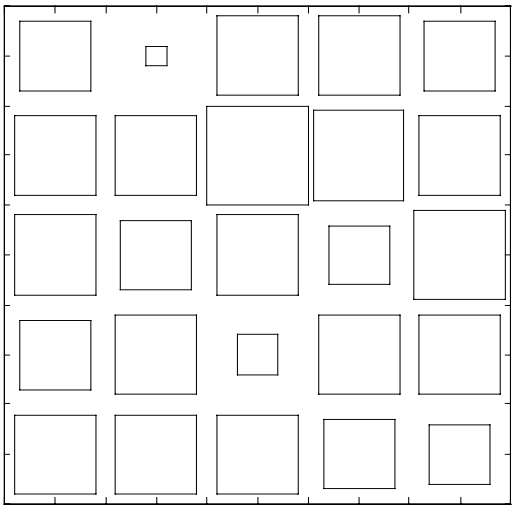


Рис. 11. Распределение ошибок обучения по кластерам карты Кохонена. Ошибка на данных каждого кластера пропорциональна размеру соответствующего квадрата.

Для исследований была выбрана комбинированная задача определения диаметра пламени по остальным параметрам пожара и измерениям внутри контейнера. Эта задача имеет минимальную ошибку обучения (22%) при использовании данных из всей области параметров. Для данной задачи был выполнен кластерный анализ на основе сети встречного распространения. Слой Кохонена представлял собой карту из 5x5=25 нейронов. Рис. 11 отражает распределение ошибки обучения сети по кластерам карты, определяемым нейронами Кохонена.

Результирующее распределение ошибок близко к равномерному, однако имеются две области (кластеры 1-2 и 4-3 в матричных обозначениях) с относительно малыми ошибками. Эти кластеры определяют области частичной регулярности задачи в 7-мерном пространстве параметров.

Наиболее регулярной является область кластера 1-2. Анализ значений параметров, отвечающих центроиду данного кластера, позволяет заключить, что наименьшая ошибка определения диаметра пламени путем решения обратной задачи достигается для высокотемпературных (около 1000^{o}C), длительных (более получаса) пожаров при промежуточных расстояниях до эпицентра (масштаба диаметра пламени).

Следующий шаг исследований состоял в классификации записей в базе данных по построенным кластерам обученной сети Кохонена. Из полного набора измерений около 2.5% данных оказалось относящимися к наиболее регулярному кластеру 1-2. Эти данные были использованы для обучения многослойной сети с обратным распространением с 7 входами и одним выходом. Результирующая ошибка регуляризованного решения составила лишь 8.5%, что приблизительно в три раза меньше ошибки обучения на полном наборе векторов.

Результаты описанных исследований могут быть обобщены в *нейросетевую технологию* решения обратных и комбинированных задач:

- Для данной комбинированной задачи оценивается степень ее некорректности по ошибке обучения нейронной сети с обратным распространением, использующей известные параметры в качестве входов, а запрашиваемые параметры в качестве оцениваемых выходов.
- Если указанная ошибка мала (имеется сходимость нейросетевой аппроксимации), то построенная нейросеть дает искомое решение. Следует далее оценить ошибку обобщения, исходя из априорных теоретических соображений о сложности нейросети, или на основе прямых вычислений с использованием тестовых данных. При недостаточном качестве обобщения можно попытаться уменьшить число нейронов в скрытых слоях нейросети и применить алгоритмы удаления наименее значимых связей.
- В случае неприемлемо больших ошибок обучения применяется уже описанная технология дифференциальной оценки степени некорректности

задачи. После кластерного анализа данных на основе сети Кохонена[16] оценивается распределение ошибок обучения в пространстве параметров модели. Далее строится система малых экспертов, использующих данных отдельных кластеров, или строится более укрупненная оценка на основе сети встречного распространения.

Промышленная нейросетевая модель, созданная по данной технологии будет содержать материнскую сеть Кохонена и семейство малых сетей-экспертов с обратным распространением ошибки. Такая системная модель предоставляет пользователю

- семейство решений для прямых задач
- регуляризованное решение обратных и комбинированных некорректно поставленных задач с оценкой точности, в областях значений параметров с малой локальной степенью некорректности задачи, либо
- диагностическое сообщение о невозможности уверенного прогноза вследствие принадлежности вектора пользовательского запроса области сильной неустранимой нерегулярности задачи.

Предлагаемый подход к нейросетевому моделированию сложных технических систем относительно прост в реализации и непосредственно соответствует ежедневным информационным потребностям инженеров, связанных с эксплуатацией таких систем.

[16] Сеть Кохонена здесь предпочтительнее других алгоритмов кластеризации, так как при образовании относительно больших областей корректности, они будут представлены *макрокластером* пространственно близких нейронов.

Итоги

Подведем итоги этой главы. Нейронные сети являются естественным инструментом для построения эффективных и гибких информационных моделей инженерных систем. Различные нейроархитектуры отвечают различным практическим требованиям.

Сети двойственного функционирования с обратным распространением ошибки и сети встречного распространения обладают хорошими обобщающими свойствами и дают количественные решения для прямых информационных задач.

Внутренние регуляризирующие особенности нейронных сетей позволяют решать также обратные и комбинированные задачи *с локальной оценкой точности*. Для некорректно поставленных задач моделирования предложена нейросетевая информационная технология построения гибридной нейроархитектуры, содержащей кластеризующую карту Кохонена и семейство сетей с обратным распространением, обучаемых на данных индивидуальных кластеров. В этой технологии выявляются области частичной корректности задачи, в которых дается решение с высокой локальной точностью. Для остальных областей признакового пространства нейросеть автоматически *корректно* отвергает пользовательские запросы.

В работе рассмотрены примеры применения методики решения обратных задач к моделированию отклика сложной инженерной системы - промышленного контейнера на внешние аномальные условия (тепловая нагрузка вследствие пожара). Результаты исследований могут быть использованы для технических рекомендаций и требований к противопожарным службам и ресурсам.

Автор благодарен В.В.Легонькову и Л.И.Шибаршову за полезные обсуждения и консультации.

Литература

1. Марчук Г.И. *Методы вычислительной математики*. 3-е изд., М.:Наука,1989
2. Винер Н. *Кибернетика, или Управление и связь в животном и машине*. пер. с англ., 2-е изд., М, 1968.
3. P. Wasserman, *Neurocomputing. Theory and practice*, Nostram Reinhold, 1990. (Рус. перевод. Ф.Уоссермэн. Нейрокомпьютерная техника. М. Мир, 1992).
4. Горбань А.Н., Россиев Д.А. *Нейронные сети на персональном компьютере.* Н. Наука, 1996.
5. T. Kohonen, "Self-organized formation of topologically correct feature maps", *Biological Cybernetics*, Vol. 43, pp.59-69, 1982.
6. T. Kohonen, *Self-Organizing Maps*, Springer, 1995.
7. R. Hecht-Nielsen, "Counterpropagation networks", *Applied Optics*, Vol. 23, No. 26, pp. 4979-4984, 1987.
8. R. Hecht-Nielsen, "Counterpropagation networks", *Proc. First IEEE Int. Conf. on Neural Networks*. eds. M.Candill, C.Butler, Vol. 2, pp.19-32, San Diego, CA: SOS Printing. 1987.
9. D. E. Rummelhart, G. E. Hinton, R. J. Williams, "Learning representations by back-propagating errors", *Nature*, Vol.323, pp.533-536, 1986.
10. Тихонов А.Н., Арсенин В.Я. *Методы решения некорректных задач*. 2 изд. М. Наука, 1979.
11. F.Girosi, M.Jones, and T.Poggio. "Regularization Theory and Neural Networks Architectures". *Neural Computation*, Vol.7, pp.219-269, 1995.
12. T.M. Martinetz, S.G. Berkovich, K.J. Schulten. "Neural-gas network for vector quantization and its application to time-series prediction", *IEEE Transactions on Neural Networks*, Vol. 4, No. 4, p.558-569, 1993.
13. B.Fritzke. "A growing neural gas networks learns topologies", *In Advances in Neural Information Processing Systems 7,* eds. G.Tesauro, D.S.Touretzky, T.K.Leen, MIT Press, Cambridge MA, pp.625-632, 1995.

Глава 5.

Медицинская нейроинформатика

Д.А.Россиев

Отдел медицинской нейроинформатики, КрасГМА

Введение

В последнее время сильно возрастает значение информационного обеспечения самых разных медицинских технологий. Оно становится критическим фактором развития практически во всех областях знания, поэтому разработка и внедрение информационных систем является на сегодняшний день одной из самых актуальных задач.

Анализ применения персональных ЭВМ в медицинских учреждениях показывает, что наибольшее использование компьютеров идет для обработки текстовой документации, хранения и обработки баз данных, ведения статистики и финансовых расчетов. Отдельный парк ЭВМ используется совместно с различными диагностическими и лечебными приборами.

В большинстве этих областей использования ЭВМ применяют стандартное программное обеспечение - текстовые редакторы, системы управления базами данных, статистические пакеты и др. Однако некоторые из важнейших участков лечебно-диагностических технологий практически не используют возможности ЭВМ. Прежде всего это диагностика, назначение лечебных мероприятий, прогнозирование течения заболеваний и их исходов. Причины этого носят чрезвычайно сложный характер и постоянно дискутируются. Основные из них - недостаточно развитая техническая база, низкая компьютерная грамотность участников технологий. Большое значение имеет психологический аспект применения ЭВМ. Это серьезная причина, связанная с характером работы врача. Врач явля-

ется исследователем, и его работа носит творческий характер, однако он несет прямую ответственность за результат своей деятельности. Принимая решение о диагнозе или лечении, он опирается на знания и опыт, свои собственные и коллег, являющихся для него авторитетом. Очень важно при этом обоснование решения, если оно подсказывается со стороны. Результат, подсказанный компьютерной программой, работающей по алгоритму, созданному другим человеком, как показывает практика, во многом лишает исследователя творческой инициативы. Навязанное таким образом решение, даже будучи достоверным, психологически подвергается серьезному сомнению.

Существенную роль играют также особенности медико-биологической информации. Большинство медицинских данных имеют описательный характер, выражаются с помощью формализмов, подверженных крайней вариабельности. Данные, даже выражаемые с помощью чисел, также в большинстве случаев не могут быть хорошо упорядочены и классифицируемы, т.к. изменяются в зависимости от клинических традиций различных школ, геосоциальных особенностей регионов и даже отдельных учреждений, а также от времени.

Все задачи, решаемые человеком, с позиций нейроинформационных технологий можно условно классифицировать на две группы.

1. Задачи, имеющие известный и определенный набор условий, на основании которого необходимо получить четкий, точный, недвусмысленный ответ по известному и определенному алгоритму.
2. Задачи, в которых не представляется возможным учесть все реально имеющиеся условия, от которых зависит ответ, а можно лишь выделить приблизительный набор наиболее важных условий. Так как часть условий при этом не учитывается, ответ носит неточный, приблизительный характер, а алгоритм нахождения ответа не может быть выписан точно.

Для решения задач первой группы с большим успехом можно использовать традиционные компьютерные программы. Как бы ни был сложен алгоритм, ограниченность набора условий (входных параметров) дает возможность составления алгоритма решения и написания конкретной программы, решающей данную задачу. Нет никакого смысла в использовании нейроинформационных тех-

нологий для решения таких задач, так как в этом случае нейросетевые методы будут априорно хуже решать такие задачи. Единственным исключением является случай, когда алгоритм вычисления ответа слишком большой и громоздкий и время на решение конкретной задачи по этому алгоритму не удовлетворяет практическим требованиям; кроме того, при получении ответа не требуется абсолютная точность.

При решении задач второй группы применение нейротехнологии оправдывает себя по всем параметрам, при выполнении, однако, двух условий: во-первых, наличия универсального типа архитектуры и единого универсального алгоритма обучения (отсутствие необходимости в их разработке для каждого типа задач), во-вторых, наличия примеров (предыстории, фиксированного опыта), на основании которых производится обучение нейронных сетей. При выполнении этих условий скорость создания экспертных систем возрастает в десятки раз, и соответственно снижается их стоимость.

Практически вся медицинская и биологическая наука состоит именно из задач, относящихся ко второй группе, и в большинстве этих задач достаточно легко набрать необходимое количество примеров для выполнения второго условия. Это задачи диагностики, дифференциальной диагностика, прогнозирования, выбора стратегии и тактики лечения и др. Медицинские задачи практически всегда имеют несколько способов решения и “нечеткий” характер ответа, совпадающий со способом выдачи результата нейронными сетями.

Разработка математических методов решения медико-биологических задач ведется уже не одну сотню лет. Учеными предложено огромное количество способов проверки гипотез и продукции выводов. В 60-е годы были разработаны методы анализа, получившие некоторое распространение и вызвавшие волну публикаций. Общим признаком, объединяющим их, является наличие явных алгоритмов принятия решений. “Диагностический алгоритм включает в себя совокупность правил, определяющих порядок переработки медицинской информации с целью постановки диагноза” [1]. Несмотря на то, что наиболее популярные методы до сих пор активно используются в теоретической биологии и медицине, в клинической практике они не нашли широкого применения. Это связано, во-

первых, с тем, что методы, ориентированные на обработку групповых данных, слабо применимы к отдельным объектам, а во-вторых, с особенностями самой медико-биологической информации. Решения в медицинских и биологических задачах зависят от большого количества неодинаковых по значимости факторов. Поэтому, даже если удается выстроить правила вывода, связывающие условия задачи с решением, метод, как правило, хорошо работает только на той группе объектов, на которой производились исследования. Естественно, создать универсальный алгоритм невозможно, и при использовании метода для другой подобной группы объектов его почти всегда приходится полностью переконструировать практически заново.

Многолетние исследования, проводимые с самыми различными явными алгоритмами, показали, что медицинские задачи, имеющие неявный характер, решаются явными методами с точностью и удобством, совершенно недостаточными для широкого практического использования в конкретных задачах диагностики, прогнозирования и принятия решений [2].

Поиски и изучение неявных алгоритмов, позволяющих автоматически накапливать и затем использовать опыт при обучении [3], продолжаются уже более 100 лет [4]. Однако первые серьезные попытки создания нейронных сетей были сделаны в 40-50-х годах, когда У.Маккалох и У.Питтс выдвинули основные положения теории работы головного мозга. С появлением дешевых ЭВМ произошел резкий скачок в этой области, которая в начале 80-х годов сформировалась в целую науку - нейроинформатику [5,6,7].

Неявные задачи медицины и биологии явились идеальным полем для применения нейросетевых технологий, и именно в этой области наблюдается наиболее яркий практический успех нейроинформационных методов.

Рассмотрим несколько наиболее интересных нейросетевых приложений для биологии и медицины, созданных различными авторами и школами.

Наибольший интерес для практического здравоохранения представляют системы для диагностики и дифференциальной диагностики заболеваний. При этом для принятия решений могут использоваться самые разнообразные данные - анамнез, клинический осмотр (создаются экспертные системы диагностики,

ограничивающиеся только этим набором [8]), результаты лабораторных тестов и сложных функциональных методов. Список областей медицины, в которых начали применяться новые технологии, чрезвычайно обширен и продолжает расти.

Одним из наиболее интенсивно развиваемых направлений является применение нейросетей в кардиологии.

В Италии разработана чрезвычайно интересная экспертная система для диагностики и лечения артериальной гипертонии [9]. Система включает в себя три нейросетевых модуля, причем ответы одних являются входными данными для других. В начале исследования больному проводят измерение систолического и диастолического давления каждые полчаса в течение суток. Данные за каждый час усредняются. Таким образом, образуется массив из 48 величин артериального давления (по 24 для систолического и диастолического). После этого первый модуль, состоящий из двух трехслойных нейросетей (в каждой из которых 2 входных, 4 "скрытых" и 24 выходных нейрона), на основании данных о поле и возрасте больного рассчитывает аналогичные "должные" величины и сравнивают их с реальными. Параллельно второй модуль (двухслойная нейросеть с 17 входными и 4 выходными нейронами) на основании клинических данных (симптоматика, анамнез) рассчитывает возможные сочетания гипотензивных лекарственных средств, которые могут быть использованы для лечения данного больного. Данные, снятые с выходов обоих модулей, вместе с клиническими данными подаются на вход последнего, третьего модуля (6-слойная нейросеть). Этот модуль оперирует 4 группами гипотензивных препаратов (диуретики, бетаадреноблокаторы, ингибиторы ангиотензина, блокаторы кальциевых каналов). Цель - назначить суточный (почасовой) график приема больным лекарств каждой (если требуется) из 4 групп. Поэтому этот модуль имеет 96 выходных нейронов (4 препарата х 24 часа). С каждого выходного нейрона снимается доза, соответствующая одному препарату, назначаемому на данный час суток. Естественно, что в реальной ситуации большинство выходных данных равны нулю. Таким образом, создается оптимальная для пациента схема лечения гипертонии. Нужно отметить, что система учитывает некоторые особенности приема препаратов больными, например, затруднение приема препаратов ночью (назначает

ночной прием только в крайних случаях), запрет на назначение мочегонных лекарств на ночь. Отличительной чертой системы является возможность пользователя (врача) передавать нейронной сети свой опыт. Для этого создателями программы предусмотрен специальный блок, который выводит на экран компьютера суточные кривые артериального давления и предлагает врачу ввести в компьютер суточную схему приема гипотензивных препаратов в необходимых, по его мнению, дозах. Введенный пример помещается в базу данных. В любое время можно инициировать доучивание нейронных сетей с новыми примерами.

В одной из работ приводится метод выявления атеросклеротических бляшек в артериях [10]. Для этого применяется нейросеть, интерпретирующая флюоресцентные спектры, получаемые при исследовании тканей с помощью лазера.

Аналогичным образом проводится диагностика заболеваний периферических сосудов [11], например, определение форм артериита [12].

Проводится комплекс исследований по использованию нейросетей для диагностики инфаркта миокарда [13,14,15]. Автор приводит данные по чувствительности (77,7%) и специфичности (97,2%) нейросетевого теста. В работе [16], кроме того, с помощью нейронной сети устанавливали диагностическую значимость клинических параметров при диагностике инфаркта миокарда.

Нейросетевой анализ акустических сигналов позволяет проводить диагностику клапанных шумов сердца [17], и оценивать систолическую и диастолическую фазы сердечного сокращения с постановкой предварительного диагноза [18].

Нейросети используются терапевтами для диагностики заболеваний печени по лабораторным данным исследования функций печени [19]; дифференциальной диагностики заболеваний печени [20] и желчного пузыря по УЗИ [21].

Нейропрограммы могут с успехом работать с медицинскими данными, относящимися к субъективным категориям, например, в психиатрии [22]. Оценка субъективных данных дает возможность распознавания психических симптомов] и диагностики и изучения некоторых психиатрических симптомокомплексов.

Актуальная проблема диагностики злокачественных новообразований, возможно, получит новый уровень осмысления с началом применения нейроалгоритмов. Так, в работе [23] показана 80%-я точность ранней диагностики меланом кожи - одного из самых злокачественных заболеваний.

Одним из серьезных направлений применения нейронных сетей является интерпретация медицинских данных. В последние годы идет бурное развитие новых средств диагностики и лечения. При этом наблюдается "вторая волна" изучения и использования древних, старинных методов, и, наоборот, применение последних технических новшеств. Нередко и те и другие методы при использовании предоставляют врачу массу самых разнообразных данных. При этом встает проблема их грамотной и корректной интерпретации. Поиск глубинных закономерностей между получаемыми данными и патологическими процессами начинает отставать от разработки все новых и новых методов, поэтому применение для этой цели нейросетей может оказаться чрезвычайно выгодным.

В одной из старинных методик диагностики по пульсу используются 14 характеристик пульса, измеряемых с нескольких точек. Распознавание и интерпретация данных требует огромного опыта врача, практически невозможного в современных условиях. Нейросеть была применена для "узкой" диагностики только по одной из точек [24], позволяющей оценивать состояние левой почки. Пульс считывается специальным датчиком, совмещенным с микрофоном. Полученная пульсовая кривая (сфигмограмма) передается в компьютер. Вначале программа анализирует несколько пульсовых волн и выстраивает "среднюю" волну. После этого по 5 точкам этой волны нейронная сеть оценивает состояние левой почки.

Классической проблемой в кардиологии является интерпретация электрокардиограмм, требующая значительного опыта врача. Сотрудники Университета Глазго (Великобритания) ведут исследования по применению нейросетей для ЭКГ-диагностики инфарктов миокарда [25]. Входными данными для сетей являются избранные параметры 12-канальной электрокардиограммы и 12-канальной векторкардиограммы (длины зубцов, расстояния между зубцами). Исследователи обучили огромное количество нейросетей (167 сетей для диагностики инфаркта

миокарда передней стенки и 139 сетей для инфаркта нижней стенки) на массиве данных из 360 электрокардиограмм. Обученные сети затем тестировали отдельную выборку с заранее известными ответами (493 случая). Одновременно для получения отдельной серии ответов на тестируемой выборке был использован логический метод (с заранее заданным алгоритмом). Затем сравнивались результаты тестирования выборки лучшими нейросетями и с помощью логического алгоритма. Сравнение показало, что во многих случаях чувствительность и специфичность нейросетевого теста оказались выше, чем у логического метода. Авторы делают справедливый вывод, что в случаях, когда логический алгоритм решения задачи все-таки можно выстроить, разумно комбинировать в экспертных системах оба подхода.

Интерпретация ЭКГ с помощью нейросетей была применена для диагностики злокачественных желудочковых аритмий [26]. Трехслойная сеть с 230 входными синапсами была обучена на 190 пациентов (114 с хронической сердечной недостаточностью и 34 с дилятационной миокардиопатией) различать наличие (у 71 пациента) и отсутствие (у 119 пациентов) желудочковой тахикардии. Результаты тестирования сравнивались с логическим методом интерпретации данных. Показано, что нейросетевой тест обладает большей чувствительностью (73% по сравнению с 70 для логического метода) и специфичностью (83 и 59%).

Интересная работа описывает моделирование применения нейросетей для работы электрокардиостимуляторов (искусственных водителей ритма) [27]. Выпускаемые за рубежом электрокардиостимуляторы задают ритм не жестко, а в зависимости от исходного ритма, генерируемого синусовым узлом сердца. Например, если синусовый узел при какой-либо патологии генерирует недостаточное количество импульсов, водитель ритма компенсирует ритм. Таким образом, электрокардиостимулятор представляет собой систему вход→преобразование→выход, где входом является ритм синусового узла, выходом - собственный ритм электрокардиостимулятора, а преобразование осуществляется по заданному логическому алгоритму. Авторы смоделировали замену логического преобразователя нейронной сетью, так как взаимоотношения между

генерацией импульсов в синусовом узле и требуемым ритмом не линейны и применяемые алгоритмы на практике не всегда эффективны. Нейросеть, обученная на 27 здоровых людях в ситуациях с различной физической нагрузкой, показала гораздо лучшую способность задавать ритм, чем логический алгоритм, применяющийся в электрокардиостимуляторе.

Одной из самых сложных задач для нейросетей в практической медицине является обработка и распознавание сложных образов, например рентгенограмм. В работе [28] описывается экспертная система интерпретации рентгенограмм груди у новорожденных с выбором одного и более диагнозов из 12.

Созданы нейросетевые экспертные системы для классификации опухолей молочной железы (определения, доброкачественная опухоль, или злокачественная) по данным маммографии (сканограмма молочной железы) [29]. По данным, которые приводят авторы, точность такого вывода до применения нейросети составляла не более 75%. При тестировании системы, нейросеть, анализирующая сканограмму, давала правильный ответ в 100% случаев. При тестировании изображение, получаемое в результате метода, представляется в виде матрицы точек размером 1024x1024 пиксела с 10-битовой шкалой яркости. Изображение подается на нейросеть, имеющую 2 входных, 80 "скрытых" и 2 выходных нейрона. При этом один из выходных нейронов "отвечает" за доброкачественную опухоль, другой за злокачественную. Диагноз определяется в зависимости от выходного нейрона, выдавшего больший по величине ответ. Столь высокий процент правильности распознавания, возможно, случаен, и объясняется недостаточным количеством примеров, использовавшихся при обучении и тестировании нейросети (по 10 примеров). Однако даже при такой малой обучающей выборке нейросеть выигрывала по сравнению с традиционным методом интерпретации сканограммы.

Несколько работ посвящены нейросетевой обработке лабораторных анализов и тестов. Приводится нейросетевой метод интерпретации лабораторных данных биохимического анализа крови [30]. В работе показаны преимущества нейронных сетей в сравнении с линейным дискриминантным анализом, которым параллельно обрабатывались данные.

Особое место среди нейросетевых экспертных систем занимают прогностические модели, применяемые, например, для прогнозирования исходов заболеваний.

В 1990 году американская фирма "Апачи Медикл Системз Инк." установила в реанимационном отделении одной из больниц штата Мичиган экспертную систему "Апачи - III" [31]. Ее цель - прогнозирование исхода заболевания у больных, находящихся в тяжелом состоянии. Для прогноза в компьютер необходимо ввести 27 параметров больного: первичный диагноз, симптомы, степень утраты сознания, наличие или отсутствие СПИД и других заболеваний. После этого система выдает вероятность выживания больного в диапазоне от 0 до 100 процентов. Ценность применения системы заключается в том, что она позволяет очень быстро оценить динамику изменения состояния больного, незаметную "на глаз". Например, можно получить ответ у системы до и после введения какого-либо лекарства, и, сравнив ответы, посмотреть, будет ли наблюдаться эффект от терапии. Без программы же изменение состояния иногда не удается обнаружить в течение нескольких дней. Тестирование показало, что 95% прогнозов, которые делает программа, сбываются с точностью до 3%, что значительно точнее, чем у лучших врачей. Необходимо отметить, что система была обучена на данных, взятых из историй болезней 17448 пациентов, лечившихся в 40 больницах штата в 1989 году. Очевидно, что если качество работы системы обеспечивается таким большим объемом выборки, возможности перенастройки системы не слишком велики. Идеология авторов, создавших эту систему, заключается в как можно большем охвате различных примеров и вариантов (сбор данных в 40 больницах), а не в возможности индивидуализации системы к конкретной клинике. Поэтому данная система не способна к подучиванию в процессе работы, опыт "зашит" в нее жестко. Это может быть существенным недостатком при установке программы в регионы, резко отличающиеся по социально-географическим условиям от тех, где проводилось обучение. Кроме того, огромный массив примеров для обучения повышает стоимость программы.

Прогностические нейросетевые модели могут использоваться в демографии и организации здравоохранения. Создана экспертная система, предсказы-

вающая, умрет ли человек (в возрасте 55 лет и старше) в ближайшие 10 лет. Прогноз делается по результатам ответов на 18 вопросов анкеты. В анкету включены такие вопросы, как раса, пол, возраст, вредные привычки, семейное положение, семейный доход. 4 из 18 вопросов выявляют индекс массы тела (body mass index) в различные периоды жизни респондента. Индекс рассчитывается как отношение веса к квадрату роста (индекс более 27 кг/м считается тучностью). Повышенное внимание к этому показателю говорит о его значимости для прогноза жизни.

Развитие нейросетевых методов дает возможность их использования как инструмента научных исследований, с помощью которого можно изучать объекты и явления.

Судя по литературным данным, именно биологические научные исследования являются наиболее развиваемой областью применения нейросетей [32]. В последнее время биологи, знакомые с исследованиями в области нейроинформатики, приходят к выводу, что многие системы в живых организмах работают по принципам, сходным с алгоритмами нейронных сетей (или наоборот, нейронные сети работают по принципу биосистем). Таким образом, можно наблюдать "взаимное стимулирование" научных разработок в биологии и нейроинформатике. В работе [33] эндокринная система человека рассматривается как нейронная сеть из 30 элементов, которые представлены различными гормонами, взаимодействующими друг с другом с помощью прямых и обратных связей. Похожие исследования проводятся для иммунной системы [34].

Применение нейросетей для исследований в области нейрофизиологии строится на похожих принципах функционирования нейросетей и нервных структур живых организмов [35]. С помощью нейросети осуществлена попытка моделирования простейшей нервной системы [36].

Сделана попытка применения нейросети для классификации живых организмов [37]: нередко биологам, открывающим новые виды организмов, требуется определить, к какому виду (классу, типу) относится тот или иной представитель флоры или фауны (как правило, это касается микроорганизмов и растений). Система способна работать при отсутствии некоторых входных данных. Это яв-

ляется существенным преимуществом, так как часто при изучении живых объектов не всегда возможно получить всю необходимую информацию.

Нейросети использованы для идентификации человеческих хромосом. В биологических исследованиях, а также в криминалистике, часто бывает нужно определить, к какой из 23 имеющихся у человека пар хромосом относится выделенная хромосома. Точность существующих методов достигала 75 - 85%. Нейроклассификатор, на вход которого подается 30 признаков изображения хромосомы, определяет ответ с точностью, приближающейся к 100% [38].

Анализ публикаций о применении нейросетевых технологий в медицине показывает, что практически отсутствуют какие-либо методологии разработки нейросетевых медицинских систем, о чем свидетельствует как отсутствие работ такого профиля, так и огромное разнообразие подходов к нейросетевым алгоритмам обучения и архитектурам нейронных сетей. Это подтверждает то, что медицинская нейроинформатика как наука находится еще, в основном, на стадии накопления фактического материала.

Нужно отметить, что все медицинские приложения нейронных сетей для практического здравоохранения (диагностика, лечение, прогнозирование) созданы зарубежными авторами. Большинство отечественных работ направлено на исследование самих нейронных сетей и моделирование с их помощью некоторых биологических процессов (в основном, функций нервной системы).

Общая черта, объединяющая приведенные примеры - отсутствие единой универсальной технологии создания таких приложений. В публикуемых разработках используются самые разнообразные архитектуры и алгоритмы функционирования нейронных сетей. Это приводит к тому, что для почти для каждой задачи разрабатывается своя собственная архитектура, и часто - уникальный алгоритм или уникальная модификация уже существующего. С точки зрения практического применения такие экспертные системы почти не отличаются от традиционных программ принятия решений; предложены даже методы преобразования традиционных экспертных систем в нейросетевые [39]. Их разработка требует участия специалистов по нейроинформатике, а возможности конструирования пользователем практически отсутствуют. Это делает такие системы чрезвычайно

дорогими и не очень удобными для практического применения, поэтому в публикациях авторы в основном сравнивают качество работы нейросетевых алгоритмов и традиционных систем, работающих по правилам вывода.

Что же можно предложить взамен? Об этом и пойдет речь ниже. На протяжении нескольких лет красноярская научная группа «НейроКомп» [40], объединяющая ученых-математиков и медиков (Красноярский ВЦ СО РАН, КГТУ, Красноярская Медицинская Академия и ряд клинических учреждений города), разрабатывает технологии создания нейросетевых экспертных систем, которые применяются в практической медицине и биологии на обычных IBM-совместимых компьютерах [41,42,43].

Наиболее важным отличием предлагаемого подхода является возможность конструирования экспертных систем самим врачом-специалистом, который может передать нейронной сети свой индивидуальный опыт, опыт своих коллег, или обучать сеть на реальных данных, полученных путем наблюдений. При использовании разработанного нами пакета программ MultiNeuron 2.0 [44] для конструирования экспертной системы не требуется участие специалистов-математиков и программистов, что делает создаваемые системы более дешевыми, а главное, адаптированными к конечному пользователю.

Далее мы рассмотрим основные принципы и особенности этих технологий.

Все неалгоритмируемые или трудноалгоритмируемые задачи, решаемые нейронными сетями, можно классифицировать на два принципиально различающихся типа в зависимости от характера ответа - задачи классификации и задачи предикции.

Задачи классификации - основная и очень обширная группа медико-биологических задач. Ответом в них является класс - выбор одного варианта из заранее известного набора вариантов. Классификация может быть бинарной (элементарная классификация) - в этом случае набор возможных ответов состоит из двух вариантов (классов), и n-арной, где число классов более двух. Примерами бинарной классификации могут служить как объективные категории (пол чело-

века - мужской или женский; характер опухоли - доброкачественный или злокачественный), так и субъективные категории (здоров человек или болен; наличие или отсутствие склонности к простудным заболеваниям). В некоторых случаях не представляется возможным отнесение ответа задачи к объективной или субъективной категории, и это не имеет принципиального значения для обучения и работы нейросетевой экспертной системы.

Важной чертой задачи классификации по определению является возможность выбора одного и только одного варианта решения (класса). Поэтому постановка диагноза не может считаться одной классификационной задачей, т.к. у одного человека может одновременно присутствовать несколько патологий. В случае невозможности выбирать один вариант ответа (множественности выбора) задача подразделяется на подзадачи, каждая из которых представляет собой классификационную задачу.

Другой вид задач для нейросетей - задачи предикции, или предсказания. Они подразделяются на предсказание числа (одномерная предикция) и вектора (векторная предикция, более общий случай). Отличие от классификационных задач заключается в том, что ответ в задачах предикции может быть дробным и принимать любые значения на каком-либо интервале.

Векторная предикция предполагает, что ответ может быть представлен в виде нескольких независимых друг от друга чисел, образующих точку (или вектор) в многомерном пространстве, размерность которого равно количеству предсказываемых чисел. Число координат вектора называется при этом размерностью вектора ответа.

При решении реальных задач возможны различные комбинации предикции и классификации, и постановка задачи должна быть сделана самим предметным специалистом.

Архитектура нейронной сети

Основой работы самообучающихся нейропрограмм является нейронная сеть, представляющая собой совокупность нейронов - простых элементов, связанных между собой определенным образом. Нейроны и межнейронные связи

задаются программно на обычном компьютере или могут иметь "материальную" основу - особую микросхему (нейрочип), которые применяются в специально созданных нейрокомпьютерах. Структура взаимосвязей между нейронами в нейрокомпьютере или нейропрограмме аналогична таковой в биологических объектах. Искусственный нейрон имеет коммуникации с другими нейронами через синапсы, передающие сигналы от других нейронов к данному (дендриты) или от данного нейрона к другим (аксон). Кроме того, нейрон может быть связан сам с собой. Несколько нейронов, связанных между собой определенным образом, образуют нейронную сеть.

Нейросеть, также как и биологический аналог, должна иметь каналы для связи с внешним миром. Одни каналы обеспечивают поступление информации из внешнего мира на нейросеть, другие выводят информацию из нейросети во внешний мир. Поэтому одни нейроны сети рассматривается как входные, другие же - как выходные. Часть нейронов может не сообщаться с внешним миром, а взаимодействовать с входными, выходными и такими же нейронами ("скрытые" нейроны).

Очевидно, что существует огромное количество способов соединения нейронов, растущее с увеличением числа нейронов в сети. Наиболее употребительной является слоистая архитектура, в которой нейроны располагаются “слоями”. В наиболее общем случае аксоны каждого нейрона одного слоя направлены к нейронам следующего слоя. Таким образом, нейроны первого слоя являются входными (принимающими информацию из внешнего мира), нейроны последнего слоя - выходными (выдающими информацию во внешний мир). Схема трехслойной сети изображена на рисунке 1.

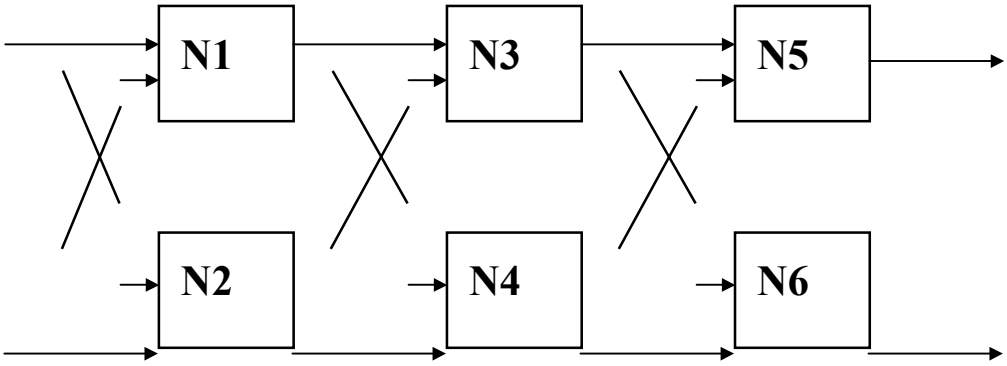


Рисунок 1. Трехслойная сеть с 6 нейронами

Другой вид архитектуры - полносвязная, когда каждый нейрон соединен с каждым, в том числе сам с собой. Пример простейшей нейросети из 3 нейронов показан на рисунке 2. Для удобства изображения из каждого нейрона выходит не один, а несколько аксонов, направленных на другие нейроны или во внешний мир, что аналогично присоединенным к одному аксону через синапсы нескольким дендритам.

Слоистые сети являются частными случаями полносвязных.

Для построения экспертных систем мы выбрали именно полносвязные нейросети, исходя из следующих соображений. Во-первых, при одинаковом числе нейронов полносвязные сети имеют большее количество межнейронных связей, что увеличивает информационную емкость сети. Во-вторых, полносвязная архитектура является намного более универсальной, что не требует экспериментов с вариациями схемы соединений для каждой задачи. В-третьих, в случае эмуляции сети на обычной ЭВМ полносвязные сети обладают серьезными преимуществами, прежде всего в скорости функционирования и простоте программной реализации без ущерба качеству обучаемости.

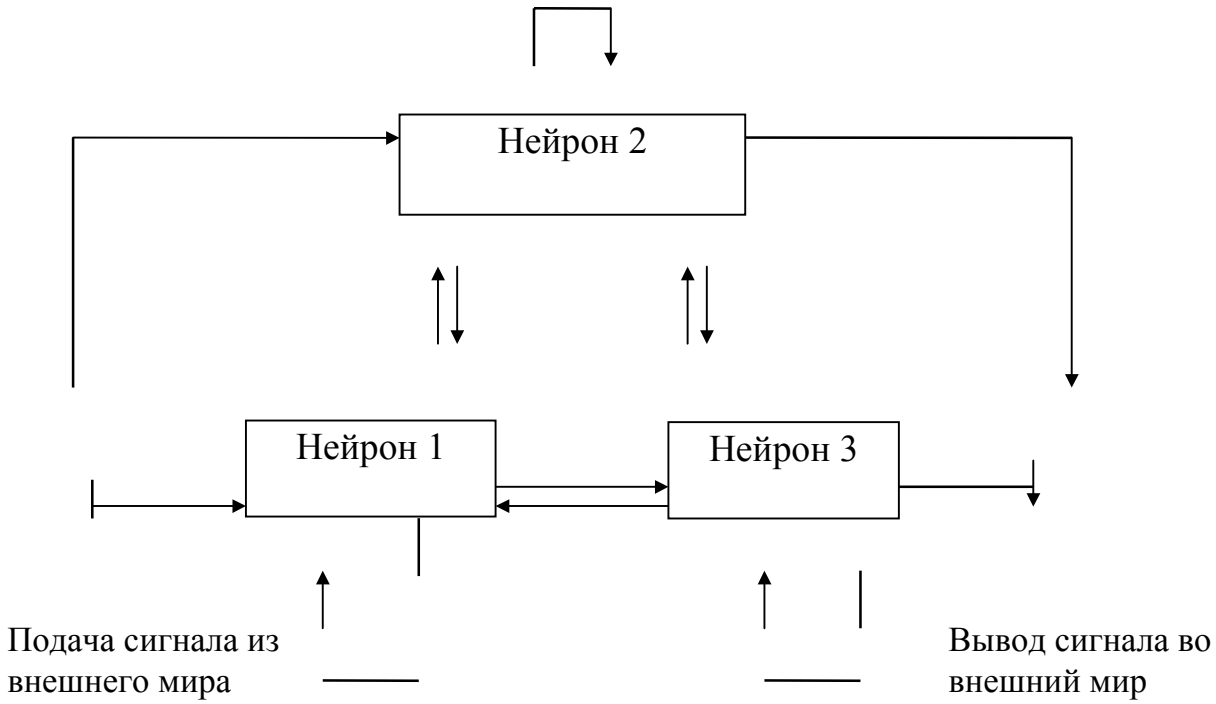


Рисунок 2. Схема простейшей нейронной сети из 3 нейронов. Сеть имеет 13 синапсов, 4 из которых служат для связи с внешним миром, а остальные соединяют нейроны между собой

Функционирование нейрона

Рассмотрим устройство и функционирование отдельного нейрона. Каждое соединение от нейрона к нейрону называется синапсом. На рисунке 3 представлен нейрон с группой синапсов, соединяющих нейрон либо с другими нейронами, либо с внешним миром. Для рассмотрения работы нейрона неважно, приходит ли сигнал к нейрону из внешнего мира или с другого нейрона, и неважно, куда отправляется сигнал с нейрона. В полносвязных сетях выходной сигнал направляется всем остальным нейронам.

Нейрон состоит из двух функциональных блоков: входного сумматора и собственно нейрона, или преобразователя.

Рисунок 3. Схема нейрона

Функционирование нейрона происходит следующим образом.

В текущий момент времени через входные синапсы (на рисунке их 3) на нейрон направляются сигналы от других нейронов и/или из внешнего мира. Каждый синапс имеет параметр, называемый весом синапса, и представляющий какое-либо число. Сигнал, проходящий через синапс, умножается на вес этого синапса. В зависимости от веса, сигнал может быть усилен (модуль веса > 1) или ослаблен (модуль веса < 1) по амплитуде. Сигналы от всех синапсов, ведущих к данному нейрону, принимает сумматор.

Сумматор производит суммирование всех пришедших сигналов и подает на собственно нейрон (преобразователь) одно число - полученную сумму. Величина этого числа будет зависеть как от величин исходных сигналов, так и от весов синапсов. Нейрон, получивший это число, преобразует его согласно своей функции, в результате которой получается другое число, и отправляет его по "аксону" всем остальным нейронам через соответствующие синапсы. Последующие нейроны производят с полученными сигналами такие же операции, лишь с тем различием, что во-первых, веса их синапсов могут быть уже другими, во-вторых, другие нейроны могут иметь другой вид функции преобразования. В конструируемых нами нейронных сетях все нейроны имеют одну и ту же функцию. Эта функция, называемая характеристической, имеет вид:

$$f(X) = X/(C+|X|) \qquad (1)$$

где X - сигнал, поступаемый с сумматора, C - константа, называемая характеристикой нейрона. Экспериментальным путем мы получили, что оптимальный диапазон характеристики для решения подавляющего большинства задач составляет от 0,1 до 0,8. График характеристической функции представлен на рисунке 4.

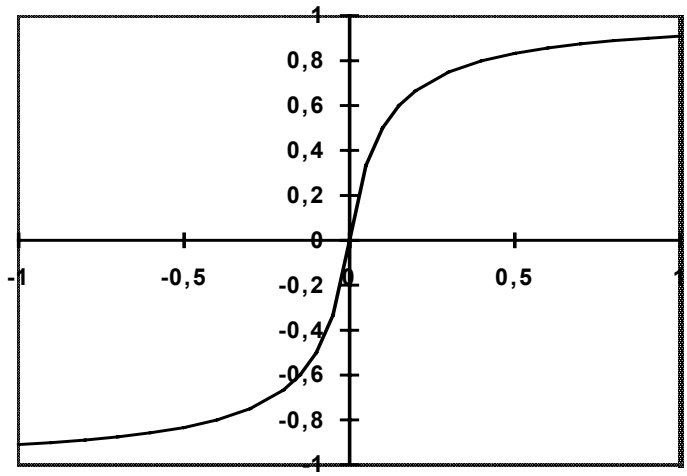


Рисунок 4. График характеристической функции

Выбор функции такого вида обусловлен тем, что она гладкая, непрерывная на всем диапазоне переменных X, диапазон значений всегда ограничен.

Функционирование нейросети

В случае эмуляции нейросети на обычном компьютере все математические операции осуществляет программа. Нейронная сеть при этом представляет собой массив синаптических весов. Этот массив может находиться либо на диске компьютера в виде файла определенного формата (при хранении нейросети) либо в оперативной памяти компьютера (когда нейросеть функционирует).

При создании новой нейросети в памяти компьютера отводится место под массив синаптических весов, называемый картой. Этот массив заполняется совершенно случайными числами из определенного диапазона. Поэтому каждая созданная сеть даже при одних и тех же параметрах (число нейронов, их характеристика) является уникальной. Уникальность сетей проявляется в том, что сети с одинаковыми параметрами, обучающиеся на одинаковых задачах, ведут себя неодинаково. Это касается времени обучения, качества обучения, уверенности в выдаваемых ответах при тестировании. В этом еще одно сходство сетей с биообъектами.

Рассмотрим работу сети безотносительно к процессу обучения или выдачи решения. После инициализации сети веса всех синапсов случайны. В начальный

момент времени на входные нейроны через входные синапсы (один или несколько) подается из внешнего мира вектор входных сигналов, представляющий набор чисел (или одно число). Далее этот сигнал начинает распространяться по всем связям между нейронами, изменяясь при прохождении через каждый нейрон согласно схеме функционирования нейрона. В конечном итоге, после одного прохода выходные нейроны выдадут во внешний мир какие-либо сигналы. Вся процедура однократного прохождения сигналов по нейронной сети является тактом функционирования сети. Можно не считывать сигналы с выходных нейронов после одного такта функционирования, а продолжить обмен сигналами еще несколько раз, не подавая сигналов на вход. Обычно количество тактов функционирования между подачей сигналов на вход и снятием сигналов с выхода фиксировано для данной сети. Как правило, этот параметр задается при инициализации сети и хранится в файле сети вместе с матрицей коэффициентов (как и некоторые другие параметры - число нейронов, характеристика нейронов и т.д.).

Таким образом, нейронная сеть, получающая на входе некоторый сигнал, способна после прохода его по нейронам выдавать на выходе определенный ответ, который зависит от весовых коэффициентов всех нейронов и от самого сигнала. Очевидно, что при осуществлении таких процедур на только что инициализированной сети мы будем получать на выходе сигналы, лишенные всякого смысла (весовые коэффициенты случайны). Чтобы добиться выдачи сетью требуемого результата, необходимо обучить ее.

Общая схема обучения нейронной сети

Для обучения нейронной сети необходима обучающая выборка (задачник), состоящая из примеров. Каждый пример представляет собой задачу одного и того же типа с индивидуальным набором условий (входных параметров) и заранее известным ответом. Например, в качестве входных параметров в одном примере могут использоваться данные обследования одного больного, тогда заранее известным ответом в этом примере может быть диагноз. Несколько примеров с разными ответами образуют задачник. Задачник располагается в базе данных, каждая запись которой является примером.

Не останавливаясь на математических алгоритмах, подробно описанных в монографии [45], рассмотрим общую схему обучения нейросети.

1. Из обучающей выборки берется текущий пример (изначально, первый) и его входные параметры (представляющие в совокупности вектор входных сигналов) подаются его на входные синапсы обучаемой нейросети. Обычно каждый входной параметр примера подается на один соответствующий входной синапс.
2. Нейросеть производит заданное количество тактов функционирования, при этом вектор входных сигналов распространяется по связям между нейронами (прямое функционирование).
3. Измеряются сигналы, выданные теми нейронами, которые считаются выходными.
4. Производится интерпретация выданных сигналов, и вычисляется оценка, характеризующая различие между выданным сетью ответом и требуемым ответом, имеющимся в примере. Оценка вычисляется с помощью соответствующей функции оценки. Чем меньше оценка, тем лучше распознан пример, тем ближе выданный сетью ответ к требуемому. Оценка, равная нулю, означает что требуемое соответствие вычисленного и известного ответов достигнуто. Заметим, что только что инициализированная (необученная) нейросеть может выдать правильный ответ только совершенно случайно.
5. Если оценка примера равна нулю, ничего не предпринимается. В противном случае на основании оценки вычисляются поправочные коэффициенты для каждого синаптического веса матрицы связей, после чего производится подстройка синаптических весов (обратное функционирование). В коррекции весов синапсов и заключается обучение.
6. Осуществляется переход к следующему примеру задачника и вышеперечисленные операции повторяются. Проход по всем примерам обучающей выборки с первого по последний считается одним циклом обучения.

При прохождении цикла каждый пример имеет свою оценку. Вычисляется, кроме того, суммарная оценка множества всех примеров обучающей выборки.

Если после прохождения нескольких циклов она равна нулю, обучение считается законченным, в противном случае циклы повторяются.

Количество циклов обучения, также как и время, требующееся для полного обучения, зависят от многих факторов - величины обучающей выборки, количества входных параметров, вида задачи, типа и параметров нейросети и даже от случайного расклада весов синапсов при инициализации сети.

Методологические аспекты обучения нейросетей

Иногда (при решении медико-биологических задач - крайне редко) встречаются ситуации, когда сеть не может обучаться. Это происходит в том случае, когда на определенном этапе обучения исчерпываются дальнейшие возможности поиска закономерностей между обучающими параметрами и результатами. Простейшая ситуация - когда два примера с совершенно одинаковыми наборами параметров подаются сети как принадлежащие различным классам (в классификаторах) или имеющие различное значение ответа (в предикторах). Очевидно, оба этих примера всегда будут попадать в одну и ту же точку в пространстве, их невозможно будет отделить друг от друга, и процесс обучения остановится. Программа, управляющая нейросетями, сигнализирует об окончании процесса обучения, причем указывает, что дальнейшее обучение невозможно. Задача специалиста, обучающего нейросети - избежать таких ситуаций, для чего нужны четкая постановка задачи и тщательный контроль обучающей выборки.

Обученная нейросеть автоматически записывается на диск компьютера как обыкновенный файл и может храниться там, сколько необходимо. В любой момент времени можно считать сеть с диска и продолжить обучение решению данной задачи со старой или новой обучающей выборкой. Одна нейросеть обучается решать только одну задачу классификации или предикции, однако может использовать для обучения различные обучающие выборки. Они могут различаться по количеству примеров, но должны соответствовать друг другу по числу обучающих параметров, числу классов (в классификационной задаче), а главное, по смыслу.

Говоря об обучении нейросетей, следует рассмотреть еще один важный аспект этой темы. Мы уже знаем, что успех обучения во многом зависит от числа

нейронов в сети, или, точнее, от числа синапсов. Именно весовые коэффициенты синапсов хранят "опыт" сети. Теоретически, бесконечно увеличивая число нейронов и синапсов, всегда можно добиться полного обучения сети на данном задачнике, однако это ли является целью создателя экспертной системы? Очевидно, нет. Главное, чтобы обученная сеть хорошо распознавала примеры, как раз не входящие в задачник.

Проблема заключается в том, что сеть с заведомо большим (избыточным) числом синапсов (относительно данного задачника) может хорошо обучиться, просто "механически запомнив" имеющиеся примеры. Такая сеть обучится быстро (нет необходимости как можно более точной подстройки весов) за счет количества, а не качества.

Хорошим практическим выходом из данной затруднительной ситуации были бы сети, способные автоматически наращивать число нейронов при невозможности дальнейшего обучения, не теряя при этом уже имеющегося опыта. Последнее условие вызывает значительные трудности. Нейросеть представляет собой единое целое, и добавление нового нейрона к сети, работающей в рамках имеющейся сейчас концепции, приведет к необходимости полностью переучивать сеть. Это требует обращения к первоначальному задачнику, что во многих случаях неприемлемо.

Поэтому создателю самообучающихся систем приходится идти на компромисс: либо делать сеть с некоторым избытком нейронов, имеющую резерв для накопления опыта, но обладающую относительно низкой способностью к экстраполяции, либо обучить сеть с небольшим числом нейронов, которая вряд ли сможет набрать потом дополнительный опыт. Все это, конечно, зависит еще и от задачника - насколько тесные взаимосвязи имеются между обучающими параметрами и известными ответами примеров. Чем больше таких взаимосвязей, тем меньше необходимость в "механическом запоминании" примеров. Практика показывает, что большинство биологических и медицинских задач имеют достаточно хорошие взаимосвязи, конечно, при грамотной постановке задачи и выборе обучающих параметров. Однако в рамках предлагаемой методологии, с учетом высокой скорости обучения нейросетей разработаны стратегия и тактика

обучения, позволяющие обойти вышеуказанный компромисс, за счет, правда, большего времени, необходимого для обучения.

Тестирование примеров

При тестировании примеров необходимость в системе подстройки весов синапсов отпадает, поэтому при создании экспертных систем блок программы, содержащий алгоритмы обучения, может не включаться в программу в случае, если не предполагается доучивать сети в процессе работы экспертной системы. Тестирование примеров нейросетью может проводиться с различными целями:

1. Проверка того, как обучилась нейросеть;
2. Решение конкретных задач.
3. Моделирование.

В первом случае осуществляется тестирование выборки с заранее известными ответами примеров. Таким образом можно проверить, правильно ли сеть определяет ответы примеров и насколько уверенно она это делает. Определенный сетью ответ примера сравнивается с заранее известным. Как правило, сначала тестирование проводится на той выборке, на которой сеть обучалась. Если сеть обучилась полностью, то при тестировании той же самой обучающей выборки ответы всех примеров будут определяться правильно. Гораздо больший интерес представляет тестирование аналогичной выборки с заранее известными ответами, но примеры которой не участвовали в обучении сети. Неправильное определение ответов некоторых примеров может быть вызвано несколькими причинами:

1. Выборка, по которой обучалась нейросеть, недостаточно полно отражает картину соответствия ответов обучающим параметрам, иначе говоря, в обучающей выборке слишком мало примеров.
2. Выборка, по которой обучалась нейросеть, составлена тенденциозно. Это означает, что для обучения подбирались примеры, которые, по мнению исследователя, "являются самыми яркими представителями своего класса или группы". Это серьезная ошибка. При такой выборке, конечно, нейросеть будет обучаться намного лучше, но способность ее к тестированию других примеров

существенно падает. В действительности необходимо обучать сеть на реальных данных, какими бы они противоречивыми ни были. Если сеть не сможет обучиться полностью, можно применить некоторые меры, которые будут рассмотрены ниже.

3. Обучающая выборка имеет недостаточное количество обучающих параметров и сеть не может найти закономерности между входными сигналами и ответами.
4. При создании сети не оптимально были выбраны некоторые сетевые параметры, например, число нейронов, число тактов функционирования или характеристика сети. Ниже мы подробно остановимся на методологии оптимального выбора этих значений.
5. Задана неверная классификационная модель (при обучении нейросетей-классификаторов). Возможно, на самом деле примеры группируются в 3 класса, а пользователь задает только 2. Далее будет рассмотрен метод коррекции классификационной модели.

При решении конкретных задач сети подаются примеры, ответ которых неизвестен. В этой ситуации программа не может проверить правильность решения.

В классификационных задачах при ответе нейросеть не только выдает результат - класс тестируемого примера. Как уже говорилось, в отличие от большинства экспертных систем, работающих по четким правилам, решение задачи на основе опыта всегда имеет “нечеткий” характер. Поэтому кроме класса тестируемого примера сеть вычисляет коэффициент уверенности в данном решении. Коэффициент уверенности зависит от заданного уровня надежности и рассчитывается по формуле:

$$КУ = (Max1-Max2) / R \times 100\%, \qquad (2)$$

где Max1 - ответ выходного нейрона, отвечающего за класс-"победитель", Max2 - ответ выходного нейрона, выдавшего следующий по максимальной величине сигнал, R - уровень надежности. Судя по формуле ясно, что уверенность сети зависит от того, насколько наибольший из выходных сигналов превышает второй по величине сигнал. Естественно, если КУ получается более 100%, он при-

равнивается к этому числу. Из того, что в знаменателе правой части формулы стоит уровень надежности, следует, на первый взгляд, парадоксальный вывод: сеть, обученная лучше (уровень надежности больше) даст меньшую уверенность в ответе, чем сеть, обученная хуже. Однако при внимательном рассмотрении проблемы делается ясно, что при тестировании примера первой сетью, выражение, стоящее в числителе, также будет больше (ответ одного нейрона будет намного больше ответа другого) вследствие лучшей обученности. Кроме того, КУ выражает все же уверенность конкретной сети, которая во многом зависит от того, насколько тестируемый пример близок к примерам, на которых обучалась эта сеть.

Если пример отличается достаточно сильно, лучше обученная сеть будет и сомневаться больше, чем сеть с меньшим "опытом".

Один из показателей качества обучения - определение прогностической способности нейросети - состоит в подсчете процента правильно распознанных примеров. При сравнении качества обучения двух нейросетей, в случае, когда обе сети дают одинаковую прогностическую способность, можно подсчитывать средний процент уверенности при тестировании выборки. Он рассчитывается как средняя арифметическая процентных величин уверенности, полученных при тестировании каждого примера за известный результат.

Иногда необходимо знать, к каким еще классам, кроме найденного, близок тестируемый пример. Это можно сделать несколькими способами, из которых достаточно трудно выбрать наилучший, однако самый оптимальный, на наш взгляд, заключается в том, чтобы просто сравнить сигналы, полученные со всех выходных нейронов (их можно выразить в процентах от максимально возможного). Понятно, что когда все выходные сигналы близки друг к другу, сеть затрудняется дать уверенный ответ.

Из вышесказанного вытекает очень полезный для практики вывод. Изменяя в различных направлениях значения параметров примера и повторяя его тестирование, можно видеть, что и на сколько нужно изменить, чтобы пример стал принадлежать к требуемому классу. Это может быть полезным для медицинской диагностики и прогнозирования. Предположим, что сеть обучена дифференци-

ровать больных и здоровых людей по набору клинических параметров. Изменяя на компьютере эти параметры, можно добиться, чтобы пример, определяемый как "больной", стал принадлежать классу "здоровый". Таким образом, станет ясно, какие клинические параметры подлежат изменению для улучшения состояния больного.

Создание медицинских нейросетевых экспертных систем

Любая экспертная система должна состоять условно из четырех блоков: интерфейс с пользователем, база знаний, вычислительный блок, блок объяснений, позволяющий пользователю проследить "ход рассуждений" системы в конкретном случае. Связующим элементом между этими блоками является метод, с помощью которого экспертная система в ответ на запрос пользователя выдает результат (заключение). Такие методы можно разделить на три основные группы:

1. Методы логических правил “в чистом виде”, когда формализация правил получения результата осуществляется специалистом;
2. Те же методы, однако формализация правил осуществляется исследователем, наблюдающим за работой специалиста со стороны;
3. Методы, основанные на принципе “смотри и учись”.

Создание даже простых экспертных систем, основанных на методах 1 и 2, представляет собой нелегкую задачу, прежде всего потому, что требует совместной работы специалистов различного профиля. Традиционные экспертные системы, основанные на базах знаний и логических правилах, требуют для создания довольно большого времени и средств. Создание традиционной экспертной системы можно условно разделить на несколько этапов.

1. Постановка задачи: определение целей работы экспертной системы, набора входных данных и формы представления ответа.
2. Сбор данных: набор репрезентативного материала для статистических исследований и его структурирование - разделение на подгруппы по разнообразным признакам.

3. Статистическая обработка: выявление закономерностей, связывающих входные данные с ответом - расчет средних и относительных величин, их сравнение, корреляционный, регрессионный, факторный анализы и т.д.
4. Создание базы знаний: оформление логических правил, по которым должна работать экспертная система.
5. Программирование алгоритмов: перенесение логических правил на язык программирования.
6. Создание интерфейса системы: разработка средств взаимодействия системы с пользователем - формы ввода данных, вывода ответа и т.п.
7. Отладка и тестирование: проверка работы программы и испытание в реальных условиях.

При создании логических экспертных систем наибольшую часть времени занимают 3, 4 и 5 этапы, требующие совместной работы как предметных специалистов, так и программистов и математиков. Несмотря на появление компьютерных средств проектирования экспертных систем, основная работа все равно возложена на специалистов. При этом возникают сразу несколько серьезных проблем.

Первая из них состоит в том, что при решении сложных реальных задач (экономика, проектирование, инженерия, биология) число логических правил значительно увеличивается. Часто возникает настолько сложная система взаимосвязей между ними, что ее просто не удается осмыслить. Разбивка задачи на блоки также не всегда помогает: во-первых, это тоже не всегда просто сделать, во-вторых, при разбивке иногда могут теряться некоторые взаимосвязи.

Вторая, еще более серьезная проблема состоит в том, что далеко не всегда удается выразить вычислительный процесс логическими правилами. Это может быть связано как со сложностью самой задачи, так и с особенностями деятельности предметного специалиста. Особенно ярко это проявляется в медицине, где процесс принятия решения во многом опирается на интуицию и опыт врача, не являющегося экспертом в области собственного мышления. Во всех этих случаях говорят, что задача не поддается алгоритмированию. Кроме того, даже если создателям удается разработать алгоритм, никогда нет достаточной гарантии, что он

будет корректно работать в реальных условиях, а это можно проверить только после окончания всех работ по созданию системы.

В создании самообучающейся системы также можно выделить несколько этапов, часть из которых совпадает с этапами создания традиционных систем.

1. Постановка задачи. То же, что и для традиционных систем плюс выбор оптимальной структуры нейронной сети и методов обучения (для большинства задач структура и методы стандартны).
2. Сбор обучающих данных. Набор примеров для обучения сети, каждый из которых представляет массив входных данных и соответствующий ему заранее известный ответ.
3. Создание и обучение нейросети. Данный этап не требует проведения статистических вычислений, а если задача укладывается в стандартную схему (в большинстве случаев), то и программистской работы. Если задача нестандартная, требуется адаптация структуры нейросети и метода вычисления оценки при обучении. Обучение нейросети в большинстве стандартных случаев представляет собой автоматический процесс, который только после его окончания требует участия специалиста для оценки результатов. Естественно, часто может требоваться корректировка, создание дополнительных сетей с другими параметрами и т.д., однако всегда есть возможность оценить работу системы на любом этапе обучения, протестировав контрольную выборку. Разрабатывая методологию создания нейросетевых экспертных систем, мы исходили из возможности разработки наиболее индивидуализированных (рассчитанных на одного конкретного пользователя-специалиста) систем самим этим специалистом. Конечно, ничто не мешает объединять в одной системе индивидуальный опыт нескольких специалистов. Отсутствие "математических" этапов реализует такие возможности. Предметный специалист в состоянии самостоятельно поставить задачу, более того, никто, кроме него, не сможет сделать это лучше. Сбор материала также должен осуществлять предметный специалист. Схемы постановки задач, способы представления данных и способы продукции ответа нейросетью разработаны таким образом, что большинство задач во многих областях укладываются в эти стандартные схемы. Поэтому при наличии хо-

рошо продуманных инструментальных программных средств работы с нейронными сетями и документации к ним большинство специалистов способны самостоятельно разрабатывать не очень сложные нейросетевые приложения.

4. Создание интерфейса. То же, что и для традиционных экспертных систем.
5. Отладка и тестирование. Этап включает в основном отладку работы программы, т.к. тестирование часто проводится в процессе обучения сетей.
6. Доучивание. Этап, характерный только для обучающихся систем. При создании нейроэкспертных программ довольно редко возможно сразу собрать достаточное количество данных для хорошего обучения сети. Поэтому, создавая нейросистему, исследователи определяют наилучшие параметры сетей и проводят стартовое обучение. В последующем пользователи доучивают систему в условиях реальной работы и реальных данных, передавая ей опыт. Более того, коренное отличие методологии создания нейросетевых систем от традиционных состоит именно в том, что система никогда не создается сразу готовой, и более того, никогда не является полностью законченной, продолжая накапливать опыт в процессе эксплуатации.

Резюмируем имеющиеся преимущества нейросетевых экспертных систем перед обычными, которые, как уже говорилось, проявляются только при решении трудноалгоритмируемых задач.

1. Нейросети принимают решения на основе опыта, приобретаемого ими самостоятельно. "Самостоятельно" в данном случае означает то, что создателю экспертной системы не требуется устанавливать взаимосвязи между входными данными и необходимым решением, затрачивая время на разнообразную статобработку, подбор математического аппарата, создание и проверку математических моделей.
2. Решение, принимаемое нейросетью, не является категоричным. Сеть выдает решение вместе со степенью уверенности в нем, что оставляет пользователю возможность критически оценивать ее ответ.
3. Нейросеть позволяет моделировать ситуацию принятия решения.

4. Нейросети дают ответ очень быстро (доли секунды), что позволяет использовать их в различных динамических системах, требующих незамедлительного принятия решения.
5. Возможности нейросетей (коррекция классификационной модели, минимизация обучающих параметров и др.) позволяют упрощать процесс создания экспертных систем, определять направления научного поиска.

Главным критерием работы нейросетевых экспертных систем должна быть практика - многократные испытания и проверки в самых различных условиях.

Определенным препятствием использования нейросетей может является все же некоторая ограниченность задач, решаемых ими. Иногда в блоке трудноалгоритмируемых задач, решаемых с помощью самообучающейся экспертной системы, могут присутствовать элементы четких правил. В таком случае совершенно логично комбинировать в одной экспертной системе несколько нейросетей или даже обычные математические методы и строить из них иерархические блоки, одни из которых используют для своих действий результаты работы других. Следует подчеркнуть, что применение неявных алгоритмов не противоречит и не отменяет использование формальных методов, а может дополняться ими при необходимости. Например, если с помощью нейросети определяется оптимальная комбинация лекарственных препаратов для лечения пациента, и имеется совершенно четкое противопоказание к назначению определенного препарата, в экспертную систему может быть введен простой логический блок, препятствующий назначению этого лекарства независимо от решения нейросетей.

Ниже мы рассмотрим принципы и особенности создания нейросетевых экспертных систем для биологии и медицины.

Постановка задачи

Постановка задачи всегда является прерогативой предметных специалистов.

Прежде всего, необходимо определить, что представляет собой акт работы экспертной системы, начинающийся со ввода данных (условия задачи) пользователем и заканчивающийся выдачей ответа. Например, в медицинских диагностических задачах акт работы системы чаще всего начинается с введения в ЭВМ

данных о пациенте (симптоматика, анамнез, результаты анализов и т.д.) и заканчивается выдачей возможного диагноза и/или прогноза. Часто одна задача компонуется из нескольких подзадач. В этом случае каждой подзадаче может соответствовать отдельный акт работы системы, хотя для пользователя это может быть совершенно незаметным. Например, по данным о пациенте требуется установить диагноз и назначить соответствующее лечение. Установление диагноза на базе введенных параметров - первая подзадача, назначение терапии - вторая, при этом для назначения терапии нужны не только исходные данные, но и результат решения предыдущей задачи - диагноз.

Полезно сразу же для каждой подзадачи определить ее тип (классификация, предикция, векторная предикция).

Обычно каждая подзадача решается одной нейросетью или несколькими нейросетями, объединенными в один функциональный блок (малые эксперты [46]). При этом введенные данные подаются последовательно на каждую нейросеть блока и каждая нейросеть выдает ответ. Ответы могут различаться, поэтому в такой ситуации требуется разработать способ получения единственного ответа. Это можно сделать двумя способами:

1. Путем логических правил. Например, если 5 нейросетей выдали ответ "здоров", а 2 - "болен", то общее решение - "здоров", т.к. за него проголосовало большее число нейросетей-экспертов. Если не удается формализовать правило, можно принимать решение на основании степеней уверенности каждой из сетей.
2. Путем надстройки над блоком малых экспертов нейросети-”супервизора”, которая обучена принимать решение по результатам работы этих малых экспертов.

После определения акта работы системы и разбивки (если требуется) задачи на подзадачи следует разработка схемы обучающих примеров для каждой из подзадач. Схема примера включает список входных и выходных параметров для данной подзадачи.

Определение списка входных данных - квалифицированная работа предметного специалиста, требующая знания изучаемой области и ориентировочной

важности тех или иных параметров, необходимых для получения ответа. Желательно в начале работы над проектом задать некоторую избыточность списка входных данных. В дальнейшем “лишние” параметры можно будет легко исключить из работы системы, добавить же новые несколько труднее, прежде всего потому, что потребуется вновь обращаться за этими параметрами к источникам данных.

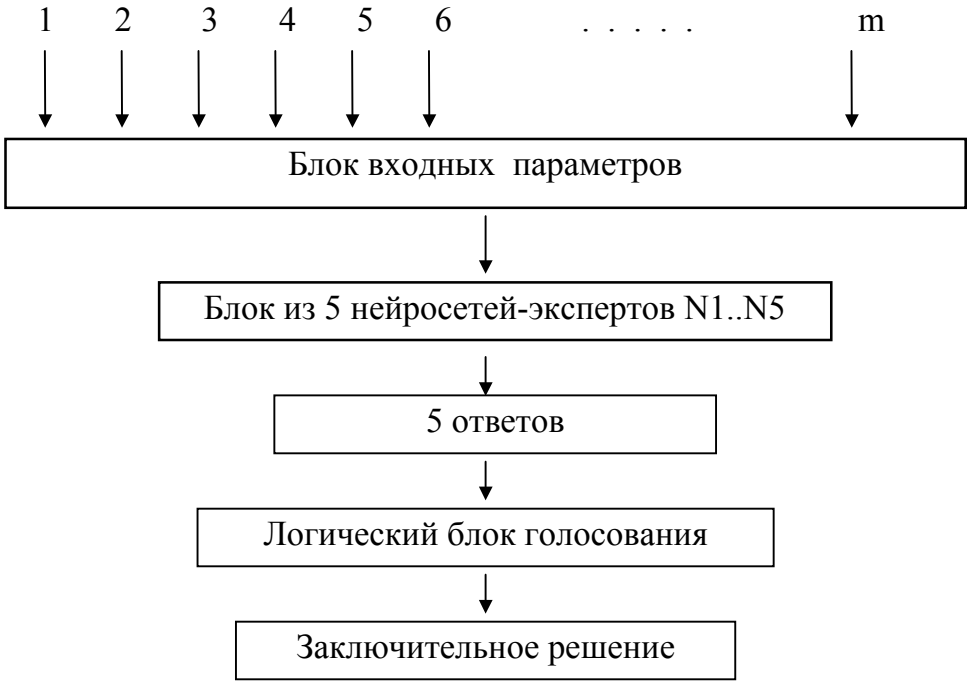


Рисунок 5. Схема движения информации для экспертной системы с 5 нейросетями-экспертами, решающими одну и ту же задачу и логическим блоком, осуществляющим голосование для выдачи заключительного решения

Далее желательно составить наглядную схему движения информации в экспертной системе начиная со ввода данных из внешнего мира и заканчивая выводом ответа (ответов) пользователю. Пример такой схемы приведен на рисунке 5.

Необходимо определить ориентировочные размеры обучающих выборок. Так, в классификационных задачах (подзадачах) желательно, чтобы каждый класс ответа был представлен достаточным количеством примеров. Необяза-

тельно, но желательно, чтобы количества примеров каждого класса не различались слишком сильно (более, чем на порядок).

Предварительное обучение и тестирование нейросетей можно проводить на небольшом количестве примеров с целью определения дальнейшей стратегии и тактики обучения, а также уточнения общего размера обучающей выборки.

Представление обучающих данных

Следующий этап работы над проектом - разработка представления обучающих данных. Нейросети оперируют с информацией, представленной только в виде чисел. Числа подаются на входные синапсы нейросети; ответы, снимаемые с выходных нейронов, также представляют собой числа, поэтому для оценки примера заранее известный ответ также должен быть представлен в виде числа (чисел). Информация же, на основании которой нейросеть должна давать ответ, может быть самого разнообразного вида: термины, описывающие какие-либо ситуации, числа различного вида и величины, динамические ряды, массивы, графики, динамические кривые, двух- и трехмерные изображения и т.д. Поэтому возникает необходимость корректного представления этой информации в виде чисел, сохраняющих смысл и внутренние взаимосвязи в данных.

Существует огромное количество способов представления информации для различных целей.

Для работы нейронных сетей с медико-биологическими данными мы предлагаем по возможности полную классификацию данных, с которыми может столкнуться создатель медицинских экспертных систем, и оптимальные способы их представления в численном виде, учитывающие специфику работы нейросетевых систем. Однако это не готовые рецепты, поэтому в каждом конкретном случае все же требуется квалифицированное решение специалиста.

1. Число с "плавающей точкой". Один из самых распространенных типов данных. Данные такого типа могут принимать любые значения, дробные или целые. Чаще всего они положительны, но могут быть и отрицательными. Немаловажно, что для работы нейронных сетей практически не имеет значение, подчиняется или нет вариационный ряд этих данных закону нормального распределения. Как правило, данные такого вида располагаются на каком-либо интервале с

нечеткими границами. Примером может послужить большинство данных лабораторных анализов. Данные в виде чисел с "плавающей точкой" не требуют каких-либо перерасчетов и могут использоваться в готовом виде. Следует особо отметить, что способы кодирования таких показателей путем разбивки на интервалы и присвоения каждому интервалу порядкового номера нежелательны для нейронных сетей.

2. Взаимоисключающие варианты. Один из наиболее сложных типов данных, требующих продуманного представления. Информация при этом представлена в виде одного и только одного варианта из заранее известного и ограниченного набора вариантов и не может принимать вид дробного числа. Простейшим примеров может служить пол человека - мужской или женский. Такая информация требует численного кодирования. В приведенном примере можно закодировать мужской пол как 1, женский - как 2 или наоборот. Однако далеко не всегда можно сделать такое простое и произвольное кодирование. Для обучения нейросетей подобную информацию логично подразделять на 3 основных подтипа.

А. Неупорядоченные варианты - приведенный пример с полом человека. Их можно кодировать произвольным способом. Часто к этому типу относятся данные, представляемые всего двумя вариантами (да - нет, согласен - не согласен, болел - не болел и т.д.).

Б. Упорядоченные варианты. Такие данные находятся в определенных взаимосвязях друг с другом, которые могут быть выражены отношениями типа "больше", "меньше". Примером может служить степень тяжести заболевания (I,II,III). В любом случае варианты располагаются в определенном порядке - как правило, возрастания или убывания.

В. Частично упорядоченные варианты. Способ упорядочивания не очевиден, однако его можно найти, если это необходимо для решения задачи.

3. Совместимые варианты. Информация может быть представлена одним или одновременно несколькими вариантами из известного и ограниченного набора вариантов. Примером может являться наличие у обследуемого каких-либо заболеваний или заболеваний, перенесенных в детстве. В таких случаях имеется два различных подхода к кодированию данных.

А. Если варианты неупорядоченные, наилучший (но, к сожалению, не всегда удобный) способ состоит в том, чтобы разбить признак на несколько признаков, количество которых равно количеству вариантов и каждый из них кодировать отдельно. Каждый подпризнак в таком случае делается самостоятельным признаком и подается на отдельный входной синапс нейросети.

Б. Если варианты упорядоченные, можно применить принцип битовой маски.

4. Дата, время. Очень часто медицинские данные содержат даты различных событий в жизни обследуемых. Для численного представления временных точек необходимо в каждом признаке выбирать соответствующую точку отсчета и, отталкиваясь от нее, выражать временной интервал в удобных единицах (секунды, часы, сутки, годы). Например, если у больного указана дата возникновения какого-либо заболевания, удобнее перевести ее в возраст, который соответствовал этому событию. Сделав это для всех обследуемых, специалист приведет показатель к единой шкале. Если требуется другой подход, можно посчитать, сколько времени прошло с момента возникновения заболевания до момента настоящего обследования. В каждом случае предметный специалист должен выбрать способ представления, наиболее хорошо отражающий смысл параметра.

5. Графики. Разработка способа представления графиков часто требует довольно большого времени.

А. Наиболее простой, но не всегда возможный способ - дискретное разложение графика. Однако для подачи такой информации потребуется большее количество входных синапсов нейросети.

Б. Можно измерять на графике специально выбранные величины, отражающие наиболее важные характеристики явления. Например, для ЭКГ - это ширина и высота различных зубцов, наличие отрицательных зубцов (да, нет) и т.д. Такой способ может оказаться и экономнее, и эффективнее, однако требует хорошего знания изучаемого явления. Во многих случаях приходится прибегать к совмещению обоих методов, особенно в экспериментальных работах при изучении новых явлений с помощью нейросетей.

6. Произвольные изображения. Для подачи изображения на нейросеть оно представляется в виде большого массива чисел (иногда состоящего из сотен тысяч элементов). Каждое число при этом отражает соответствующую точку изображения, разделенного на маленькие фрагменты. Чем больше фрагментов, тем точнее представление изображения, но тем больший массив для этого требуется. Стандартное графическое изображение компьютеров IBM AT 286 имеет размер 640*350 точек и может быть представлено массивом из 224000 чисел. Кодировка каждой точки может быть различной. Если изображение монохромное и не допускает полутонов, каждая точка может быть отображена значениями 0 (белая) или 1 (черная) или наоборот. Для хранения такого значения достаточно 1 бита информации, поэтому для представления всего изображения потребуется 28000 байт, что вполне приемлемо. Если же изображение цветное или имеет полутона, то каждая точка должна выражаться числом, разрядность которого достаточна для представления любого цвета или полутона.

После разработки списка обучающих параметров для каждой подзадачи и определения способа представления каждого параметра и ответа можно приступать к формированию обучающих выборок - сбора информации и помещения ее в базы данных.

Инициализация нейронной сети

После формирования обучающей выборки производится инициализация нейронной сети (сетей), для которой определяются стартовые параметры. Данная подглава посвящена выбору стартовых параметров нейросети в зависимости от обучающей выборки и цели обучения.

Разработанная нами методика создания нейросетевых экспертных систем включает фиксированный набор стартовых параметров, выбор которых может осуществляться предметным специалистом. Однако даже этот небольшой набор может быть минимизирован путем автоматизации установки достаточно очевидных параметров.

Список стартовых параметров включает следующие пункты (параметры, выбор которых может быть автоматизирован, помечены звездочками):

1*. Выбор типа нейронной сети;

2.* Имя файла нейронной сети;

3. Схема подачи обучающих данных, которая определяет количество входных сигналов и соответствующих им входных синапсов, а также ответ;

4*. Включение или выключение нормирования входных сигналов;

5*. Количество нейронов;

6*. Параметр плотности (количество нейронов, на которые подается один входной сигнал);

7*. Время отклика нейросети (число тактов функционирования нейросети при прохождении одного примера с момента подачи входных сигналов до выдачи ответа);

8*. Величина характеристики нейронов.

Как видно, в минимизированном варианте списка параметров специалист-пользователь должен всего лишь поставить задачу, не задумываясь о выборе остальных параметров.

Рассмотрим методику задания каждого параметра, а также вариант, который принимается при автоматической установке.

Следует заметить, что в рамках принятой нами идеологии создание нейросетей должно осуществляться при наличии стартовой обучающей выборки, находящейся в базе данных, что значительно облегчает выбор параметров пользователем и дает возможность автоматизации выбора параметров.

1. Выбор типа нейронной сети. Определяется типом задачи (классификация, предикция, векторная предикция). Перед выбором необходимо убедиться в соответствии обучающей выборки типу нейросети. При автоматическом выборе программа, после определения пользователем набора обучающих параметров и выбора поля базы данных, в котором содержится ответ, осуществляет анализ этого поля. Если данные в нем представлены целыми числами небольшой размерности или фиксированным набором чисел с “плавающей точкой”, устанавливается классификационный тип нейронной сети. Если же данные представлены числами с “плавающей точкой” с достаточно широким варьированием, устанавливается предикционный тип нейросети.

2. Имя файла нейронной сети. При автоматической установке имя файла связывается с установленной задачей.

3. Схема подачи обучающих данных и ответа не может быть автоматизирована, так как изначальный набор обучающих параметров и параметры, содержащие ответ, могут быть заданы только самим специалистом. При наличии базы данных пользователь указывает, какие поля базы рассматриваются как обучающие, и в каких полях содержатся ответы. Каждая запись базы является примером, если ячейка в поле, содержащем ответ, не пуста, и если пользователь специально не указал, что данная запись не должна участвовать в обучении. Каждый входной (обучающий) параметр можно при обучении и тестировании подавать на один (стандартно) или несколько (при необходимости) входных синапсов нейросети. Второй вариант можно использовать в тех случаях, если пользователь заранее считает, что какой-либо параметр должен играть особое значение для получение ответа и его влияние необходимо усилить. Для примера рассмотрим две схемы. Первая (Таб. 1) задается программой по умолчанию (стандартно) и не меняется, если создателю нейросети нет необходимости переопределять форму подачи данных. Вторая схема (Таб. 2) является результатом направленных действий создателя нейросети.

Таблица 1.

Схема подачи обучающих параметров и определение ответа, устанавливаемые стандартно

Поля базы данных	Количество входных синапсов, на которые подается обучающий параметр	Номера входных синапсов, на которые подается обучающий параметр
1. Диагноз	1	- (Ответ)
2. Возраст	1	1
3. Пол	1	2
4. Количество эритроцитов	1	3
5. Количество лейкоцитов	1	4
6. Гемоглобин	1	5

В первой схеме база данных содержит 6 полей, содержащих данные об обследуемых. Первое поле содержит ответ (диагноз), остальные поля являются обучающими. Если поле, содержащее ответ, первое в базе данных, не требуется переопределять ответ. В приведенном примере 5 обучающих полей, данные каждого из них будут подаваться на один соответствующий входной синапс.

Во втором примере поле, содержащее ответ, стоит последним; кроме того, имеются поля, содержащие код обследуемого и его фамилию. В этой ситуации пользователь должен указать, что ответом является именно поле “Диагноз”.

Подавать на вход нейросети код обследуемого не имеет никакого смысла, поэтому данное поле выключается из схемы подачи входных сигналов, несмотря на то, что оно может иметь числовой тип. Поле, содержащее фамилию обследуемого, также выключается из схемы, т.к. оно является не числовым, а символьным. Поля, содержащее данные о количестве эритроцитов и гемоглобине, подаются соответственно на 2 и 3 входных синапса создаваемой нейросети, что будет определять их относительно большее влияние на выдачу ответа у обученной се-

ти. Таким образом, несмотря на то, что число обучающих параметров равно по прежнему 5, число входных синапсов будет увеличено до 8.

Таблица 2.

Схема подачи обучающих параметров и определение ответа, заданные создателем нейросети

Поля базы данных	Количество входных синапсов, на которые подается обучающий параметр	Номера входных синапсов, на которые подается обучающий параметр
1. Код пациента	-	-
2. Ф.И.О. пациента	-	-
3. Возраст	1	1
4. Пол	1	2
5. Количество эритроцитов	2	3, 4
6. Количество лейкоцитов	1	5
7. Гемоглобин	3	6, 7, 8
8. Диагноз	-	- (Ответ)

Подачу некоторых входных параметров на несколько входных синапсов можно рекомендовать и в случаях, когда общее число обучающих параметров мало (единицы).

4. Включение или выключение нормирования входных сигналов. Нормирование входных сигналов представляет собой один из видов предобработки и является исключительно важным в методологии создания нейросетевых экспертных систем. При нормировании на входной синапс подается не величина параметра (для данного примера), а ее эквивалент, полученный путем пересчета по определенной схеме. Мы применяем нормирование входных сигналов на диапазон [-1...1]. Система нормирования имеет два аспекта - методологический и

технологический. Первый состоит в том, что нормирование позволяет унифицировать представление информации внутри “черного ящика” нейросетевой экспертной системы и поэтому отменяет необходимость создателю и/или пользователю экспертной системы контролировать диапазоны числовых значений медико-биологических данных. Технологический аспект заключается в следующем. Разработанная технология обучения нейронных сетей предусматривает универсальную структуру и алгоритмы обучения для медико-биологических данных любого характера. Однако в результате проводимых нами многочисленных экспериментов было установлено, что наиболее универсальная и быстрообучающаяся архитектура полносвязной сигмоидной (имеющей характеристическую функцию нейронов) нейросети оптимально работает при нахождении входных сигналов в диапазоне приблизительно [-1...1]. При расширении этого диапазона происходит сначала практически незаметное, плавное, а затем быстро нарастающее снижение качества обучения, особенно ярко проявляющееся при сильно различающихся диапазонах у разных входных сигналах.

Еще в большей степени нормирование влияет на экстраполяционные возможности нейросети. Исследования показали, что процент правильно распознанных примеров, не входящих в обучающую выборку, максимален также при нормировании входных данных на диапазон [-1...1].

Учитывая результаты экспериментов, разработанная методология предусматривает нормирование всех входных параметров на диапазон [-1...1]. Каждый входной сигнал перед подачей на синапс пересчитывается по формуле

$$Y_i = 2 * (x_i - min_i) / (max_i - min_i) - 1 \qquad (3),$$

где x - исходный сигнал, Y_i- получаемый нормированный сигнал, min и max - соответственно минимальное и максимальное значения интервала входных параметров в поле, подаваемом на синапс i.

Диапазон входных параметров рассчитывается для каждого поля, подаваемого на синапсы нейросети. Расчет происходит при создании нейросети. После создания нейросеть хранит минимум и максимум диапазона в своем файле. Расчет диапазонов происходит автоматически исходя из значений, заданных в примерах обучающей выборки. Поэтому желательно, чтобы обучающая выборка

содержала примеры с возможными крайними значениями всех обучающих параметров. Если впоследствии, например при дообучении или тестировании, какое-либо значение будет находится вне промежутка min..max, пересчитанное значение будет соответственно лежать за пределами диапазона [-1...1]. Если "выпадение" из интервала будет небольшим (десятые доли), это практически не повлияет на качество дообучения и тестирование.

Тем не менее, рекомендуется после создания нейросети (до начала всякого обучения!) внимательно просмотреть диапазоны значений, соответствующие каждому входному синапсу, и если в будущем предполагается подача значений вне данного диапазона, вручную изменить его. Ни в коем случае нельзя менять диапазоны после обучения нейросети, т.к. это приведет к последующей некорректной обработке входных сигналов, сбою при дообучении и некорректному тестированию. При автоматическом задании параметров сети всегда устанавливается нормирование, автоматический подсчет диапазонов значений и расширение их на заданную относительную величину. Отмена нормирования предусматривается в основном для проведения экспериментов и исследований, как возможность дополнительной настройки.

Кроме нормирования входных сигналов, устанавливается нормирование значений ответов у нейросетей-предикторов. Проведенные эксперименты показали безусловное преимущество такого подхода. Нормирование значений ответов проводится по схеме, аналогичной нормированию входных сигналов.

5. Количество нейронов. Этот параметр определяет суммарное количество синаптических связей нейронной сети. Для полносвязной нейросети оно равно квадрату числа нейронов. Если сеть использует адаптивную матрицу входных сигналов (в нашей методике это делается всегда), общее число подстраиваемых связей равно квадрату числа нейронов плюс произведение числа входных синапсов и плотности подачи входных сигналов (на сколько нейронов подается каждый входной сигнал).

Оптимальное количество нейронов во многом зависит от решаемой задачи. Большее число нейронов повышает гарантию успешного обучения, но увеличивает размер нейросети, а значит, время ее срабатывания при тесте и время ее

загрузки с диска компьютера в память. В общем случае, число нейронов может быть равно числу обучающих параметров. В классификаторах без адаптивной матрицы входных сигналов количество нейронов устанавливается автоматически и равно сумме числа обучающих параметров и количества классов в задаче. Если набор входных параметров избыточен (предполагается, что для обучения можно обойтись лишь частью этого набора) можно задавать число нейронов меньшим.

Не представляется возможным корректно исследовать зависимость качества обучения от числа нейронов, т.к. невозможно создать сети с одинаковыми стартовыми картами и разным числом нейронов. При сравнении сетей с различными картами на качество будет влиять (хотя и не сильно) индивидуальность сети, определяемая случайными стартовыми значениями синаптических весов.

При автоматическом задании числа нейронов предлагается следующая технология. Инициализируется и обучается нейросеть с небольшим количеством нейронов (например, 2). При невозможности обучения, в зависимости от суммарной оценки и количества входных синапсов рассчитывается новое, увеличенное количество нейронов. Инициализируется и обучается новая сеть, повторяющая параметры предыдущей, но имеющая заново установленное большее число нейронов. Процесс повторяется до тех пор, пока какая-либо из сетей не обучится полностью. Недостатком метода является увеличение времени, затрачиваемого на обучение. Преимущества - создателю экспертной системы нет необходимости экспериментировать с различным количеством нейронов, на что в конечном итоге может потребоваться гораздо больше времени. Кроме того, предлагаемый способ позволяет избежать создания заранее слишком избыточной сети, которая может быстро и хорошо обучиться на обучающей выборке, однако будет обладать низкой экстраполяционной способностью.

6. Параметр плотности. Плотность подачи входных сигналов определяет, на сколько нейронов будет одновременно подаваться сигнал с каждого входного синапса (не путать подачу некоторых, определяемых создателем сети, сигналов на несколько входных синапсов). При числе нейронов, равном количеству входных параметров, и плотности, равной единице, каждый входной сигнал подается на один соответствующий нейрон. Если в этом же случае плотность будет равна

числу нейронов, то каждый входной сигнал будет подаваться на все нейроны. Такое "размазывание" сигналов по нейронам обеспечивает адаптивная матрица входов. В общем случае, это всегда приводит к улучшению качества обучения, хотя и замедляет срабатывание сети и незначительно увеличивает ее размер. При автоматическом задании плотности она устанавливается равной числу нейронов.

7. Время отклика. Определяет число тактов функционирования сети с момента подачи входных сигналов до момента снятия выходных сигналов. Эксперименты показали, что в подавляющем числе медико-биологических задач следует задавать этот параметр равным 2 (редко 3). Это связано с относительно тесными взаимосвязями (часто приближающимися к линейным), присутствующими в медицинских данных. Большие значения времени отклика, как правило, приводят к замедлению обучения и снижению прогностической способности нейросети. Несколько большие значения времени отклика можно порекомендовать для сетей-предикторов.

Эксперименты показали, что увеличение времени отклика приводит также к увеличению общего числа тактов функционирования, необходимого для полного обучения сетей практически на всех медицинских задачах, обсуждаемых ниже. Кроме того, увеличение реального времени, требуемого на один такт работы нейросети при большом времени отклика, увеличивает еще и машинное время, требуемое для обучения. Поэтому при автоматическом задании времени отклика оно принимается равным 2.

8. Величина характеристики нейронов. Как уже говорилось, при прохождении сигнала через нейроны он пересчитывается по функции нейрона, имеющей заданный параметр характеристики. В результате величина сигнала всегда уменьшается, причем чем больше характеристика нейрона, тем существеннее уменьшение одного и того же сигнала (Рисунок 6).

Кроме того, разность двух поданных сигналов (уже нормированных) после их преобразования по характеристической функции будет различной в зависимости от характеристики и зон, в которых находятся поданные сигналы. Функция с малой характеристикой обеспечивает относительно большую разность между преобразованными сигналами, которые до преобразования находились или на

значительном расстоянии друг от друга, или группировались в зоне наиболее крутого участка функции (около нуля). Наоборот, функция с большой характеристикой лучше разделяет сигналы, попадающие в крайние зоны и находящиеся на небольшом расстоянии друг от друга. Так как довольно значительная часть медико-биологических параметров подчиняется закону нормального распределения или же носят дискретный характер, нейронные сети с малой характеристикой нейрона обучаются на этих данных гораздо быстрее, особенно, если в задаче, например, классификации, классы достаточно легко разделимы в пространстве входных параметров. Однако сети с большей характеристикой обладают значительно лучшими экстраполяционными способностями, хотя и хуже обучаются.

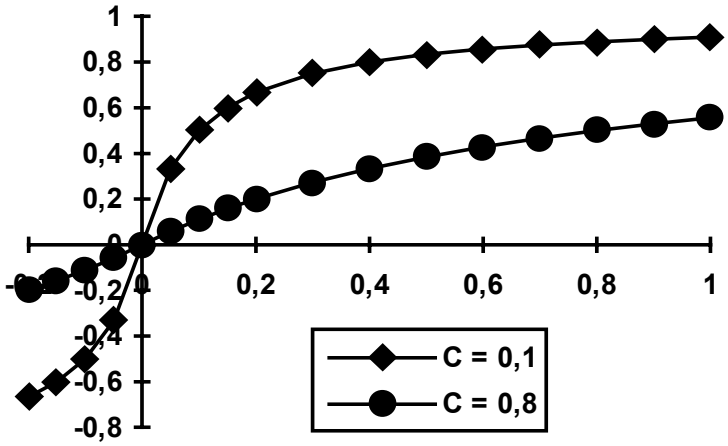


Рисунок 6. Схема изменения сигнала при прохождении через нейрон в зависимости от величины характеристики

Если характеристика устанавливается пользователем, рекомендуется вначале обучать сеть с небольшой характеристикой, а при успешном обучении генерировать и обучать новые сети, увеличивая этот параметр. Нельзя менять характеристику у обучающейся или обученной сети, т.к. это приведет к некорректной

обработке входных сигналов. Мы рекомендуем диапазон значений характеристики от 0,1 до 0,8.

При автоматическом задании этого параметра характеристика вначале устанавливается небольшой, а затем, в случае успешного обучения нейросети, постепенно повышается.

Стратегия и тактика обучения нейронных сетей

Под стратегией обучения нейросетей понимается общий план обучения, включающий разбивку задачи на подзадачи, определение типа и количества нейросетей, обучаемых по каждой из них, предварительные параметры сетей, планирование действий в случае возможных затруднений.

Тактика представляет собой возможное вмешательство оператора в процесс обучения с целью его ускорения, подстройки некоторых параметров, а также оптимальные действия при невозможности дальнейшего обучения сети.

Сформулируем цель обучения, на достижение которой направлены стратегия и тактика. Целью будем считать полное на данном задачнике обучение нейросети (набора нейросетей-экспертов) с минимально возможным числом подстраиваемых связей, максимально возможной характеристикой и максимально возможным уровнем надежности (минимальным уровнем отклонения). Дополнительным условием может быть минимальное (или определенное пользователем) число входных сигналов.

Критерием достижения цели будем считать результат тестирования набора примеров с известными ответами, не входящих в обучающую выборку. Требуемый результат определяется пользователем.

Рассмотрим, как влияют на качество и длительность обучения некоторые параметры. В таблицу 3 сведены зависимости между обучением (требуемым временем и качеством) сети, ее параметрами, особенностями задачи и используемого компьютера.

Таблица 3.

Зависимость времени и качества обучения нейросетей от их параметров, особенностей задачи и применяемого компьютера. Стрелки в двух правых колонках показывают, как изменяются длительность и качество обучения при изменении параметра, указанного слева

Параметр	Длительность обучения	Качество обучения
Увеличение тактовой частоты компьютера	↓	-
Вывод информации в процессе обучения на экран в графическом режиме (по сравнению с текстовым режимом)	↑	-
Увеличение количества примеров	↑	↑
Увеличение количества обучающих параметров	↑	↑
Увеличение количества классов (в классификаторах)	↑	↓
Увеличение размерности вектора ответа (в предикторах)	↑	↓
Близкое количество примеров каждого класса (в классификаторах)	↓	↑
Повышение уровня надежности (в классификаторах)	↑	↑
Повышение уровня отклонения (в предикторах)	↓	↓
Увеличение числа нейронов	↑	↓
Увеличение параметра плотности	↑	↑
Увеличение времени отклика сети	↑	↓
Увеличение характеристики	↑	↑

Разработанная методология включает две стратегии: с участием пользователя и без него. Автоматизированное выполнение стратегии может проводиться в двух вариантах - повторять принципы “ручной” стратегии, или использовать другие правила. Кроме того, стратегии автоматического обучения подразделяются еще на два класса - стратегии решения подзадачи и стратегии решения единой задачи, состоящей из взаимосвязанных (параллельно или иерархически) подзадач. Мы заранее отказались от “ручной” стратегии решения единой комплексной задачи, так как, во-первых, они встречаются не так уж часто, во-вторых, при их решении (особенно при достаточно большом количестве связанных подзадач)

число подстраиваемых параметров, за которым необходимо следить пользователю, может очень сильно вырасти.

Примером может служить минимизация набора обучающих параметров в системе из нескольких иерархически связанных нейросетей, одни из которых используют для выдачи результата ответы других как входные параметры. Следует заметить, что пользователь вовсе не должен обладать знаниями этих стратегий. Именно для такого случая и разработаны автоматизированные стратегии обучения. Рассмотрим вначале принципы “ручной” стратегии для решения одной подзадачи (сведенной к элементарной - классификации или предикции).

Как правило, первый стратегический шаг для решения подзадачи - создание нейросети с параметрами, максимально выгодными для подтверждения факта принципиальной возможности обучения. В этом случае задается число нейронов и плотность, равные исходному количеству обучающих параметров (в случае классификационных задач эти параметры можно еще увеличить на число классов в задаче). Задаются минимальные значения времени отклика нейросети, характеристика и уровня надежности (для предикторов - требуемый уровень допустимого отклонения). Если такая нейросеть обучается успешно, можно считать, что задача имеет закономерности, связывающие входные параметры и результат.

Далее желательно найти максимальный уровень надежности (в классификаторах) или минимальный уровень отклонения (для предикторов), с которым может обучиться сеть. Для этого нет необходимости генерировать новую нейросеть. Разработанный нами программный инструментарий (программы “MultiNeuron 1.0” и “MultiNeuron 2.0 Professional”) позволяет задать режим доучивания с автоматическим ступенчатым повышением уровня надежности. Это означает, что после каждого успешного полного обучения уровень надежности будет автоматически повышаться на заданную пользователем величину (шаг), и обучение будет продолжаться, пока не перестанет быть возможным. После окончания процесса предыдущий уровень надежности, при котором сеть обучилась полностью, и является максимально возможным для данной сети.

В случае предикторов используется понятие уровня отклонения, который определяет максимальное расхождение между требуемым (заранее известным) и

вычисляемым сетью ответами, при котором пример считается правильно распознанным. При этом качество обучения тем выше, чем меньший уровень отклонения позволяет сети тестировать все примеры правильно, т.е. увеличивается точность ответа сети. Автоматический режим при этом снижает уровень отклонения после каждого полного обучения. Минимально возможный уровень отклонения равен нулю (абсолютное совпадение ответов) и практически не может быть достигнут.

Один из принципов стратегии - создание и обучение для каждой подзадачи не одной, а нескольких нейросетей. Желательно создать несколько сетей с разными характеристиками и после их обучения сравнить их прогностические способности. Весьма вероятно, что при тестировании контрольной выборки разными нейросетями у каждой из них будут свои собственные ошибки. Решение в таком случае принимается на основании голосования и сравнения коэффициентов уверенности каждой из сетей.

Как правило, классификаторы, распознающие несколько классов одновременно, обучаются несколько хуже, чем бинарные классификаторы. В наших исследованиях встречались ситуации, когда задача с 5 классами не могла быть решена одним 5-классовым классификатором, а система из нескольких бинарных классификаторов прекрасно решала задачу. В подобных ситуациях есть несколько способов решения проблемы.

Изменение весов классов.

При обучении классификатора каждому классу задачи можно присвоить вес - "степень внимания", которая будет уделяться нейросетью примерам данного класса. Исходно веса всех классов равны единице. При увеличении веса какого-либо класса оценка каждого примера, принадлежащего этому классу, будет умножаться на его вес. Таким образом, нейросеть будет прилагать особые усилия для распознавания примеров этого класса. При задании нулевого веса оценки всех примеров класса будут равны нулю и эти примеры будут считаться распознанными. Практически это эквивалентно исключению всех примеров класса из обучения.

Изменение весов классов - элемент как стратегии, так и тактики. В стратегическом плане можно спроектировать систему "малых экспертов". Если задача включает, например, 3 класса, можно использовать 3 нейросети, каждая из которых обучается с повышенным весом "своего" класса. В итоге такая нейросеть будет различать все 3 класса, но "свой" класс - особенно хорошо. При тестировании примера решение принимается опять таки голосованием с учетом особого мнения "экспертов".

Разделение задачи.

Если одна нейросеть не может обучиться различать несколько классов или делает это недостаточно хорошо, можно разделить задачу между несколькими разными сетями (принцип, основанный на дихотомии). Это делается следующим образом. Предположим, задача имеет 4 класса. Создается 3 бинарных классификатора. Первый обучается отличать примеры 1 класса от примеров 2,3 и 4 классов, объединенных в один класс. Второй бинарный классификатор обучается отличать примеры 2 класса от примеров 3 и 4 класса, также объединенных в один класс. Третья сеть обучается различать примеры 3 и 4 класса. Тестирование примера при этом осуществляется так: пример подается первой сети. Если она относит его к 1 классу, ответ и есть 1 класс, тестирование прекращается. Если сеть относит пример ко 2 классу, это означает, что на самом деле он может принадлежать 2,3 или 4 классам. Поэтому пример подается второй сети. В зависимости от ее ответа тест может остановиться или продолжиться третьей сетью. В такой системе пример в конце концов оказывается распознанным.

Для предикционных задач подобный способ неприемлем по причине отсутствия классов, поэтому простейший способ - вычисление среднего ответа у нескольких нейросетей. Более сложный метод - привлечение классификаторов для решения предикционных задач. В случае одномерной предикции можно разбить интервал значений ответа на несколько участков (необязательно равных) и обучить классификатор распознавать, к какому участку принадлежит пример. Если точности ответа недостаточно, для каждого участка создается и обучается свой предиктор, умеющий определять ответ только у примеров, отнесенных классификатором к этому участку. Здесь мы наблюдаем иерархическую систему

нейросетей, в которой предикторы подчинены "центральному" классификатору. Подобная система может работать и в задачах классификации, когда число классов слишком велико.

После успешного обучения по вышеприведенным схемам желательно переобучить сети с повышением характеристики, особенно в случае недостаточно удовлетворительных результатов контрольного тестирования.

В общем случае, уменьшение количества связей нейросети (путем контрастирования существующей сети или обучения новых сетей с меньшим числом нейронов и плотностью) лучше всего производить в последнюю очередь, после того, как были настроены предыдущие параметры и результаты тестирования по-прежнему высоки. То же касается и минимизации входных параметров, которая, так же как и принципы автоматической стратегии, будут рассмотрены ниже.

Тактика обучения.

В подавляющем большинстве случаев при обучении сети не требуется никакого вмешательства в этот процесс. Однако все же бывают ситуации, когда сеть не может обучаться. Здесь требуются некоторые действия пользователя, которые мы называем "тактикой обучения". Как правило, нейросеть перестает обучаться не сразу, а через некоторое время после начала процесса. При этом нейросетевая программа сигнализирует оператору о невозможности дальнейшего обучения и в простейшем случае останавливает процесс.

Первое, что необходимо сделать - проверить корректность заданных параметров нейросети. Самая простая причина - недостаточное количество нейронов, малая плотность или слишком высокая характеристика. Если же все параметры заданы оптимально, можно применить специальное средство, называемое “удар”.

Удар может помочь в ситуации, когда сеть попадает в локальный минимум при минимизации целевой функции обучения. Когда производится удар, в карту синаптических весов вносится случайный вклад, максимальная величина которого задается заранее. Новое значение синаптического веса вычисляется по формуле

$$x = x_0 + B * (Random - 0,5) \qquad (4)$$

где x - новое значение синаптического веса, x_0 - исходное значение синаптического веса, B - заданный уровень удара (в диапазоне 0..1), Random - случайно сгенерированное для данного синаптического веса число (в диапазоне 0..1).

После удара обучение продолжается. Проведенные эксперименты показывали, что особенно эффективен удар у сетей с адаптивной матрицей входных сигналов. Задание оптимального уровня удара вручную может потребовать некоторого опыта пользователя. Для этого желательно наблюдать за динамикой снижения средней оценки на протяжении нескольких десятков циклов до момента остановки обучения. Если за этот период оценка снижалась быстро, лучше задавать уровень удара в пределах 0,1-0,2. Если оценка на протяжении многих циклов "зависла" около какого-либо значения, следует задать небольшой уровень удара (0,0001 - 0,001). Задание максимального уровня удара (1,0) практически полностью обновляет карту весов, что равнозначно инициализации новой сети.

Частое применение удара небольшого уровня может быть полезным для создания помехоустойчивых нейросетей.

В разработанных нами нейросетевых программах используется как ручной, так и автоматический режимы удара. В последнем случае после остановки обучения программа производит удар и продолжает обучение. Только если после трех (или количества, определенного пользователем) попыток удара обучение по-прежнему невозможно, программа останавливает процесс.

Если и применение удара не приносит эффекта, причину невозможности обучения следует искать в обучающей выборке. Как уже говорилось, самая частая причина - наличие в выборке идентичных по обучающим параметрам примеров, имеющих различный ответ. В принципе, это почти тупиковая ситуация, и единственным выходом из нее является увеличение количества обучающих параметров, при котором добавленные параметры будут различными у таких примеров. В самом деле, очевидно, что различные ответы в такой ситуации определяются факторами, не нашедшими отражение в обучающих параметрах. В простейшем случае можно исключить один из идентичных примеров, однако в таком случае в экспертную систему заранее закладывается возможность ошибки.

Невозможность дальнейшего обучения в классификаторах иногда может быть спровоцирована неверной постановкой задачи, точнее, заданной классификационной моделью. Например, при бинарной классификации примеры одного из классов представляют на самом деле два различных подкласса, группирующихся по параметрам "по разные стороны" от другого класса задачи. В этом случае можно использовать нейросеть для коррекции классификационной модели путем кластеризации примеров.

Существует несколько методов нейросетевой кластеризации, наиболее простой из которых заключается в последовательном исключении из процесса обучения наиболее трудных примеров с помощью “слабых” нейросетей, настройка параметров которых заранее не дает возможности для полного обучения (например, слишком малое число нейронов). В процессе прохождения каждого цикла обучения программа отслеживает самый трудный пример на этом цикле - пример с наибольшей оценкой. После захода сети в тупик можно исключить этот пример из дальнейшего обучения, пометив его, и продолжить процесс. В конце концов может накопиться некоторое количество исключенных примеров, после чего сеть обучится полностью. Часто получается, что все исключенные примеры принадлежат только одному из классов задачи, что является свидетельством неоднородности класса. Это можно проверить путем обычных статистических методов, сравнив группу исключенных примеров как с оставшимися примерами этого же класса, так и с примерами других классов. Различия могут послужить поводом для изменения классификационной модели.

Это не самый лучший способ кластеризации, однако он может с успехом применяться как при создании экспертных систем, так и в научных исследованиях для изучения различных явлений. По сравнению с многочисленными существующими способами кластеризации он имеет определенное преимущество, заключающееся в возможности “наведенной” кластеризации, т.е. такой, к которой пользователь “подталкивает” программу, не зная заранее точных критериев - параметров, по которым ее следует проводить, и имея лишь интуитивное представление о возможном результате. В практике врача-исследователя такое может встречаться довольно часто, например, когда он интуитивно относит какие-либо

2 примера к разным классам, однако не может формализовать принцип разделения. Применение нейросети может помочь в этом случае найти закономерность и экстраполировать ее на большой набор данных.

Как уже говорилось, изменение весов классов может считаться тактическим действием, так как может осуществляться прямо в процессе обучения. Поводом для изменения веса класса обычно служит его средняя оценка, значительно (в несколько раз) превосходящая оценки остальных классов.

Дополнительные возможности нейросетей и автоматизированные стратегия и тактика обучения

В этом разделе будут рассмотрены несколько дополнительных возможностей нейросетей, разработанных в рамках построения теории и методологии создания нейросетевых экспертных систем, и полезных для практического применения.

Обученная нейросеть не только умеет распознавать примеры, но и хранит достаточно важную информацию, которую можно получить, грамотно сформулировав вопрос для сети. Возможность нейросетей содержательно отвечать на поставленные вопросы позволяет использовать их не только как "решающие" устройства, но и как инструменты для проведения научных исследований.

Вычисление информативности обучающих параметров и минимизация их набора.

Врач, ставящий диагноз больному, всегда старается анализировать весь комплекс сведений о пациенте - жалобы, анамнез болезни и жизни, данные клинического осмотра и лабораторных анализов. При этом одна часть параметров имеет принципиальное значение для принятия решения, другая же не столь важна и даже может просто игнорироваться.

Нейросеть, принимающая решение на основе опыта, поступает сходным образом. Поэтому исследователю очень важно знать, какие из параметров важны для сети, а какие - не очень. Следует сказать, что абсолютно незначимых параметров практически не бывает, и каждый входной сигнал в большей или меньшей степени влияет на принятие решения.

Разработанный нами метод позволяет выявить относительную значимость входных параметров, которая может представлять самостоятельный интерес или служить для последующей минимизации их набора [47].

Для этого включаются несколько циклов обучения нейронной сети с заранее завышенным уровнем надежности и внесением в матрицу синапсов случайного вклада после прохождения каждого цикла. При прохождении циклов используется метод подсчета градиентов по входным сигналам, подаваемым на нейронную сеть, и последующего их сравнения, осуществляемого двумя различными методами по выбору пользователя программы.

Первый метод насчитывает значимость каждого параметра как максимальное значение модуля градиента по всем примерам. Этот режим полезно применять в тех случаях, когда в обучающей выборке имеются примеры, "выбивающиеся" из общей массы, и которые могут существенно влиять на принятие решения об ответе. Данный метод будет прежде всего учитывать наличие таких примеров.

Второй метод насчитывает значимость каждого параметра как среднюю величину модуля градиента по всем примерам обучающей выборки. Результат применения этого метода показывает среднюю значимость параметров по всей обучающей выборке. Если в обучающей выборке имеются примеры, "выбивающиеся" из общей массы, и которые могут существенно влиять на принятие решения об ответе, то влияние таких примеров на значимость параметров будет нивелироваться. Данный метод полезно применять в тех случаях, когда обучающая выборка достаточно однородна и необходимо, чтобы возможные "выбивающиеся" примеры существенно не влияли на оценку значимости параметров. Результатом применения метода является набор значений, каждое из которых соответствует определенному входному параметру и может сравниваться с остальными.

Применение метода имеет ценность как для научных исследований, так и при практическом применении нейросетей для экспертных систем, так как позволяет уменьшать набор входных параметров, подаваемых нейросети для получения ответа. Действительно, часто для диагностики и прогноза больному проводят сложные и дорогостоящие методы исследования, порой небезвредные для

здоровья. Однако во многих случаях представляется возможным получить ответ и без них.

В нейросетевых программах предусмотрен как ручной, так и автоматический режимы минимизации входных параметров, причем последний может применяться как самостоятельно, так и в случае задания автоматического определения стратегии обучения. Использование “ручной” минимизации параметров имеет смысл только после того, как нейросеть достаточно хорошо обучена, в противном случае исключение параметров может привести к ухудшению экстраполяционных возможностей сети и затруднению последующего обучения.

Если режим минимизации включен, он начинает работать после полного обучения нейронной сети. При этом осуществляется вычисление относительных показателей значимости и входной синапс, имеющий минимальную значимость. После этого этот синапс отключается от приема сигналов, и обучение продолжается. Предполагается, что нейросеть сможет обучиться без данного параметра. Если сеть вновь обучилась, после повторного вычисления значимости выключается следующий синапс и т.д. В итоге сеть перестанет обучаться, когда набор входных параметров будет слишком мал для полноценного обучения. После этого можно проанализировать различные варианты исключения параметров и выбрать их оптимальный набор. Эксперименты с самыми разными медицинскими задачами показывали, что во многих случаях приблизительно четверть обучающих параметров может быть исключена совершенно безболезненно. Это подтверждает гипотезы об априорной избыточности данных в задачах медицинской диагностики и прогнозирования, решаемых человеком.

Подстройка параметров примера для получения требуемого ответа нейросети.

При применении обученной нейронной сети может возникнуть обратная задача, относящаяся к области нейросетевого моделирования - выяснить, в каком направлении и на какие величины необходимо изменить параметры примера, чтобы сеть выдала требуемый ответ. Приведем простой гипотетический пример. При тестировании примера сетью, разделяющей больных и здоровых, получен ответ "больной". Как нужно изменить параметры этого примера, чтобы он "пе-

решел" в класс "здоровый"? Как уже говорилось, можно вручную изменять параметры, каждый раз тестируя пример после такого изменения. Однако это не самый быстрый и удобный способ, особенно, если число параметров велико, а решение принимается несколькими нейросетями. Поэтому нами был разработан специальный автоматизированный метод подстройки параметров для получения требуемого ответа нейросети.

При решении данной проблемы прежде всего необходимо разделить все параметры примера на подстраиваемые и неподстраиваемые. Например, невозможно изменить возраст пациента, поэтому такой параметр является неподстраиваемым. После определения набора подстраиваемых параметров для каждого из них задается диапазон возможных изменений. Необходимо помнить, что в случае каждого примера эти диапазоны могут диктоваться конкретной ситуацией (одному больному допустимо снизить частоту сердечных сокращений до 50, другому же - только до 70).

После этого для каждой нейросети (если их несколько) необходимо задать требуемый ответ - тот, которого должна добиться эта сеть. Важно отметить, что выполняемая затем автоматическая процедура подстройки выполняется одновременно несколькими нейросетями, так как подстройка, выполненная одной сетью, может резко ухудшить ответ других. Таким образом, каждая сеть, как правило, идет на некоторый компромисс.

В итоге вырабатывается оптимальное решение, одинаково хорошо (или плохо) удовлетворяющее все сети, принявшие участие в процедуре. Исследователь может проанализировать решение, сравнив ответы каждой нейросети до и после подстройки примера [48].

Автоматическое задание стратегии обучения.

Рассмотрим полностью автоматизированное задание стратегии построения решающего блока. Прежде всего определяется список подзадач, причем в каждой подзадаче определяется список обучающих параметров (он может быть общим для всех подзадач) и ответы. Затем задается, сколько нейросетей-экспертов будут решать каждую подзадачу (в простейшем случае - по одному). Далее определяется, будет ли проводится минимизация обучающих параметров. Если да, то

можно задать произвольную минимизацию (нахождение минимально возможного для решения задачи списка параметров) или определяется набор параметров, который желательно оставить в списке или, наоборот, исключить из списка. Эту операцию можно провести отдельно для каждой подзадачи, для групп подзадач или для всей задачи в целом. Как дополнительный параметр, можно указать для каждой подзадачи контрольные выборки, на которых будут проверяться результаты обучения и требуемое качество обучения (в процентах правильно распознанных примеров (для классификаторов) или максимально допустимого отклонения (для предикторов). Это все, что должен сделать пользователь при автоматизированном построении решающего блока.

Непосредственно обучение проводится исходя из следующей стратегии. Обучение каждого эксперта в каждой подзадаче начинается с автоматической инициализации нейросети минимальной конфигурации (число нейронов - 2, плотность - 1, характеристика - 0,1, уровень надежности - 0,1 (для классификаторов), уровень отклонения для предикторов - требуемый или вычисленный исходя из характера значений ответа (в самом общем случае берется 1/3 минимальной разности между значениями ответа), время отклика - 2. Проводится обучение нейросети. Дальнейшие действия определяются результатом обучения. В случае успешного обучения проводится тест указанной контрольной выборки и в случае неудовлетворительного результата инициализируется новая сеть с увеличенной характеристикой (шаг увеличения - 0,1). При невозможности обучения инициализируется новая сеть с большим числом нейронов и плотностью.

Таким способом проводится обучение сетей по всем подзадачам. Обученные сети запоминаются. Если задана минимизация параметров, создаются рабочие копии обученных нейросетей и проводится вычисление значимости параметров для каждой сети, после чего следует минимизация параметров у каждой нейросети. Если требуется связанная минимизация (для нескольких нейросетей одновременно), окончательное отключение минимально значимого на данном этапе обучения параметра производится только после того, как все нейросети в связанной группе успешно обучились с исключением данного параметра. В процес-

се обучения могут автоматически применяться тактические методы (удар, изменение весов классов).

Исходя из приведенного описания, управление стратегией осуществляется по заранее заданному алгоритму. При необходимости пользователь может корректировать стратегию в процессе построения решающего блока на любом его этапе.

Экспертные системы, созданные на основе нейросетевой технологии группой «НейроКомп»

Создание каждой ЭС проводилось согласно разработанной технологии и включало: изучение проблемы; постановку задачи; набор обучающих данных и тестирующих примеров; обучение нейросетей; определение оптимальной схемы ЭС; проведение дополнительных экспериментов; разработку и создание интерфейса программы; подключение к ней обученных нейросетей; испытание системы на примерах, не входящих в обучающую выборку; доучивание системы на этих примерах.

Прогнозирование осложнений инфаркта миокарда.

Поиски возможностей прогнозирования осложнений, которые могут возникнуть в госпитальный период инфаркта миокарда, очень актуальны и являются одной из наиболее сложных задач кардиологии. Прогнозирование необходимо осуществлять при поступлении больного в стационар, сразу же после проведения стандартных методов обследования. Оно должно быть быстрым, проводиться неоднократно в процессе наблюдения за больным по мере поступления новых данных о его состоянии.

Получение прогноза осложнений позволяет врачу целенаправленно проводить профилактику, усилить наблюдение за больным, скорректировать режим физической активности пациента (особенно в предполагаемые сроки возникновения осложнения). Прогноз может определять более длительное и интенсивное лечение антикоагулянтами при угрозе тромбоэмболических осложнений, антиаритмическими препаратами - для профилактики аритмий.

При создании базового ядра ЭС прогнозирования осложнений инфаркта миокарда выбрано 4 вида осложнений, достаточно частых и/или довольно опасных [49]. Это фибрилляция предсердий, тромбоэмболические осложнения, перикардит и возникновение/усугубление хронической СН. При постановке задачи мы исходили из необходимости прогнозировать возможность возникновения каждого из четырех выбранных осложнений в отдельности и возможного срока его появления, считая от момента поступления больного в стационар. Таким образом, задача разбивается на 8 подзадач, четыре из которых решаются нейросетями-классификаторами (возникновение осложнений), и четыре - нейросетями-предикторами (сроки возникновения осложнений).

Были выбраны 32 обучающих параметра, отражающие клиническое состояние больного инфарктом миокарда на момент поступления в клинику, данные анамнеза и результаты лабораторных и функциональных исследований: возраст, пол, глубина и локализация инфаркта (по данным электрокардиографии), количество инфарктов в анамнезе, характеристика предшествующей стенокардии, наличие и тяжесть гипертонической болезни и сердечной недостаточности, наличие в анамнезе нарушений сердечного ритма и проводимости, эндокринных заболеваний, тромбоэмболий, хронического бронхита, концентрация калия и натрия в крови, частота сердечных сокращений, характеристика выбранных показателей ЭКГ, размеры отделов сердца по данным эхокардиографии.

Исследованы 300 клинических примеров (Таб. 4). Для экспериментов из общей выборки отдельно для каждого типа нейросетей были выделены обучающая группа (250 человек) и контрольная (тестирующая) группа (50 человек). Разделение выборки производилось случайным образом. Для прогнозирования возникновения каждого осложнения создавались несколько нейросетей, составляющих консилиум. Все нейросети вначале обучались на 250 пациентах обучающей выборки, а затем тестировались на контрольной группе. Результаты теста определялись голосованием в каждом консилиуме. Результаты теста контрольной выборки для прогнозирования фибрилляции предсердий приведены в Таб. 5.

Таблица 4.

Количество больных инфарктом миокарда поклассам в каждой из четырех выделенных по осложнениям подгрупп

Осложнение		**Количество больных**
Фибрилляция предсердий	нет пароксизмальная форма постоянная форма	217 60 23
Перикардит	нет есть	172 128
Тромбоэмболические осложнения	нет есть	242 58
Возникновение или усугубление сердечной недостаточности	нет есть	144 156

Таблица 5.

Результаты тестирования консилиума нейросетей, прогнозирующих возникновение и форму фибрилляции предсердий (ФП), проведенные на контрольной выборке из 50 примеров, не участвующих в обучении

Известный класс	Вычислено как... Класс 1	Класс 2	Класс 3
Нет ФП - 30 примеров	**28**	2	-
Пароксизмальная форма ФП - 12 примеров	-	**11**	1
Постоянная форма ФП - 8 примеров	-	1	**7**

Процент правильно распознанных примеров в тестирующей выборке при прогнозировании возникновения перикардита составил 76%, тромбоэмболий - 82%, возникновения/усугубления сердечной недостаточности - 78%.

Отдельно были созданы и обучены нейросети-предикторы для прогноза сроков возникновения осложнений (начиная с момента поступления больного в клинику).

Система назначения оптимальной стратегии лечения больных облитерирующим тромбангиитом и прогнозирования его непосредственных исходов.

Облитерирующий тромбангиит (болезнь Бюргера) - тяжелое воспалительное заболевание сосудов мелкого и среднего калибра, сопровождающееся тромбозом и нарушением их проходимости. Этиология этого заболевания до настоящего времени остается неизвестной. Подавляющее большинство больных тромбангиитом - мужчины молодого возраста (18 - 45 лет).

Лечение облитерирующего тромбангиита - трудная, далеко еще не решенная задача. В начальных стадиях заболевания обычно ограничиваются терапевтическими мероприятиями - назначением дезагрегантов, противовоспалительных и антигистаминных препаратов. Однако заболевание часто носит злокачественный характер и быстро приводит к ампутации конечности.

В 1990 - 1994 гг. в отделении хирургии сосудов Краевой Клинической Больницы № 1 г. Красноярска под наблюдением находилось 130 больных облитерирующим тромбангиитом. На каждого больного заполнялась анкета, состоящая из 3 разделов.

Первый раздел (104 пункта) включает вопросы, касающиеся анамнеза жизни и данного заболевания, сопутствующих заболеваний; состояние органов и систем, подробное описание имеющегося у больного тромбангиита с характеристикой состояния сосудов конечностей, данные лабораторных и инструментальных методов исследования, характеристику проводившегося ранее лечения. Другими словами, этот раздел отражает исходный статус больного на момент его поступления в стационар.

Второй раздел (11 пунктов) характеризует проведенное в стационаре лечение (консервативное и/или оперативное).

Третий раздел (4 пункта) содержит сведения о непосредственных исходах проведенного лечения.

Существующие методы лечения тромбангиита часто малоэффективны и процент выполняемых ампутаций остается высоким. Перед врачом стоит задача подобрать оптимальное сочетание методов лечения, действующих на ведущие

звенья патогенеза у конкретного больного. Целью проводимого исследования стало создание нейросетевой ЭС для прогноза непосредственных исходов заболевания и выбора оптимального сочетания терапевтических и хирургических воздействий. Соответственно этому ЭС подразделяется на два функциональных блока, каждый из которых решает свой круг задач [50].

Один блок (блок "И" - исходы) прогнозирует непосредственные исходы заболевания, которые зависят от двух групп параметров. Первая группа - исходный статус больного, фиксированные параметры, отражающие состояние больного на момент поступления в клинику, а также данные анамнеза. Однако исход заболевания зависит не только от исходных параметров, но и от проводимого в клинике лечения. Поэтому вторая группа параметров, необходимая для прогноза - примененные методы лечения. Эти параметры неизвестны при поступлении, на этот момент их можно только предполагать. Однако при обучении нейросети используются уже пролеченные больные с известным набором терапевтических и хирургических воздействий. Обучив нейросети прогнозировать исходы тромбангиита в зависимости от исходного статуса и проведенного лечения, можно моделировать результат, оставляя неизменными фиксированные параметры и подстраивая предполагаемые методы лечения.

Другой блок ЭС обучается прямому выбору наиболее оптимальных методов лечения (блок "Л"), используя только первый, фиксированный набор параметров (Рис. 6).

С учетом поставленной задачи были сформированы подгруппы примеров для обучения нейросетей. Для решения подзадач блока "И" примеры были сгруппированы по классам четырьмя способами (Таб. 6), для блока "Л" - 11 способами (Таб. 7).

Для тестирования обученных нейросетей использовались 35 клинических примеров обследованных и пролеченных больных с известными исходами заболевания. Эти примеры не входили в обучающую выборку. Тест каждого примера проводился следующим образом. Сначала тестировались 4 нейросети, прогнозирующие исходы заболевания, причем в качестве параметров лечения, проведенного в клинике, использовались данные о реально назначенном хирур-

гами лечении. Затем пример тестировался нейросетями, назначающими оптимальный набор методов лечения.

После этого проводился повторный тест нейросетями, прогнозирующими исходы, но теперь в пример подставлялись предполагаемые методы лечения, назначенные нейросетями.

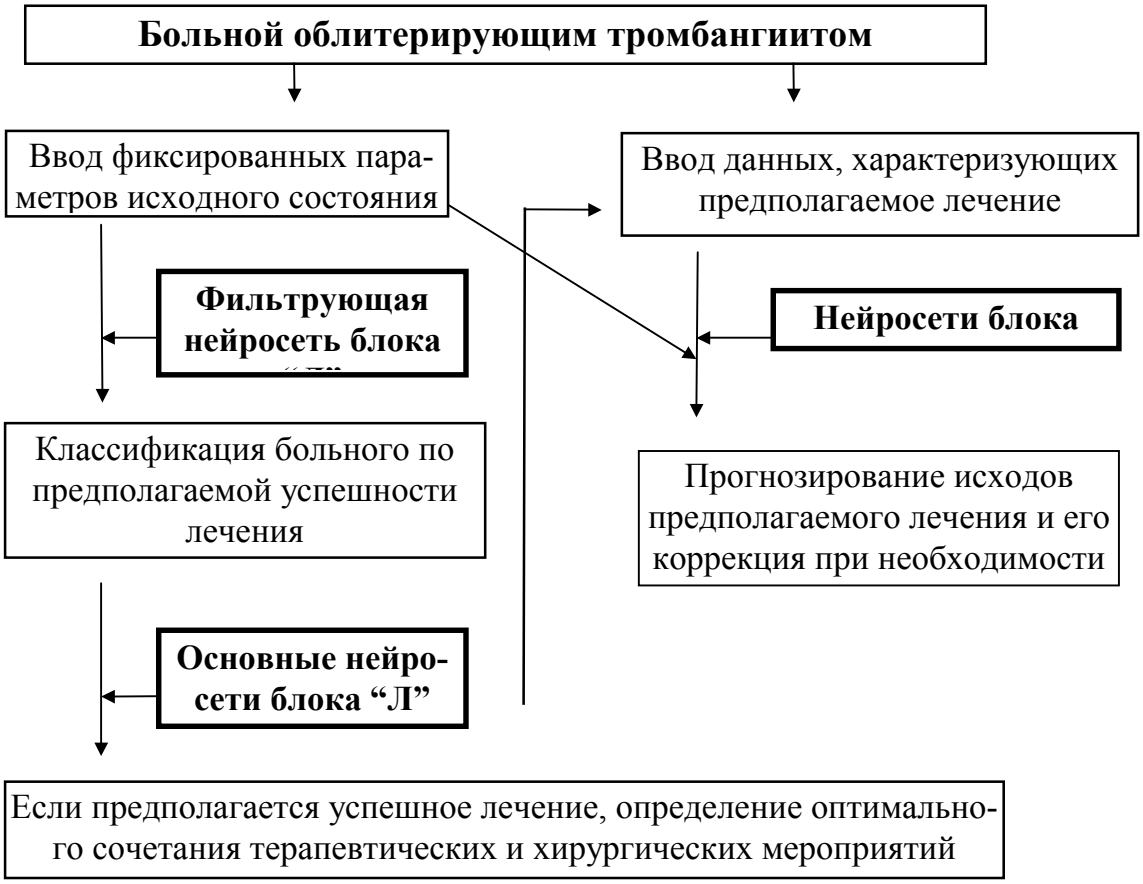


Рисунок 6. Схема функционирования ЭС

Таблица 6.

Подзадачи первого (“И”) блока ЭС

Подзадача	Классы	Число примеров
1. Прогноз динамики ишемии	1. Уменьшение ишемии 2. Без изменений 3. Усиление ишемии	85 38 7
2. Прогноз динамики трофических расстройств	1. Динамики нет 2. Уменьшение трофических расстройств	50 80
3. Прогноз исчезновения болей в покое	1. Боли остались 2. Боли исчезли	57 73
4. Ампутация	1. Не производилась 2. Производилась	123 7

Таблица 7.

Подзадачи второго (“Л”) блока ЭС

Подзадача	Классы	Число примеров
Прогнозирование возможности успешного лечения (“фильтрующая” нейросеть)	1. Больные с неблагоприятным исходом после лечения 2. Больные с благоприятным исходом и больные со смоделированным лечением	24 106
1. Назначение вазапростана	1. Не назначать 2. Назначать	122 8
2. Плазмоферез, иммунопротекторы	1. Не назначать 2. Назначать	111 19
3. Пульстерапия	1. Не проводить 2. Проводить	126 4
4. Симпатэктомия	1. Не проводить 2. Односторонняя 3. Двухсторонняя	104 21 5
5. Деструкция надпочечников	1. Не проводить 2. Односторонняя 3. Двухсторонняя	84 25 21
6. Эмболизация надпочечников	1. Не проводить 2. Односторонняя 3. Двухсторонняя	82 41 7
7. Компактотомия или РОТ	1. Не проводить 2. Проводить	120 10
8. Ревизия	1. Не проводить 2. Проводить	127 3
9. Ампутация	1. Не проводить 2. Проводить	125 5
10. Реконструктивная операция	1. Не проводить 2. Проводить	126 4
11. Некрэктомия	1. Не проводить 2. Проводить	125 5

Рассмотрим результаты первого теста. Нейросеть, прогнозирующая динамику ишемии, сделала правильное заключение в 29 случаях из 35 (82,9%). Регресс трофических расстройств прогнозировался правильно в 14 из 17 случаях имеющихся трофических расстройств (только у 17 больных имелись трофические расстройства, поэтому в остальных случаях прогноз не имел смысла). Это составило 82,4%. Нейросеть, прогнозирующая динамику болевого синдрома,

сделала правильное предсказание в 22 из 25 случаев (88%). Ответы сети, отвечающей за предполагаемую ампутацию, совпали в 33 случаях из 35 (94,3%).

Всего в тестирующей выборке имелось 9 больных, у которых не совпал хотя бы один прогноз. У остальных пациентов все прогнозы делались 4 нейросетями правильно. Из этих 9 больных 7 были отнесены нейросетью-”фильтром” к категории “трудных”.

Интерес представляют два формально неверных результата нейросети, прогнозирующей ампутацию. У данных больных ампутация не была проведена в сроки госпитализации, однако была выполнена в первые 2 месяца после выписки. Нейросеть правильно предсказала отсроченные исходы, поэтому эти два случая нельзя считать явными ошибками.

Рассмотрим результаты теста нейросетей, назначающих лечение больным. У 12 примеров из 25 (35 примеров минус признанные “трудными” нейросетью-”фильтром”) лечение, назначенное нейросетями, полностью совпало с реально назначенным хирургами, и только в одном случае прогноз исходов оправдался лишь в более поздние сроки. Рассмотрим этот случай.

Больной С., 42 лет, болен в течение 5 лет, поступил с критической ишемией (IV степени). Больному была назначена консервативная терапия и вазапростан. На фоне такой терапии 3 из 4 нейросетей прогнозировали улучшение состояния. Однако на момент выписки эффекта от лечения не наблюдалось. Тем не менее, при дальнейшем наблюдении за больным отмечался регресс ишемии, регресс трофических расстройств и купирование болей в покое.

В остальных случаях наблюдались некоторые расхождения. При этом у 10 больных нейросети назначили другое лечение, которое (по результатам анализа хирургов), можно было действительно назначить данным больным. Исходы, прогнозируемые после такого лечения, были такими же, как и в случае реально назначенных лечебных мероприятий. В 3 примерах лечение, назначенное нейросетями, было похоже на реально назначенное, однако превосходило его по мощности. В этих случаях прогнозировались исходы, гораздо лучшие, чем наблюдавшиеся в реальности.

Система дифференциальной диагностики “острого живота”.

Задача дифференциальной диагностики "острого живота", несмотря на огромное количество разработок в этой области, для практической хирургии остается одной из самых актуальных проблем. Свыше 90% больных хирургических стационаров - больные с острыми хирургическими заболеваниями органов брюшной полости; оперативные вмешательства при этой патологии составляют от 20 до 40% всей оперативной деятельности хирургических стационаров.

При постановке диагноза и планировании лечения от хирурга требуется принятие единственно правильного решения, от которого часто зависит жизнь больного. Решение должно быть принято как можно в более сжатые сроки, причем в условиях, зачастую не позволяющих получить развернутые данные лабораторных и инструментальных исследований.

Опыт практической хирургии показывает, что наиболее часто встречающимися острыми заболеваниями брюшной полости являются следующие: острый аппендицит (ОА), острый холецистит (ОХ), острый панкреатит (ОП), осложнения язвенной болезни желудка и двенадцатиперстной кишки, острая кишечная непроходимость (ОКН), дифференциальная диагностика которых представляет в некоторых случаях значительные сложности, особенно при поступлении больных в клинику в отдаленные сроки от начала заболевания или с изначально атипичным или стертым процессом.

Изучены 216 историй болезней из архивов клинических больниц г. Красноярска, специализирующихся на приеме экстренной хирургической патологии - ГКБ N 7, ККБ N1, ГБСМП, что позволило работать с уже верифицированными диагнозами, подтвержденными комплексными клинико-лабораторными исследованиями, данными гистологического исследования и операционных находок. Данные еще о 49 пациентах были получены в результате непосредственного клинического наблюдения от момента поступления в хирургический стационар и до постановки окончательного диагноза.

Всего проанализировано 265 клинических примеров (ОА - 63, ОХ - 64, ОП - 52, ОКН - 27, КЯБ - 37, ПЯБ - 22).

При определении входных параметров, необходимых для работы ЭС, проанализирована структура историй болезней различных клиник, данные литературы, возможности лабораторно-инструментальных баз хирургических стационаров. Таким образом, набор входных параметров отражает клинический минимум обследования пациента, используемый врачами хирургических отделений в практической деятельности.

Для сбора клинических данных была разработана анкета, содержащая набор входных параметров и диагноз, являющийся ответом (всего 131 показатель).

При постановке задачи для обучения нейросетей мы исходили из того, что ЭС должна выбирать один или несколько предполагаемых диагнозов из заданного набора (6 диагнозов) на основании 131 параметра пациента при поступлении в клинику. Для экспериментов создавались наборы из 7 нейросетей. В каждом наборе одна нейросеть представляла собой шестиклассовый классификатор, выдающий в качестве ответа один диагноз из шести, а остальные шесть нейросетей - бинарные классификаторы, обучающиеся отличать каждый из рассматриваемых диагнозов от всех остальных.

Для обучения нейросетей были взяты 216 примеров, данные которых были взяты из историй болезни пациентов с уже подтвержденными диагнозами. Остальные 49 примеров больных, наблюдавшихся в клинике, были оставлены для тестирования ЭС.

Результаты клинической проверки приведены в таблице 8.

Прогностическая способность ЭС после стартового обучения составила 83,7%. Однако мы провели тщательный разбор всех случаев ошибок, который выявил следующие закономерности.

Основное количество ошибок возникало при диагностике острого холецистита и острого панкреатита, причем в этих случаях нейросистема путала эти диагнозы между собой. Более того, при этом правильный диагноз набирал вес (сумму выходных сигналов нейронов нейросетей, ответственных за данный класс), находящийся на втором месте. Это означает, что в случае диагностики острого холецистита (наибольший вес) острый панкреатит оказывался на втором месте (следующий по величине вес) и наоборот. Как известно, при развитии па-

тологического процесса в одном из органов панкреато-дуоденальной системы другие органы, как правило, не остаются интактными, и клинически решить вопрос о преобладании патологии в одном из них зачастую не представляется возможным. В таких случаях больному выставляется диагноз холецистопанкреатит. 5 из 6 примеров, которые при тестировании распознавались ошибочно (острый холецистит как панкреатит и наоборот) относились, судя по анализу клинических данных, как раз именно к такой группе. Поэтому результаты, выданные нейросистемой для этих примеров, не являются явными диагностическими ошибками, так как в этих случаях оба диагноза занимали первые два места по набранным весам.

Таблица 8.

Результаты теста 49 примеров консилиумом из 14 нейросетей

Диагноз	Число примеров	Число правильно распознанных примеров
ОА	19	19 (100%)
ОХ	7	4 (57%)
ОП	12	10 (84%)
ОКН	2	1 (50%)
КЯБ	3	3 (100%)
ПЯБ	6	4 (66%)
Всего	49	41 (83,7%)

Для нейросистемы представлял определенные трудности дифференциальный диагноз между острым панкреатитом и перфоративной язвой (2 ошибки), что имеет место и в клинической практике - при остром панкреатите с выраженным болевым синдромом в 4 случаях из 265 проанализированных клинических примеров, врачами приемного покоя выносились оба диагноза в качестве предварительного и предпочтение одному из них отдавалось уже в процессе динамического наблюдения за состоянием пациента. Также следует учитывать, что достаточно небольшое количество примеров 6 класса (перфоративная язва - 22), как показали результаты теста, несомненно, недостаточно для качественного обучения нейросетей.

В одном случае ЭС распознавала пример 4 класса (ОКН) как острый аппендицит. При анализе примера выявлено, что оба диагноза вынесены врачом

приемного покоя в качестве предварительного и в дальнейшем, при изучении данных истории болезни выяснилось, что больная после проведения консервативной терапии была выписана с диагнозом "кишечная колика". Данная ошибка достаточно серьезная и говорит о необходимости расширения списка диагнозов, определяемых ЭС.

При создании ЭС мы исходили из необходимости выведения полной информации о весах всех диагнозов, выдаваемых системой в каждом случае. Так как диагнозы холецистопанкреатита подразумевают наличие одновременно 2 ответов (два класса), случаи, когда веса этих двух ответов находятся рядом, не могут рассматриваться как ошибки работы системы, при этом наиболее вероятный диагноз формулируется как холецистопанкреатит. С учетом этой поправки, прогностическая способность системы составляет 93,9% (3 ошибки из 49 примеров).

Проведенный анализ значимости обучающих данных показал, что тремя наиболее значимыми параметрами являются боль в правом подреберье, симптом Ортнера и наличие камней в желчевыводящих путях. В число наиболее значимых параметров попали основные характеристики болевого синдрома, важные диагностические признаки (симптомы Кохера, Щеткина - Блюмберга, Ровзинга, Воскресенского) и большинство основных характеристик клинического состояния пациента. Минимально допустимым для работы ЭС является набор из 29 параметров, при этом необходимо учитывать, что увеличение числа вводимых параметров существенно повышает диагностическую точность системы.

Нейросети для изучения иммунореактивности

Недостаток четких лабораторных критериев иммунологической недостаточности, большая вариабельность различных показателей состояния иммунной системы [51], особенно при разных патологиях, определяет необходимость поиска простых и эффективных методов, позволяющих дифференцировать состояния иммунореактивности. Наибольшей информативностью обладают показатели субпопуляционного состава лимфоцитов крови, а также параметры их внутриклеточного метаболизма [52]. Однако трудно делать вывод о состоянии иммун-

ной системы, основываясь на величинах отдельных параметров. Необходима комплексная оценка, учитывающая многообразие гомеостатических состояний иммунной системы.

Мы попытались применить нейросети для диагностики вторичной иммунологической недостаточности - иммунодефицита (ИД) [53]. Причинами его являются многие факторы в различном сочетании (другие заболевания, вредные привычки, наследственная предрасположенность, экологические и социальные условия, питание, авитаминоз и др.). В отличие от первичного ИД, причиной которого являются явные, легко выявляемые генетические дефекты, диагноз вторичного ИД весьма сложен.

При иммунологическом обследовании в крови больного определяют различные показатели, главные из которых - количество лимфоцитов (клеток иммунной системы) и абсолютные и относительные (соотношения) количества различных классов лимфоцитов. Кроме того, полезно оценить активность основных внутриклеточных ферментов лимфоцитов и содержание в крови иммуноглобулинов. На основании полученной иммунограммы нужно установить, имеется ли у обследуемого ИД. Сложность заключается в том, что даже у здорового человека могут наблюдаться существенные сдвиги отдельных параметров, а яркий больной может иметь незначительные изменения. Все это связано во-первых, с индивидуальными особенностями иммунной системы, а во-вторых, иммунная система может на короткое время среагировать на какие-либо внешние воздействия (стресс, переохлаждение, переутомление) весьма непредсказуемым образом. Поэтому необходимо оценивать весь комплекс параметров сразу во всех имеющихся взаимосвязях, что на глаз сделать довольно трудно даже для достаточно опытного врача.

Первоначально нейросети была поставлена задача научиться по набору параметров давать простой ответ: имеется ИД у больного или нет. Была сформирована обучающая выборка, состоящая из тщательно обследованных людей, диагноз которым был установлен на основании клинического и иммунологического анализа. Она состояла из двух классов; первый - здоровые люди (51 человек), второй - с диагнозом ИД (42 человека). У обследуемых изучалась активность ос-

новных внутриклеточных ферментов лимфоцитов [54], проводились иммунологические тесты первого уровня [55]. Определялось содержание в крови лейкоцитов, иммуноглобулинов различных классов (IgA, IgM, IgG, IgE), циркулирующих иммунных комплексов (ЦИК), относительное и абсолютное количество лимфоцитов и их субпопуляций: тотальных розеткообразующих клеток (т-РОК), теофиллинрезистентных (ТФР-РОК), теофиллинчувствительных (ТФЧ-РОК), ранних (р-РОК), и стабильных (с-РОК) [56]. Эти показатели служили обучающими параметрами.

Несмотря на кажущуюся простоту задачи и многочисленные попытки, нейросеть не смогла обучиться решать задачу в поставленном виде. Тогда было решено постепенно исключать из обучения самые трудные примеры (имеющие максимальную оценку), чтобы добиться полного обучения на оставшихся. Программа MN_TRAIN делала это автоматически, исключая труднейший пример каждый раз, когда нейросеть заходила в локальный минимум и не могла обучаться далее. Естественно, в конце концов сеть обучилась полностью, исключив из обучающей выборки 30 примеров. После обучения было обнаружено, что все до одного исключенные примеры относятся ко 2 классу (больные с ИД). Таким образом, группа примеров 2 класса оказалась разделенной нейросетью на 2 подгруппы (исключенную и оставшуюся). Далее была проведена статистическая обработка полученных групп.

Общая группа индивидуумов с диагнозом ИД достоверно отличалась от группы здоровых только по одному показателю клеточного метаболизма (фон НАДФ-МДГ, Таб. 9) и одному показателю клеточного иммунитета (соотношение ТФР-РОК/ТФЧ-РОК, Таб. 10). Оставшаяся же группа лиц достоверно отличалась от здоровых людей уже по 8 исследуемым параметрам (причем все они превышали аналогичные параметры у здоровых людей).

Таблица 9.

Активность НАД(Ф)-зависимых дегидрогеназ лимфоцитов крови у здоровых людей, лиц с иммунодефицитом и в группах, выделенных нейросетью (мкЕ + m)

		Группы лиц с ИД		
Параметры	Здоровые n=51 1	общая n=42 2	оставшаяся n=12 3	исключенная n=30 4
Фон Г6ФДГ	0,03±0,002	0,03±0,003	0,03±0,006	0,03±0,003
Фон Г3ФДГ	5,15±0,71	5,12±0,78	7,49±2,32	4,17±0,59
Фон ЛДГ	2,51±0,44	2,44±0,33	3,50±0,59	2,01±0,38 $P_3<0,05$
Фон МДГ	5,10±0,77	4,51±0,69	6,97±1,65	3,52±0,66 $P_3<0,05$
Фон НАДФ-МДГ	0,11±0,01	0,08±0,007 $P_1<0,05$	0,09±0,01	0,08±0,008 $P_3<0,05$
Фон НАДФ-ГДГ	0,44±0,11	0,48±0,12	1,05±0,35 $P_1<0,05$	0,26±0,08 $P_3<0,01$
Фон НАД-ГДГ	3,16±0,42	3,67±0,40	5,24±0,90 $P_1<0,05$	3,04±0,40 $P_3<0,05$
Г6ФДГ	4,75±1,05	5,73±0,90	9,75±2,19 $P_1<0,05$	4,12±0,78 $P_3<0,01$
Г3ФДГ	6,38±1,09	6,38±1,22	11,65±3,15	4,27±1,00 $P_3<0,01$
ЛДГ	8,06±1,03	8,29±0,85	9,33±1,43	7,87±1,05
МДГ	14,77±1,59	14,53±1,40	17,02±2,80	13,53±1,64
НАДФ-МДГ	0,92±0,26	1,78±0,50	2,54±1,17 $P_1<0,05$	1,47±0,54
НАДФ-ГДГ	1,53±0,51	1,75±0,57	2,53±1,02	1,44±0,69
НАД-ГДГ	6,68±0,68	7,16±0,83	9,77±1,93	6,11±0,84 $P_3<0,05$
Обратная ЛДГ	38,90±9,10	35,60±9,80	75,30±28,40	19,69±6,9 $P_3<0,001$
Обратная МДГ	151,00±19,60	109,20±15,20	181,10±35,10	80,4±13,7 $P_3<0,01$
ГР	6,38±0,86	6,55±0,86	8,53±1,13	5,76±0,87

Таблица 10.

Иммунологические показатели крови у здоровых людей, лиц с иммунодефицитом и в группах, выделенных нейросетью

		Группы лиц с ИД		
Параметры	Здоровые (1) n=51	общая (2) n=42	оставшаяся (3) n=12	исключенная (4) n=30
Лейкоциты	5,97±0,34	6,38±0,42	7,72±1,11 P_1<0,05	5,85±0,38 P_3<0,05
Лимфоциты %	44,7±1,2	43,5±1,5	41,8±3,2	44,2±1,7
Лимфоциты абс.	2,60±0,14	2,80±0,18	3,28±0,37 P_1<0,05	2,60±0,20
т-РОК %	62,6±2,0	68,1±1,9	69,5±4,0	67,5±2,3
т-РОК абс.	1,74±0,12	1,91±0,13	2,24±0,21	1,78±0,15
ТФР-РОК %	40,9±1,7	45,3±1,9	39,8±3,4	47,3±2,3 P_1<0,05
ТФР-РОК абс.	1,11±0,09	1,26±0,11	1,23±0,23	1,27±0,13
ТФЧ-РОК %	24,0±1,5	24,6±2,3	32,1±5,5 P_1<0,05	21,7±2,3 P_3<0,05
ТФЧ-РОК абс.	0,67±0,07	0,68±0,08	0,95±0,16	0,58±0,08 P_3<0,05
р-РОК %	46,3±1,5	45,3±1,8	44,7±3,8	45,6±2,1
р-РОК абс.	1,29±0,1	1,28±0,10	1,49±0,22	1,19±0,11
с-РОК %	8,1±1,1	8,5±1,0	9,3±1,8	8,1±1,2
с-РОК абс.	0,23±0,03	0,28±0,04	0,39±0,09 P_1<0,05	0,23±0,04
ТФЧ / ТФР	2,07±0,17	2,94±0,4 P_1<0,05	1,68±0,38	3,42±0,52 P_1<0,01,P_3<0,05
IgA	2,17±0,45	1,54±0,08	1,56±0,13	1,54±0,10
IgM	0,57±0,04	0,67±0,09	0,92±0,26 P_1<0,05	0,57±0,06
IgG	8,97±0,41	8,80±0,37	9,43±0,59	8,53±0,47
IgE	15,76±3,03	15,34±3,47	11,82±3,72	16,74±4,65
ЦИК	35,68±5,29	31,43±4,71	34,55±8,05	30,00±6,12

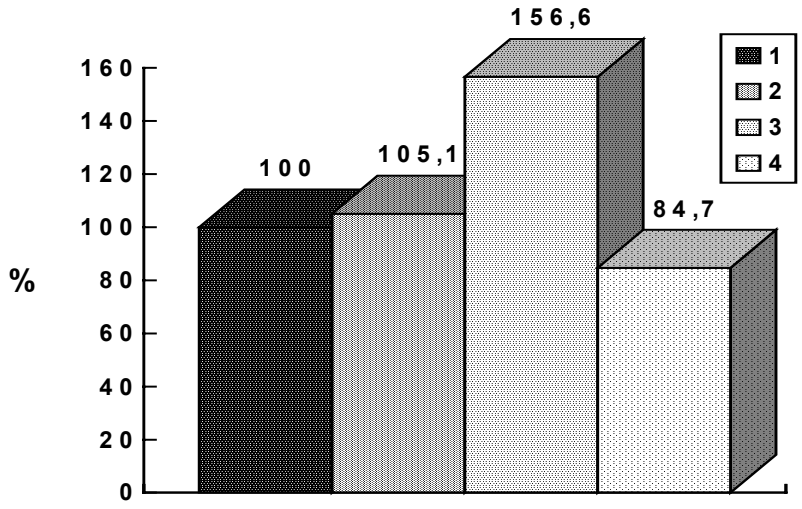


Рисунок 7. Иллюстрация сдвигов клеточного метаболизма лимфоцитов (приведены суммарные относительные величины) у больных с иммунодефицитными состояниями в сравнении со здоровыми людьми.

Уровень клеточного метаболизма лимфоцитов:

1. В группе здоровых людей (принят за 100%)
2. В группе больных иммунодефицитными состояниями (рассчитанный на всю группу)
3. В подгруппе больных, примеры которых остались в процессе обучения нейросети

В подгруппе больных, примеры которых были исключены в процессе обучения нейросети Исключенная группа достоверно отличалась от группы здоровых по 2 показателям иммунного статуса. Однако наибольшее количество достоверных различий наблюдалось между исключенной и оставшейся группами - 10 по параметрам клеточного метаболизма и 4 по параметрам иммунного статуса.

Анализ направлений параметров в полученных подгруппах показал противоположную направленность сдвигов некоторых из них по сравнению с группой здоровых. Наиболее яркие противонаправленные сдвиги наблюдались почти по всем параметрам активности внутрилимфоцитарных ферментов (Рис. 7). Интерес представляет также достоверный разнонаправленный сдвиг соотношения ТФР-РОК/ТФЧ-РОК, указывающий на различную направленность дифференцировки лимфоцитов в подгруппах, выделенных нейросетью.

Анализируя примеры больных, попавших в разные подгруппы, можно предположить, что нейросеть, используя заданные обучающие параметры, разделила общую выборку лиц с ИД на две группы с различными состояниями иммунореактивности.

В оставшейся группе при депрессии иммунного ответа наблюдается компенсаторная реакция иммунной системы, проявляющаяся прежде всего увеличением активности внутриклеточных ферментов.

В исключенных примерах большинство метаболических показателей приближается к аналогичным параметрам здоровых индивидуумов и, вероятно, наблюдается компенсаторная реакция, выражающаяся в активации иммунореактивности [57].

Таким образом, в данном случае нейронная сеть, обучавшись на неверно заданной классификационной модели (что сначала не было известно исследователям), стала источником гипотезы о разнородности одного из классов.

С учетом скорректированной классификационной модели были инициализированы 10 новых нейросетей-экспертов для классификации теперь уже всех 3 полученных групп. Нейросети имели параметры, аналогичные предыдущей. Все они полностью обучилась различать заданные 3 новых класса. При анализе значимости обучающих параметров самыми информативными оказались соотношение ТФР-РОК/ТФЧ-РОК и активности дегидрогеназ лимфоцитов.

Таким образом, новые 3-классовые нейросети накопили определенный опыт дифференцировки трех состояний иммунореактивности - характерного для здоровых людей (1 класс) и две фазы измененной иммунореактивности, характерной для состояний вторичного иммунодефицита (2 и 3 классы).

Литература

1. Парин В.В., Баевский Р.М. Медицина и техника.- М.: Знание, 1968.- С.36-49.

2. Переверзев-Орлов В.С. Советчик специалиста. Опыт разработки партнерской системы // М.: Наука, 1990.- 133 с.

3. Fu H.C., Shann J.J. A fuzzy neural network for knowledge learning // Int. J. Neural Syst.- 1994.- V.5, N.1.- P.13-22.

4. Масалович А.И. От нейрона к нейрокомпьютеру // Журнал доктора Добба.- 1992.- N.1.- С.20-24.

5. Stefanuk V.L. Expert systems and its applications // The lectures of Union's work-shop on the main problems of artificial intillegence and intellectual systems. Part 2, Minsk, 1990.- P.36-55.

6. Горбань А.Н., Фриденберг В.И. Новая игрушка человечества // МИР ПК, 1993, № 9.

7. Горбань А.Н. Нейрокомпьютер или аналоговый ренессанс // МИР ПК, 1994, № 10.

8. Rozenbojm J., Palladino E., Azevedo A.C. An expert clinical diagnosis system for the support of the primary consultation // Salud. Publica Mex.- 1993.- V.35, N.3.- P.321-325.

9. Poli R., Cagnoni S., Livi R. et al. A Neural Network Expert System for Diagnosing and Treating Hypertension // Computer.- 1991.- N.3.- P.64-71.

10. Gindi G.R., Darken C.J., O'Brien K.M. et al. Neural network and conventional classifiers for fluorescence-guided laser angioplasty // IEEE Trans. Biomed. Eng.- 1991.- V.38, N.3.- P.246-252.

11. Allen J., Murray A.. Development of a neural network screening aid for diagnosing lower limb peripheral vascular disease from photoelectric plethysmography pulse waveforms // Physiol. Meas.- 1993.- V.14, N.1.- P.13-22.

12. Astion M.L., Wener M.H., Thomas R.G., Hunder G.G., Bloch D.A. Application of neural networks to the classification of giant cell arteritis // Arthritis Reum.- 1994.- V.37, N.5.- P.760-770.

13. Baxt W.G. A neural network trained to identify the presence of myocardial infarction bases some decisions on clinical associations that differ from accepted clinical teaching // Med. Decis. Making.- 1994.- V.14, N.3.- P.217-222.

14. Baxt W.G. Complexity, chaos and human physiology: the justification for non-linear neural computational analysis // Cancer Lett.- 1994.- V.77, N.2-3.- P.85-93.

15. Baxt W.G. Use of an artificial neural network for the diagnosis of myocardial infarction // Ann. Intern. Med.- 1991.- V.115, N.11.- P.843-848.

16. Baxt W.G. Analysis of the clinical variables driving decision in an artificial neural network trained to identify the presence of myocardial infarction // Ann. Emerg. Med.- 1992.- V.21, N.12.- P.1439-1444.

17. Guo Z., Durand L.G., Lee H.C. et al. Artificial neural networks in computer-assisted classification of heart sounds in patients with porcine bioprosthetic valves // Med. Biol. Eng. Comput.- 1994.- V.32, N.3.- P.311-316.

18. Barschdorff D., Ester S., Dorsel T et al. Phonographic diagnostic aid in heart defects using neural networks // Biomed. Tech. Berlin.- 1990.- V.35, N.11.- P.271-279.

19. Okamoto Y., Nakano H., Yoshikawa M. et al. Study on decision support system for the interpretation of laboratory data by an artificial neural network // Rinsho. Byori.- 1994.- V.42, N.2.- P.195-199.

20. Maclin P.S., Dempsey J. Using an artificial neural network to diagnose hepatic masses // J. Med. Syst.- 1992.- V.16, N.5.- P.215-225.

21. Rinast E., Linder R., Weiss H.D. Neural network approach for computer-assisted interpretation of ultrasound images of the gallbladder // Eur. J. Radiol.- 1993.- V.17, N.3.- P.175-178.

22. Modai I., Stoler M., Inbar-Saban N. et al. Clinical decisions for psychiatric inpatients and their evaluation by a trained neural network // Methods Inf. Med.- 1993.- V.32, N.5.- P.396-399.

23. Ercal F., Chawla A., Stoeker W.V. et al. Neural network diagnosis of malignant melanoma from color images // IEEE Trans. Biomed. Eng.- 1994.- V.41, N.9.- P.837-845.

24. Lee H.-L., Suzuki S., Adachi Y. et al. Fuzzy Theory in Traditional Chinese Pulse Diagnosis // Proceedings of 1993 International Joint Conference on Neural Networks, Nagoya, Japan, October 25-29, 1993.- Nagoya, 1993.- V.1.- P.774-777.

25. Yang T.-F., Devine B., Macfarlane P.W. Combination of artificial neural networks and deterministic logic in the electrocardiogram diagnosis of inferior myocardial infarction // Eur. Heart J.- 1994.- V.15.- Abstr. Supplement XII-th World Congress Cardiology (2408).- P.449.

26. Hoher M., Kestler H.A., Palm G. et al. Neural network based QRS classification of the signal averaged electrocardiogram // Eur. Heart J.- 1994.- V.15.- Abstr. Supplement XII-th World Congress Cardiology (734).- P.114.

27. Nakajima H., Anbe J., Egoh Y. et al. Evaluation of neural network rate regulation system in dual activity sensor rate adaptive pacer // European Journal of Cardiac Pacing and Electrophysiology.- Abstracts of 9th International Congress, Nice Acropolis - French, Rivera, June 15-18, (228), 1994.- Rivera, 1994.- P.54.

28. Gross G.W., Boone J.M., Greco-Hunt V. et al. Neural networks in radiologic diagnosis. II. Interpretation of neonatal chest radiographs // Invest. Radiol.- 1990.- V.25, N.9.- P.1017-1023.

29. Floyd C.E.Jr., Lo J.Y., Yun A.J. et al. Prediction of breast cancer malignancy using an artificial neural network // Cancer.- 1994.- V.74, N.11.- P.2944-2948.

30. Reinbnerger G., Weiss G., Werner-Felmayer G. et al. Neural networks as a tool for utilizing laboratory information: comparison with linear discriminant analysis and with classification and regression trees // Proc. Natl. Acad. Sci., USA.- 1991.- V.88, N.24.- P.11426-11430.

31. Шварц Э., Трис Д. Программы, умеющие думать // Бизнес Уик.- 1992.- N.6.- С.15-18.

32. Aynsley M., Hofland A., Morris A.J. et al. Artificial intelligence and the supervision of bioprocesses (real-time knowledge-based systems and neural networks) // Adv. Biochem. Eng. Biotechnol.- 1993.- N.48.- P.1-27.

33. Budilova E.V., Teriokhin A.T. Endocrine networks // The RNNS/IEEE Symposium on Neuroinformatics and Neurocomputers, Rostov-on-Don, Russia, October 7-10, 1992.- Rostov/Don, 1992.- V.2.- P.729-737.

34. Varela F.J., Coutinho A., Dupire B. et al. Cognitive networks: immune, neural and otherwise // Teoretical immunology. Ed. by Perelson A.- Addison Wesley, 1988.- Part 2.- P.359-375.

35. Levine D.S., Parks R.W., Prueitt P.S. Methodological and theoretical issues in neural network models of frontal cognitive functions // Int. J. Neurosci.- 1993.- V.72, N.3-4.- P.209-233.

36. Van Leeuwen J.L. Neural network simulations of the nervous system // Eur. J. Morphol.- 1990.- V.28, N.2-4.- P.139-147.

37. Senna A.L., Junior W.M., Carvallo M.L.B., Siqueira A.M. Neural Networks in Biological Taxonomy // Proceedings of 1993 International Joint Conference on Neural Networks, Nagoya, Japan, October 25-29, 1993.- Nagoya, 1993.- V.1.- P.33-36.

38. Sweeney J.W.P., Musavi M.T., Guidi J.N. Probabilistic Neural Network as Chromosome Classifier // Proceedings of 1993 International Joint Conference on Neural Networks, Nagoya, Japan, October 25-29, 1993.- Nagoya, 1993.-V.1.- P.935-938.

39. Korver M., Lucas P.J. Converting a rule-based expert system into a belief network // Med. Inf. Lond.- 1993.- V.18, N.3.- P.219-241.

40. Gorban A.N., Rossiev D.A., Gilev S.E. et al. “NeuroComp” group: neural-networks software and its application // Russian Academy of Sciences, Krasnoyarsk Computing Center, Preprint N 8.- Krasnoyarsk, 1995.- 38 p.

41. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере // Новосибирск: Наука, 1996.- 276 с.

42. Горбань А.Н., Россиев Д.А., Коченов Д.А. Применение самообучающихся нейросетевых программ. Раздел 1. Учебно-методическое пособие для студентов специальностей 22.04 и 55.28.00 всех форм обучения // Красноярск: Сибирский Технологический Институт, 1994.- 24 с.

43. Гилева Л.В., Гилев С.Е., Горбань А.Н., Гордиенко П.В., Еремин Д.И., Коченов Д.А., Миркес Е.М., Россиев Д.А., Умнов Н.А**.** Нейропрограммы. Учебное

пособие: В 2 ч. // Красноярск, Красноярский государственный технический университет, 1994.- 260 с.

44. Gorban A.N., Rossiev D.A., Gilev S.E. et al. Medical and physiological applications of MultiNeuron neural simulator // Proceedings of World Congress on Neural Networks-1995 (WCNN'95).- Washington, 1995.- V.1, P.170-175.

45. Горбань А.Н. Обучение нейронных сетей. М.: Параграф, 1990.- 160 с.

46. Гилев С.Е., Горбань А.Н., Миркес Е.М. Малые эксперты и внутренние конфликты в обучаемых нейронных сетях // Доклады Академии Наук СССР.- 1991.- Т.320, N.1.- С.220-223.

47. Гилев С.Е., Коченов Д.А., Миркес Е.М., Россиев Д.А. Контрастирование, оценка значимости параметров, оптимизация их значений и их интерпретация в нейронных сетях // Доклады III Всероссийского семинара "Нейроинформатика и ее приложения".- Красноярск, 1995.- С.66-78.

48. Коченов Д.А., Миркес Е.М., Россиев Д.А. Автоматическая подстройка входных данных для получения требуемого ответа нейросети // Проблемы информатизации региона. Труды межрегиональной конференции (Красноярск, 27-29 ноября 1995 г.). Красноярск, 1995.- С.156.

49. Rossiev D.A., Golovenkin S.E., Shulman V.A., Matjushin G.V. Neural networks for forecasting of myocardial infarction complications // The Second International Symposium on Neuroinformatics and Neurocomputers, Rostov-on-Don, Russia, September 20-23, 1995.- Rostov-on-Don, 1995.- P.292-298.

50. Мызников А.В., Россиев Д.А., Лохман В.Ф. Нейросетевая экспертная система для оптимизации лечения облитерирующего тромбангиита и прогнозирования его непосредственных исходов // Ангиология и сосудистая хирургия.- 1995.- N 2.- С.100.

51. Лебедев К.А., Понякина И.Д. Иммунограмма в клинической практике. - М.: Наука, 1990.- 224 с.

52. Подосинников И.С., Чухловина М.Л. Метаболизм и функция лимфоцитов при первичных иммунодефицитных состояниях // Иммунология.- 1986.- N.4.- С.30-38.

53. Rossiev D.A., Savchenko A.A., Borisov A.G., Kochenov D.A. The employment of neural-network classifier for diagnostics of different phases of immunodeficiency // Modelling, Measurement & Control.- 1994.- V.42.- N.2. P.55-63.

54. Савченко А.А., Сунцова Л.Н. Высокочувствительное определение активности дегидрогеназ в лимфоцитах периферической крови человека биолюминесцентным методом // Лаб. дело.- 1989.- N.11.- С.23-25.

55. Лозовой В.П., Кожевников В.С., Волчек И.А. и др. Методы исследования Т-системы иммунитета в диагностике вторичных иммунодефицитов при заболеваниях и повреждениях.- Томск, 1986. 18 с.

56. Кожевников В.С., Волчек И.А. Функциональная активность субклассов Т-лимфоцитов и их роль в патогенезе вторичных иммунодефицитов // Функциональные характеристики циркулирующего пула иммунокомпетентных клеток: Сб. науч. тр. - Новосибирск, 1984.- С.11-17.

57. Лазарев И.А., Русаков В.И., Ткачева Т.Н. Иммунопатологические изменения у больных язвенной болезнью желудка и двенадцатиперстной кишки и их иммунокоррекция // 4 Всесоюзный Съезд гастроэнтерологов.: Тез. докл. - Москва, Ленинград, 1990.- Т.1.- С.702-703.

Глава 6.

Погрешности в нейронных сетях

А.Н. Горбань, М.Ю. Сенашова

ВЦК СО РАН, КГУ

Рассматриваются нейронные сети слоистой структуры, состоящие из слоев стандартных нейронов. Изучаются ошибки, возникающие при технической реализации сетей, а также при шумах и повреждениях.

Определены максимально допустимые погрешности, возможные для сигналов и параметров каждого элемента сети, исходя из условия, что вектор выходных сигналов сети должен вычисляться с заданной точностью. Используются два типа оценок погрешности: гарантированные интервальные оценки и среднеквадратические оценки погрешностей.

Показано, что оценки допустимых погрешностей можно получить в ходе специального процесса “обратного распространения точности”. Он состоит в функционировании сети с той же системой связей, но от выходов к входам и с заменой элементов на двойственные. Эта двойственность принципиально отличается от той, которая используется в классическом методе вычисления градиентов оценки с помощью обратного распространения ошибок (back propagation of errors).

С помощью полученных результатов объясняется наблюдаемая высокая устойчивость нейронных сетей к шумам и разрушениям.

1. Введение

В настоящее время существуют различные технические реализации нейронных сетей, в том числе нейроимитаторы, то есть компьютерные модели нейронных сетей. Нейроимитаторы являются гибкими средствами для изучения сетей и работы с ними. С нейроимитаторами можно выполнять различные

операции – обучать, определять наиболее и наименее значимые связи, контрастировать, то есть удалять наименее значимые связи и т. д.

Существует подход, получающий все большее распространение, при котором сначала конструируется и обучается нейроимитатор, а затем создается техническая реализация полученной нейросети с уже вычисленными весами синапсов.

Нейроимитатор, работающий на универсальных цифровых ЭВМ, позволяет вычислять веса синапсов с большой точностью, которую трудно получить при других технических реализациях сети (в первую очередь – аналоговых) в силу ограниченной точности технических устройств. Поэтому возникает задача приведения весов синапсов к некоторому набору конкретных значений. Ее частный случай - задача бинаризации, то есть задача приведения весов синапсов к значениям 0 или 1 (связь либо есть, либо нет - без всяких весов синапсов).

При аналоговых реализациях, различных упрощениях архитектуры (в том числе - бинаризации) технически сложно получить результат работы сети той же точности, что и результат работы нейроимитатора [3-5]. Поэтому следует ограничится некоторой точностью, с которой может работать сеть, то есть выбрать интервал, в котором могут изменяться значения вектора выходных сигналов сети.

Оценка погрешностей сигналов сети очень полезна при решении задачи упрощения нейронной сети. Зная допустимую погрешность выходного сигнала какого-либо элемента сети, мы можем заменять его более простыми, но менее точными элементами так, чтобы в итоге ошибка не превышала заданную.

Хорошо известно, что нейронные сети могут проявлять исключительную устойчивость к помехам и разрушениям. Иногда эти эффекты называют голографическими свойствами нейронных сетей, подразумевая, что полезные навыки распределены по сети примерно так же, как изображение - по голографической пластинке, и могут сохраняться при значительных разрушениях.

Как будет показано ниже, при прямом прохождении сигналов по достаточно большой сети погрешности гасятся: при больших погрешностях входных сигналов выходные сигналы сети могут иметь сравнительно малые погрешности. Это объясняет устойчивость нейронных сетей к шумам и повреждениям.

2. Два базовых подхода к оценкам погрешности

Рассмотрим два подхода к решению задачи вычисления погрешностей сигналов сети. При первом подходе (*гарантированные интервальные оценки*) вычисляются допустимые интервалы для погрешностей сигналов сети такие, что погрешность вектора выходных сигналов гарантированно (с вероятностью 1) не превышает заданную. При втором подходе (*среднеквадратические оценки погрешностей*) вычисляются среднеквадратические отклонения погрешностей сигналов. При этом часто используется предположение о том, что погрешности различных сигналов являются независимыми случайными величинами.

Существует принципиальное различие между этими двумя типами оценок. Гарантированные интервальные оценки исходят из рассмотрения наихудших возможных случаев, сколь бы малой не была их вероятность. Поэтому они, как правило, завышают реально имеющую место ошибку и слишком пессимистичны с практической точки зрения. Среднеквадратичные оценки, наоборот, стирают возможные большие уклонения и могут оказаться слишком оптимистичными.

Важное различие между двумя типами оценок демонстрируют следующие *формулы сложения.*

1. *Формула сложения для интервальных оценок*. Пусть для двух величин *x, y* определены гарантированные интервалы значений $x=x_0\pm\Delta_x$, $y=y_0\pm\Delta_y$. Тогда для их суммы имеем гарантированную оценку: $x+y= x_0+y_0 \pm(\Delta_x+\Delta_y)$, то есть $\Delta_{x+y}=\Delta_x+\Delta_y$.

2. *Формула сложения для среднеквадратичных уклонений.* Пусть для двух *независимых* величин *x, y* определены среднеквадратичные уклонения σ_x , σ_y . Тогда $\sigma_{x+y}= (\sigma_x^2 +\sigma_y^2)^{1/2}$.

3. Структура сети

Предполагаем, что сеть имеет слоистую структуру. Это самоограничение позволит несколько сократить изложение, но не влияет на общность рассмотрения - исследование любой сети может быть формально сведено к изучению слоистых сетей.

Сеть слоистой структуры состоит из слоев стандартных нейронов, связанных между собой синапсами с весами, полученными при обучении. Причем сигналы передаются только в одном направлении, с предыдущего слоя на следующий. Под стандартным нейроном [1,2] понимается набор элементов, состоящий из адаптивного сумматора, нелинейного преобразователя и точки ветвления (рис.1). Точка ветвления – это элемент, посылающий выходной сигнал нелинейного преобразователя на вход нескольких стандартных нейронов следующего слоя.

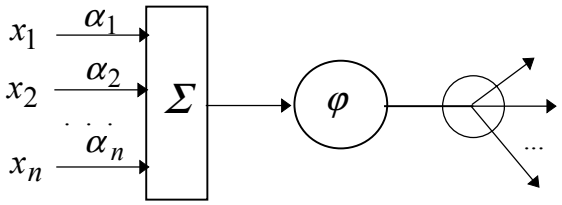


Рис.1. Стандартный нейрон.
Σ– адаптивный сумматор, φ – нелинейный преобразователь и точка ветвления.

Так как мы имеем дело с сетями слоистой структуры, состоящими из слоев стандартных нейронов, то выходные сигналы одного слоя являются входными сигналами другого слоя. В свою очередь, внутри самого стандартного нейрона выходной сигнал одного элемента (например, сумматора) является входным сигналом другого элемента (например, нелинейного преобразователя). Таким образом, можно проследить, начиная с выходных сигналов сети, от какого элемента сети пришел сигнал к данному элементу.

Стандартный нейрон является типичным участком любой нейронной сети. Поэтому достаточно выяснить, как вычисляются допустимые погрешности для элементов стандартного нейрона. В результате получим возможность вычислить допустимые погрешности для любого участка сети, двигаясь по сети от нейрона к нейрону.

4. Метод обратного распространения точности для гарантированных интервальных оценок

Пусть нам заданы допустимые погрешности вычислений для выходных сигналов сети. Для каждого элемента решим задачу: определить допустимые погрешности на входах элемента по заданным максимально допустимым погрешностям на его выходе. Если эту задачу решить для каждого элемента сети, то можно оценить допустимые погрешности для всех сигналов, проходящих через сеть, переходя по сети от элемента к элементу в обратном направлении (от выходов сети к ее входам). Этот процесс мы назовем ***обратным распространением точности.*** В ходе него движение сигналов происходит от выходов ко входам, сигнал, проходящий по связи в обратном направлении, является допустимой погрешностью сигнала, проходящего по этой связи в прямом направлении.

Последним элементом стандартного нейрона является точка ветвления, поэтому начинаем рассмотрение метода обратного распространения точности именно с нее.

Точка ветвления имеет несколько выходов. Пусть для каждого ее выхода задана допустимая погрешность ε_i (i - номер выхода). Для того, чтобы удовлетворить всем этим ограничениям погрешности, необходимо и достаточно, чтобы входной сигнал точки ветвления имел погрешность $\varepsilon = \min\{\varepsilon_i\}_{i=1}^{k}$. Таким образом, при обратном распространении точности тока ветвления заменяется на двойственный элемент, выбирающий из поступающих сигналов ε_i (т.е. погрешностей) минимальный.

Следующим элементом стандартного нейрона является нелинейный преобразователь. Пусть входной сигнал нелинейного преобразователя равен A_0, φ – его функция активации, $y = \varphi(A_0)$ – выходной сигнал и ε_1 – допустимая погрешность выходного сигнала. Вычислим максимальную погрешность ε входного сигнала нелинейного преобразователя, то есть найдем отрезок $[A_0 - \varepsilon, A_0 + \varepsilon]$ такой, что для любого $x \in [A_0 - \varepsilon, A_0 + \varepsilon]$ $\tilde{y} = \varphi(x)$ отличается от $y = \varphi(A_0)$ не более, чем на $\varepsilon_1 : |y - \tilde{y}| \le \varepsilon_1$.

Ввиду непрерывности и дифференцируемости функции активации нелинейного преобразователя очевидно, что $\varepsilon \le \varepsilon_1 / max|\varphi'(x)|$, где $x \in [\varphi^{-1}(y - \varepsilon_1), \varphi^{-1}(y + \varepsilon_1)]$.

Пойдем традиционным путем, оценивая допустимую погрешность в линейном приближении: $\varphi(A_0 \pm \varepsilon) \approx \varphi(A_0) \pm \varphi'(A_0) \cdot \varepsilon$. По условию

$$\varepsilon_1 \ge |\varphi(A_0 \pm \varepsilon) - \varphi(A_0)| \approx |\varphi'(A_0) \cdot \varepsilon|.$$

Пользуясь этим неравенством, подберем ε следующим образом: $\varepsilon \le \varepsilon_1 / |\varphi'(A_0)|$. В этом случае формула для вычисления допустимой погрешности более простая, но менее точная.

Получили погрешность, допустимую для входного сигнала нелинейного преобразователя, которая одновременно является допустимой погрешностью для выходного сигнала сумматора. Аналогично можем вычислить погрешность входного сигнала нелинейного преобразователя любого стандартного нейрона, если известна погрешность его выходного сигнала.

Двойственный к нелинейному преобразователю элемент - просто линейная связь! Ее вес равен $1/|\varphi'(A_0)|$ для линейного приближения в формуле ошибки или $1 / max|\varphi'(x)|$ - в более общем случае (в последней формуле максимум берется по отрезку $x \in [\varphi^{-1}(y - \varepsilon_1), \varphi^{-1}(y + \varepsilon_1)]$ - так что линейность здесь уже кажущаяся).

Перейдем к следующему элементу стандартного нейрона – адаптивному сумматору с n синапсами, являющимися его входами. Адаптивный сумматор – это сумматор, в котором входные сигналы x_i суммируются с весами α_i.

Каждый вход x_i сумматора Σ_0 имеет некоторую погрешность ε_i, которая вносит свой вклад в допустимую погрешность выходного сигнала сумматора. Эти погрешности могут иметь различные величины в зависимости от того, какой способ распределения допустимой погрешности выходного сигнала по входам сумматора мы выберем. Погрешности по входам сумматора могут распределяться равномерно, пропорционально и приоритетно.

Рассмотрим сначала равномерное распределение. Для этого полагаем, что на всех входах погрешности равны между собой $(\varepsilon_1 = \varepsilon_i, i = 2,\ldots,n)$.Пусть $A_0 = \sum_{i=1}^{n} \alpha_i \cdot x_i$ – выходной сигнал сумматора без погрешностей. Тогда $\{A'_0\}$ – множество выходных сигналов сумматора, получающихся, когда вектор входных сигналов сумматора пробегает вершины n-мерного куба с центром в точке $(x_1, x_2, \ldots, x_n)$ и ребром длины $2\varepsilon_1$: $\{A'_0\} = \sum_{i=1}^{n} \alpha_i \cdot (x_i \pm \varepsilon_i) = \sum_{i=1}^{n} \alpha_i \cdot x_i + \sum_{i=1}^{n} \alpha_i \cdot (\pm\varepsilon_i) = A_0 + \sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i \cdot z_i$, где $z_i \in \{-1,1\}$. Нам требуется, чтобы все множество значений $\{A'_0\}$ попало в интервал $[A_0 - \varepsilon, A_0 + \varepsilon]$. Для этого необходимо, чтобы $\max|\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i \cdot z_i| = \sum_{i=1}^{n} |\alpha_i| \cdot \varepsilon_i \le \varepsilon$, где максимум берется по всем z_i. Из этого неравенства и сделанного выше предположения о ε_i получаем требуемую оценку для равномерного распределения ε_i по входам сумматора: $\varepsilon_i \le \varepsilon / \sum_{i=1}^{n} |\alpha_i|$.

При пропорциональном распределении погрешностей допустимая погрешность выходного сигнала сумматора делится сначала на число входов, а затем для каждого входа делится на соответствующий вес синапса. То есть

погрешности распределяются пропорционально весам соответствующих синапсов. Формула расчета допустимой погрешности для каждого входа сумматора имеет вид: $\varepsilon_i = \varepsilon / (n \cdot \alpha_i)$, где ε – допустимая погрешность выходного сигнала сумматора, n – число входов сумматора, α_i – веса синапсов соответствующих входов сумматора.

При приоритетном распределении погрешностей сначала назначаются погрешности для тех входов, которые наиболее значимы по какому-либо признаку, а затем оставшуюся часть допустимой погрешности выходного сигнала сумматора распределяют между оставшимися входами равномерно или пропорционально.

Аналогично можно вычислить допустимые погрешности для входных сигналов сумматора любого стандартного нейрона, если известны погрешности для выходного сигнала сумматора.

Для адаптивного сумматора можно вычислять как допустимые погрешности входных сигналов сумматора, так и допустимые погрешности весов синапсов. Для вычисления допустимых погрешностей весов синапсов также можно использовать равномерное, пропорциональное и приоритетное распределение погрешностей. При равномерном распределении допустимые погрешности для весов синапсов вычисляются по формуле: $\varepsilon_i \le \varepsilon / \sum_{i=1}^{n} |x_i|$, где x_i – входные сигналы сумматора.

При пропорциональном распределении допустимые погрешности для весов синапсов вычисляются по формуле: $\varepsilon_i \le \varepsilon / (n \cdot x_i)$, где n – число входов сумматора, x_i – входные сигналы сумматора.

При приоритетном распределении сначала назначаются допустимые погрешности для тех весов синапсов, которые наиболее значимы по какому-либо признаку, а затем оставшуюся часть допустимой погрешности для выходного сигнала сумматора распределяют между оставшимися весами синапсов равномерно или пропорционально.

При обратном распространении точности имеет место специфическая двойственность - элементы сети заменяются на двойственные им. Однако, эта двойственность отличается от той, с которой мы встречаемся при изучении обратного распространения ошибки для вычисления градиентов функции оценки. Так, если в обычном обратном распространении двойственным элементом к точке ветвления является простой сумматор, то при обратном распространении точности вместо него, как было показано, появляется элемент, вычисляющий минимум приходящих на него сигналов. Нелинейный преобразователь при обратном распространении точности заменяется двойственным ему элементом, умножающим сигнал на число. Но если при обратном распространении ошибки множителем является значение градиента, то в нашем случае сигнал умножается на величину обратную производной от входного сигнала нелинейного преобразователя. Адаптивный сумматор также заменяется двойственным ему элементом. Этот элемент является своеобразной точкой ветвления. Но, в отличии от простой точки ветвления, он сначала преобразует приходящий к нему сигнал в соответствии с выбранным распределением погрешностей по входам адаптивного сумматора, а затем передает полученные сигналы дальше.

Теперь мы знаем, каким образом вычислять гарантированную интервальную оценку погрешности для любого элемента стандартного нейрона методом обратного распространения точности.

1) ***Точка ветвления***. Если допустимые погрешности выходных сигналов точки ветвления равны $\varepsilon_1, \varepsilon_2, ..., \varepsilon_k$, то в качестве погрешности входного сигнала точки ветвления выбирается $\min\{\varepsilon_i\}_{i=1}^{k}$ (рис.2).

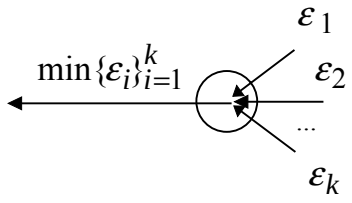


Рис.2.

2) ***Нелинейный преобразователь.*** Пусть при прямом функционировании входной сигнал нелинейного преобразователя равен A, его выходной сигнал равен y и нелинейный преобразователь имеет функцию активации φ. Если допустимая погрешность выходного сигнала нелинейного преобразователя равняется ε, то погрешность его входного сигнала не должна превышать $\varepsilon / \max|\varphi'(x)|$, где $x \in [\varphi^{-1}(y-\varepsilon), \varphi^{-1}(y+\varepsilon)]$ или в линейном приближении $\varepsilon/|\varphi'(A)|$ (рис.3).

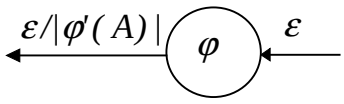
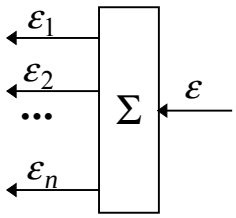


Рис.3.

3) ***Адаптивный сумматор***. Если при обратном распространении допустимая погрешность выходного сигнала адаптивного сумматора равняется ε, то погрешность каждого входа сумматора не должна превышать ε_i, где

$\varepsilon_i \le \varepsilon / \Sigma_{i=1}^{n} |\alpha_i|$ для равномерного распределения и $\varepsilon_i = \varepsilon / (n \cdot \alpha_i)$ для пропорционального распределения (рис.4).



1) для равномерного распределения

$$\varepsilon_i = \varepsilon / \sum_{i=1}^{n} |\alpha_i|$$

2) для пропорционального распределения

$$\varepsilon_i = \varepsilon / (n \cdot \alpha_i)$$

Рис.4.

Зная, как вычисляются допустимые погрешности для всех элементов стандартного нейрона, можно вычислить допустимые погрешности сигналов для всей сети. Рассмотрим участок сети, состоящий из сумматора Σ_0 и нелинейного преобразователя, результатом работы которого является выходной сигнал y, а также из сумматоров Σ_i и нелинейных преобразователей, выходные сигналы которых являются входными сигналами сумматора Σ_0 (рис.5). То есть мы рассматриваем два последних слоя нейронной сети, состоящие из стандартных нейронов.

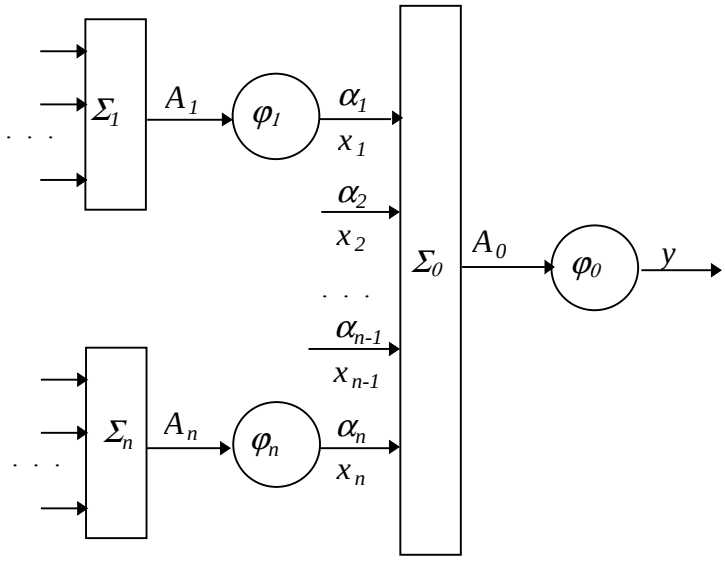


Рис.5 α_i – веса синапсов, x_i – сигналы сети,
φ_i – нелинейные преобразователи, Σ_i – сумматоры,
A_i – выходные сигналы сумматоров

Если заданы допустимые погрешности для выходных сигналов сети, можно вычислить допустимые погрешности для последнего слоя сети. Когда вычислены допустимые погрешности всех входных сигналов последнего слоя сети, переходим к вычислению допустимых погрешностей предпоследнего слоя и так далее. Переходя по сети в обратном направлении от слоя к слою, мы можем вычислить допустимые погрешности всех сигналов сети, в том числе допустимые погрешности входных сигналов.

Мы рассмотрели, как изменяются погрешности сигналов при прохождении через элементы сети. Предположим теперь, что не только сигналы имеют погрешности, но и все элементы сети передают приходящие к ним сигналы с некоторыми погрешностями. Пусть собственные погрешности элементов известны и фиксированы. Выясним, как влияют собственные погрешности элементов на погрешности сигналов.

Выясним, как действуют элементы сети, имеющие собственные погрешности, при прямой работе сети.

Точка ветвления может либо вообще не иметь погрешности, либо она имеет собственную погрешность ε_{tv}. В последнем случае сигнал x при

прохождении через точку ветвления будет изменяться, оставаясь в интервале $x \pm \varepsilon_{tv}$ (рис.6).

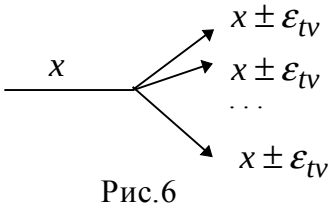


Рис.6

Предположим, что сумматор имеет собственную погрешность ε_Σ. Тогда возможны следующие варианты:

1) погрешность прибавляется к выходному сигналу сумматора, т.е. при прохождении сигналов x_i через сумматор выходной сигнал сумматора будет иметь вид: $\sum_{i=1}^{n} \alpha_i \cdot x_i \pm \varepsilon_\Sigma$;

2) погрешность сумматора действует по каждому входу пропорционально $\sum_{i=1}^{n} \alpha_i \cdot (x_i \pm \varepsilon_\Sigma^i)$ (рис.7).

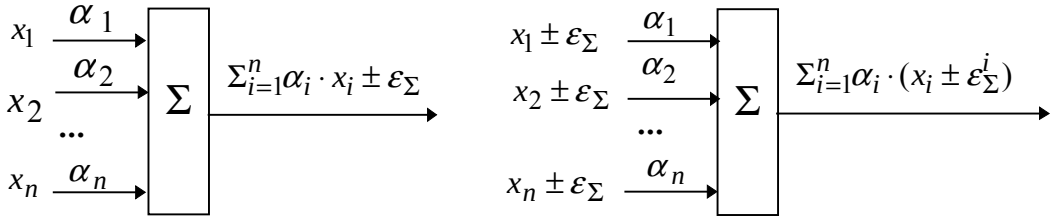


Рис.7

Считаем при этом, что погрешности ε_Σ^i равны между собой и равны ε_Σ / n, где n – число входов сумматора.

Пусть собственная погрешность нелинейного преобразователя равна ε_φ, x – входной сигнал нелинейного преобразователя, φ – его функция активации. Собственная погрешность может добавляться или к входному сигналу x:

$\varphi(x \pm \varepsilon_\varphi)$, или к выходному сигналу нелинейного преобразователя: $\varphi(x) \pm \varepsilon_\varphi$ (рис.8).

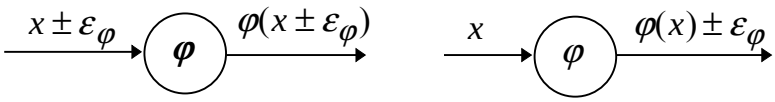


Рис.8

Мы выяснили как вычисляются допустимые погрешности сигналов сети. При этом мы не выделяли особо тот вклад, который вносят в погрешность сигнала сами элементы. Рассмотрим теперь, как вычисляются допустимые погрешности сигналов сети при обратном распространении точности с учетом собственных погрешностей элементов стандартного нейрона.

Начнем вычисление допустимых погрешностей сигналов сети с учетом собственных погрешностей элементов с точки ветвления. Пусть точка ветвления имеет собственную погрешность ε_{tv}. Предположим, что допустимые погрешности выходных сигналов точки ветвления равны $\varepsilon_1 \pm \varepsilon_{tv}, \varepsilon_2 \pm \varepsilon_{tv}, ..., \varepsilon_k \pm \varepsilon_{tv}$. Для увеличения точности вычислений необходимо накладывать на допустимые погрешности наиболее жесткие требования. Поэтому в качестве допустимой погрешности входного сигнала точки ветвления при обратном распространении следует выбирать погрешность $\min\{\varepsilon_i - \varepsilon_{tv}\}_{i=1}^{k}$.

Следующий элемент стандартного нейрона – нелинейный преобразователь. Если нелинейный преобразователь имеет собственную погрешность ε_φ, которая добавляется к его выходному сигналу, и погрешность его выходного сигнала равняется ε_1, то допустимая погрешность входного сигнала нелинейного преобразователя равняется $\varepsilon \le (\varepsilon_1 - \varepsilon_\varphi) / max|\varphi'(x)|$, где $x \in \left[\varphi^{-1}(y - \varepsilon_1 + \varepsilon_\varphi), \varphi^{-1}(y + \varepsilon_1 - \varepsilon_\varphi)\right]$ или в линейном приближении $\varepsilon \le (\varepsilon_1 - \varepsilon_\varphi) / |\varphi'(A_0)|$.

Предположим теперь, что собственная погрешность нелинейного преобразователя ε_φ добавляется к его входному сигналу $\varphi(x \pm \varepsilon_\varphi)$, и при обратном распространении точности погрешность выходного сигнала нелинейного преобразователя равняется ε_1. Рассмотрим наихудший вариант, когда входной сигнал нелинейного преобразователя находится в интервале $\left[x - \varepsilon - \varepsilon_\varphi, x + \varepsilon + \varepsilon_\varphi\right]$. В этом случае допустимая погрешность входного сигнала нелинейного преобразователя вычисляется следующим образом:

$\varepsilon \le \varepsilon_1 / |\varphi'(x)| - \varepsilon_\varphi$, где $x \in \left[\varphi^{-1}(y - \varepsilon_1) - \varepsilon_\varphi, \varphi^{-1}(y + \varepsilon_1) + \varepsilon_\varphi\right]$.

Рассмотрим допустимую погрешность в линейном приближении:
$\varphi(A_0 \pm (\varepsilon + \varepsilon_\varphi)) \approx \varphi(A_0) \pm \varphi'(A_0) \cdot (\varepsilon + \varepsilon_\varphi)$. По условию

$$\varepsilon_1 \ge |\varphi(A_0 \pm (\varepsilon + \varepsilon_\varphi)) - \varphi(A_0)| \approx |\varphi'(A_0) \cdot (\varepsilon + \varepsilon_\varphi)|.$$

Получаем: $(\varepsilon + \varepsilon_\varphi) \le \varepsilon_1 / |\varphi'(A_0)|$ или $\varepsilon \le \varepsilon_1 / |\varphi'(A_0)| - \varepsilon_\varphi$.

И, наконец, перейдем к вычислению допустимых погрешностей входных сигналов сумматора. Рассмотрим вариант, при котором собственная погрешность сумматора ε_Σ добавляется к его выходному сигналу, и допустимая погрешность выходного сигнала сумматора равняется ε. При обратном распространении точности получаем, что равномерно, пропорционально и приоритетно по выше полученным формулам распределяется погрешность $\varepsilon - \varepsilon_\Sigma$.

Если же собственная погрешность сумматора пропорционально распределяется по его входам, и допустимая погрешность выходного сигнала сумматора равняется ε, то допустимые погрешности для входов сумматора вычисляются следующим образом. Пусть $A_0 = \sum_{i=1}^{n} \alpha_i \cdot x_i$ –выходной сигнал сумматора без погрешностей. Тогда $\{A'_0\}$ – выходные сигналы сумматора с учетом собственных погрешностей сумматора ε_Σ^i и погрешностей входных сигналов ε_i:

$$\{A'_0\} = \sum_{i=1}^{n} \alpha_i \cdot (x_i \pm \varepsilon_i \pm \varepsilon_\Sigma^i) = \sum_{i=1}^{n} \alpha_i \cdot x_i + \sum_{i=1}^{n} \alpha_i \cdot (\pm\varepsilon_i) + \sum_{i=1}^{n} \alpha_i \cdot (\pm\varepsilon_\Sigma^i) =$$

$$= A_0 + \sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i \cdot z_i + \sum_{i=1}^{n} \alpha_i \cdot \varepsilon_\Sigma^i \cdot z_i,$$

где $z_i \in \{-1, 1\}$. Для того, чтобы все множество $\{A'_0\}$ попало в интервал $[A_0 - \varepsilon, A_0 + \varepsilon]$ необходимо, чтобы

$$\max \left| \sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i \cdot z_i + \sum_{i=1}^{n} \alpha_i \cdot \varepsilon_\Sigma^i \cdot z_i \right| = \sum_{i=1}^{n} |\alpha_i| \cdot \varepsilon_i + \sum_{i=1}^{n} |\alpha_i| \cdot \varepsilon_\Sigma^i \le \varepsilon,$$

где максимум берется по всем z_i. Из этого неравенства, предполагая что ε_i равны между собой, получаем требуемую оценку для ε_i:

$$\varepsilon_i \leq (\varepsilon - \varepsilon_{\Sigma}^{i} \cdot \sum_{i=1}^{n} |\alpha_i|) / \sum_{i=1}^{n} |\alpha_i| .$$

Мы получили формулы для вычисления допустимых погрешностей сигналов для любого участка сети с учетом того, что все элементы имеют собственные погрешности, которые вносят свой вклад в погрешность выходного сигнала этих элементов.

5. Метод обратного распространения точности для анализа реализуемости сетей с собственными погрешностями элементов

Ранее был рассмотрен метод обратного распространения точности с учетом собственных погрешностей элементов сети. Но этот метод может быть использован не для всех сетей. При применении метода может возникнуть ситуация, когда собственная погрешность элемента превышает погрешность сигнала, который должен выходить из этого элемента. Например, если собственная погрешность нейрона ε_φ, которая добавляется к выходному сигналу нейрона, превышает допустимую погрешность его выходного сигнала ε_1, то по формуле $\varepsilon \le (\varepsilon_1 - \varepsilon_\varphi)/|\varphi'(A)|$, где ε – погрешность входного сигнала нелинейного преобразователя, получаем отрицательную погрешность. Отрицательные погрешности в методе обратного распространения точности не имеют смысла, так как мы рассматриваем допустимую погрешность как величину интервала, в котором может изменяться погрешность. То же самое можно сказать относительно точки ветвления и сумматора. В случае получения отрицательной погрешности метод обратного распространения точности не может применяться. В такой ситуации допустимые погрешности нужно распределять специальным образом.

Рассмотрим конкретную сеть с заданными собственными погрешностями элементов. Нужно определить, возможно ли провести вычисления допустимых погрешностей методом обратного распространения точности с учетом собственных погрешностей элементов для всех сигналов сети, начиная с выходных и кончая входными сигналами. Для этого нужно выполнить прямое функционирование сети с заданными собственными погрешностями элементов и с точным вектором входных сигналов сети. Если полученный вектор выходных сигналов при этом будет отличаться от точного вектора выходных сигналов более, чем на δ (допустимую погрешность выходных сигналов сети), то с такими собственными погрешностями элементов для данной сети метод обратного распространения точности невыполним. В этом случае можно либо

увеличить допустимую погрешность выходных сигналов δ, либо уменьшить собственные погрешности элементов.

Нам нужно определить, как вычислять допустимые погрешности сигналов, если это невозможно сделать методом обратного распространения точности. Выясним, всегда ли возможно распределить их так, чтобы собственные погрешности элементов не превышали допустимых погрешностей выходных сигналов этих элементов, и при этом погрешность выходных сигналов сети не превышала δ.

Для точки ветвления и нелинейного преобразователя формулы вычисления допустимых погрешностей сигналов методом обратного распространения точности заданы жестко. Единственным элементом сети, который позволяет влиять на ситуацию, является сумматор. Для сумматора возможны три варианта распределения допустимых погрешностей по его входам, в том числе приоритетное. При приоритетном распределении можно назначать погрешности для тех входов, для которых собственная погрешность элемента превышает погрешность выходного сигнала.

Если на каком-то из элементов (нелинейный преобразователь, точка ветвления или сумматор) собственная погрешность элемента превышает допустимую погрешность его выходного сигнала, то возвращаемся к тому сумматору, чьим входным сигналом является выходной сигнал данного элемента. Для этого сумматора распределяем допустимые погрешности по его входам таким образом, чтобы можно было вычислить допустимую погрешность входного сигнала того элемента, который нас интересует.

Есть ситуации, на которые влиять невозможно. К ним относятся следующие:

1) если собственные погрешности нелинейного преобразователя или сумматора стандартных нейронов выходного слоя сети превышают погрешности их выходных сигналов;

2) если в выходном слое сумма с коэффициентами α_i собственных погрешностей элементов ε_i^{own}, выходные сигналы которых являются входными

сигналами сумматора, превышает допустимую погрешность ε выходного сигнала сумматора: $\Sigma_{i=1}^{n}\alpha_i \cdot \varepsilon_i^{own} > \varepsilon$.

Если собственная погрешность превышает допустимую погрешность выходного сигнала у элемента скрытого или входного слоя, то можно попытаться распределить допустимые погрешности так, чтобы продолжить вычисления.

Нам необходимо оптимально распределить допустимые погрешности по входам сумматора, то есть распределить их таким образом, чтобы по каждому входу допустимые погрешности входных сигналов максимально превышали собственные погрешности элементов, чьи выходные сигналы являются входными сигналами сумматора.

Рассмотрим участок сети между двумя сумматорами Σ_1 и Σ_2 (рис.9). Пусть ε – это погрешность выходного сигнала сумматора Σ_2, A – точный выходной сигнал нелинейного преобразователя.

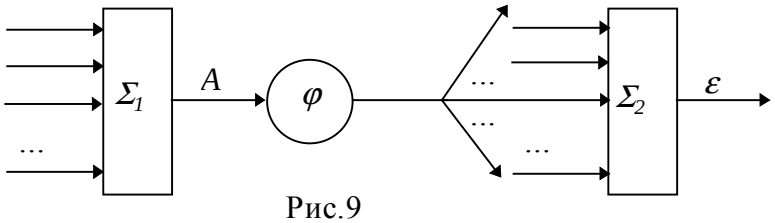


Рис.9

Предположим сначала, что собственные погрешности элементов добавляются к выходным сигналам этих элементов. То есть погрешность входного сигнала нелинейного преобразователя вычисляется по формуле $\varepsilon \le (\varepsilon_1 - \varepsilon_\varphi)/|\varphi'(A)|$, а для точки ветвления погрешность входного сигнала определяется как $\varepsilon = \min\{\varepsilon_i - \varepsilon_{tv}\}_{i=1}^{n}$.

Для каждого входа сумматора Σ_2 нам необходимо вычислить такие допустимые погрешности, которые позволили бы провести вычисления

допустимых погрешностей для точки ветвления, нелинейного преобразователя и сумматора Σ_1 предыдущего слоя. При этом допустимая погрешность выходного сигнала сумматора Σ_1 должна превышать его собственную погрешность. Это гарантирует вычисление допустимых погрешностей для слоя, предшествующего сумматору Σ_1.

Пусть сумматор Σ_1 имеет собственную погрешность ε_{Σ_1}, нелинейный преобразователь имеет собственную погрешность ε_φ и ε_{tv} – собственная погрешность точки ветвления.

Вычислим погрешность ε^{part}, которая придет к входу сумматора Σ_2 при прямом функционировании сети, начиная от выходного сигнала сумматора Σ_1.

$$\varepsilon^{part} = \varepsilon_{\Sigma_1} \cdot |\varphi'(A)| + \varepsilon_\varphi + \varepsilon_{tv}.$$

Вычисляем допустимые погрешности ε_i для входов сумматора Σ_2 при пропорциональном и равномерном распределении. Если хотя бы для одного распределения по каждому входу $\varepsilon_i^{part} < \varepsilon_i$, то можно продолжать вычисления, используя метод обратного распространения точности с учетом собственных погрешностей элементов сети. Если же для обоих распределений хотя бы по одному входу $\varepsilon_i^{part} > \varepsilon_i$, то необходимо распределять допустимые погрешности по входам сумматора Σ_2 следующим образом.

Если сумма с коэффициентами α_i погрешностей ε_i^{part} меньше допустимой погрешности выходного сигнала ($\Sigma_{i=1}^n \alpha_i \cdot \varepsilon_i^{part} < \varepsilon$), то вычисляем разность $\Sigma_{i=1}^n \alpha_i \cdot \varepsilon_i^{part} - \varepsilon$. Оставшуюся часть допустимой погрешности выходного сигнала сумматора ε распределяем равномерно по всем входам, чтобы допустимые погрешности входов превышали погрешности элементов на одну и ту же величину ξ. Тогда

$$\Sigma_{i=1}^n \alpha_i \cdot (\varepsilon_i^{part} + \xi) = \varepsilon \Rightarrow \xi = (\varepsilon - \Sigma_{i=1}^n \alpha_i \cdot \varepsilon_i^{part}) / \Sigma_{i=1}^n \alpha_i.$$

Допустимые погрешности входных сигналов сумматора будут равны $\varepsilon_i = \varepsilon_i^{part} + \xi$.

Пусть теперь собственные погрешности элементов добавляются к их входным сигналам. Допустимая погрешность входного сигнала нелинейного преобразователя при этом вычисляется следующим образом: $\varepsilon \le \varepsilon_1 / |\varphi'(A)| + \varepsilon_\varphi$. Собственная погрешность сумматора Σ_1 равна $\Sigma_{i=1}^{n} |\alpha_i| \cdot \varepsilon_{\Sigma_1}^{i}$. Допустимая погрешность точки ветвления вычисляется как было описано выше.

В этом случае погрешности ε_i^{part} вычисляются по формуле:

$$\varepsilon_i^{part} = (\varepsilon_{\Sigma_1} \cdot \Sigma_{i=1}^{n} |\alpha_i| - \varepsilon_\varphi) \cdot |\varphi'(A)| + \varepsilon_{tv}.$$

Остальные вычисления для допустимых погрешностей ε_i входных сигналов сумматора Σ_2 проводятся аналогично.

Возможен другой подход. Можно вычислять погрешности ε_i^{part} для всей сети сразу. Погрешности ε_i^{part}, вычисленные для одного слоя сети, суммируются с коэффициентами α_i того сумматора, через который они должны проходить. Полученная погрешность используется дальше для вычисления погрешности ε_i^{part} следующего слоя. При обратном прохождении сети для входов каждого сумматора будет известно, какие погрешности ε_i^{part} приходят по каждому входу, и как следует распределять допустимые погрешности сигналов. При этом погрешности ε_i^{part} вычисляются один раз и не требуется делать пересчет.

Таким образом, мы рассмотрели как распределяются допустимые погрешности сигналов для сетей с собственными погрешностями элементов. Используется встречное распространение погрешностей ε_i^{part}, которые насчитываются при прямом функционировании сети, и допустимых погрешностей сигналов, которые вычисляются в обратном направлении.

6. Метод обратного распространения точности для оценки среднеквадратических отклонений

Рассмотрим обученную нейросеть с вычисленными весами синапсов α_i. Считаем, что погрешности входных сигналов, внутренних сигналов сети и элементов отсутствуют. При векторе входных сигналов $\{z^{in}\}$ получаем вектор выходных сигналов $\{y^{out}\}$. Вектор $\{y^{out}\}$ и внутренние сигналы сети x_i будем считать точным вектором выходных сигналов и точными сигналами сети.

Рассмотрим теперь эту же сеть, но предположим, что все сигналы сети имеют некоторые погрешности. Пусть $\{y'\}$ – вектор выходных сигналов, полученный при том же векторе входных сигналов $\{z^{in}\}$, но с погрешностями внутренних сигналов сети.

Предполагаем, что внутри каждого слоя погрешности сигналов ε_i являются независимыми случайными величинами. Это предположение позволяет налагать менее жесткие требования при вычислении погрешностей сигналов.

Пусть нам задана δ – допустимая погрешность выходных сигналов сети. То есть вектор $\{y'\}$ может отличаться от вектора $\{y^{out}\}$ не более, чем на δ. Будем считать δ величиной среднеквадратического отклонения σ_{out} выходных сигналов сети $\{y'\}$.

Нам нужно выяснить, каким образом могут распределяться дисперсии сигналов при заданном σ_{out} и вычислить среднеквадратические отклонения $\sigma_i = \sqrt{D_i}$ для всех сигналов сети такие, чтобы среднеквадратическое отклонение вектора выходных сигналов $\{y'\}$ равнялось σ_{out}.

Зная среднеквадратическое отклонение выходных сигналов, можем вычислить дисперсию выходных сигналов $D_{out} = \sigma_{out}^2$, а затем, переходя от

элемента к элементу в обратном порядке, вычислим дисперсии D_i и среднеквадратические отклонения $\sigma_i = \sqrt{D_i}$ для всех сигналов сети.

Типичным участком сети является стандартный нейрон. Из стандартных нейронов состоит любая нейронная сеть. Поэтому нам достаточно определить, как вычисляются среднеквадратические отклонения сигналов для элементов стандартного нейрона. Тогда мы будем иметь возможность вычислить среднеквадратические отклонения для любого участка сети.

Выясним, как вычисляются среднеквадратические отклонения для входных сигналов точки ветвления, нелинейного преобразователя и сумматора, если нам будут известны среднеквадратические отклонения выходных сигналов этих элементов.

Если дисперсии выходных сигналов точки ветвления $D_1, D_2, \ldots, D_k$ при обратном распространении не равны между собой, то в качестве дисперсии входного сигнала точки ветвления выбирается $\min\{D_i\}_{i=1}^{k}$.

Пусть σ_1 – среднеквадратическое отклонение погрешности выходного сигнала нелинейного преобразователя. Пусть случайная величина ε (погрешность входного сигнала нелинейного преобразователя) имеет некоторую плотность распределения ρ_ε. Считаем, что математическое ожидание погрешности входного сигнала $M_\varepsilon = \int\limits_{-\infty}^{\infty} \varepsilon \cdot \rho_\varepsilon d\varepsilon = 0$ и дисперсия $D_\varepsilon = \int\limits_{-\infty}^{\infty} (\varepsilon - M_\varepsilon)^2 \rho_\varepsilon d\varepsilon = \sigma^2$.

Пусть нелинейный преобразователь имеет функцию активации φ и точный входной сигнал A. Рассмотрим линейное приближение функции активации φ в точке A. Линейное приближение имеет вид: $\varphi(A \pm \varepsilon) \approx \varphi(A) \pm \varphi'(A) \cdot \varepsilon$. Найдем математическое ожидание и дисперсию величины $\varphi(A + \varepsilon)$.

$$M_{\varphi(A+\varepsilon)} \approx \int_{-\infty}^{\infty} (\varphi(A) + \varphi'(A) \cdot \varepsilon) \cdot \rho_{\varepsilon} d\varepsilon = \varphi(A).$$

$$D_{\varphi(A+\varepsilon)} \approx \int_{-\infty}^{\infty} (\varphi(A) + \varphi'(A) \cdot \varepsilon - \varphi(A))^2 \rho_{\varepsilon} d\varepsilon = \varphi'(A)^2 \sigma^2.$$

С другой стороны, нам известно, что дисперсия выходного сигнала нелинейного преобразователя равна σ_1^2. Отсюда получаем

$$\sigma_1^2 = \varphi'(A)^2 \cdot \sigma^2 \Rightarrow \sigma = \frac{\sigma_1}{|\varphi'(A)|}.$$

Таким образом, мы вычислили среднеквадратическое отклонение входного сигнала нелинейного преобразователя для любого распределения погрешности входного сигнала ε.

Мы получили среднеквадратическое отклонение входного сигнала нелинейного преобразователя σ, которое одновременно является среднеквадратическим отклонением выходного сигнала сумматора с погрешностями входных сигналов $\sum_{i=1}^{n} \alpha_i \cdot (x_i + \varepsilon_i)$. Погрешность выходного сигнала сумматора равняется $\sum_{i=1}^{n} \alpha_i \cdot (x_i + \varepsilon_i) - \sum_{i=1}^{n} \alpha_i \cdot x_i = = \sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i$, где $\sum_{i=1}^{n} \alpha_i \cdot x_i$ – точный выходной сигнал сумматора.

Вычислим среднеквадратические отклонения σ_i входных сигналов сумматора. Рассмотрим для этого дисперсию погрешности выходного сигнала сумматора $D(\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i) = \sigma^2$.

Предположим дополнительно, что σ_i равны между собой.

$$\sigma^2 = D(\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i) = \sum_{i=1}^{n} D(\alpha_i \cdot \varepsilon_i) = \sum_{i=1}^{n} \alpha_i^2 \cdot D\varepsilon_i =$$

$$= \sum_{i=1}^{n} \alpha_i{}^2 \cdot \sigma_i{}^2 \Rightarrow \sigma_i = \frac{\sigma}{\sqrt{\Sigma_{i=1}^{n} \alpha_i{}^2}}.$$

Получили формулу для равномерного распределения среднеквадратических отклонений σ_i по входам сумматора. Если в качестве погрешности каждого входа рассматривать не ε_i, а $\alpha_i \cdot \varepsilon_i$, то получим формулу для пропорционального распределения среднеквадратических отклонений σ_i по входам сумматора.

$$\sigma^2 = D(\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i) = \sum_{i=1}^{n} D(\alpha_i \cdot \varepsilon_i) = \sum_{i=1}^{n} \sigma_i{}^2 \Rightarrow \sigma_i = \frac{\sigma}{\sqrt{n}}.$$

Кроме равномерного и пропорционального распределения среднеквадратических отклонений погрешностей по входам сумматора, может быть использовано приоритетное распределение среднеквадратических отклонений. При этом сначала назначаются среднеквадратические отклонения погрешностей для тех входов сумматора, которые наиболее значимы по какому-либо признаку, а затем оставшаяся часть среднеквадратического отклонения погрешности выходного сигнала сумматора распределяется по остальным входам равномерно или пропорционально.

Мы рассмотрели, как изменяются погрешности сигналов при прохождении через элементы сети. Предположим теперь, что не только сигналы имеют погрешности, но и все элементы сети передают приходящие к ним сигналы с некоторыми погрешностями. Пусть среднеквадратические отклонения погрешностей элементов известны и фиксированы. Выясним, как влияют собственные погрешности элементов на погрешности сигналов.

Вычислим среднеквадратические отклонения входных сигналов точки ветвления, нелинейного преобразователя и сумматора, если известны среднеквадратические отклонения выходных сигналов и собственные погрешности этих элементов.

Пусть точка ветвления имеет собственную погрешность ε_{tv} и среднеквадратическое отклонение собственной погрешности равно σ_{tv}. Собственная погрешность ε_{tv} добавляется к каждому сигналу, выходящему из точки ветвления.

Если при обратном распространении получаем дисперсии выходных сигналов точки ветвления $D_1, D_2, \ldots, D_k$ не равные между собой, то в качестве дисперсии входного сигнала точки ветвления, с учетом собственной погрешности, выбирается $min\{D_i\}_{i=1}^{k} - \sigma_{tv}^2$.

Пусть среднеквадратическое отклонение собственной погрешности нелинейного преобразователя равно σ_φ, а среднеквадратическое отклонение выходного сигнала нелинейного преобразователя равно σ_1. Собственная

погрешность нелинейного преобразователя ε_φ может добавляться либо к результату работы нелинейного преобразователя: $\varphi(A+\varepsilon)+\varepsilon_\varphi$, либо к входному сигналу нелинейного преобразователя: $\varphi(A+\varepsilon+\varepsilon_\varphi)$.

Рассмотрим оба варианта.

Пусть погрешность ε_φ добавляется к результату работы нелинейного преобразователя. Рассмотрим дисперсию

$$D(\varphi(A+\varepsilon)+\varepsilon_\varphi) = D(\varphi(A+\varepsilon)) + D(\varepsilon_\varphi) = \sigma_{own}^2 + \sigma_\varphi^2 = \sigma_1^2.$$

Отсюда получаем, что дисперсия непосредственно выходного сигнала нелинейного преобразователя равна $\sigma_{own}^2 = \sigma_1^2 - \sigma_\varphi^2$.

Среднеквадратическое отклонение для входного сигнала нелинейного преобразователя вычисляется как указано выше. В качестве дисперсии выходного сигнала в формуле используется вычисленная дисперсия $\sigma_{own}^2 = \sigma_1^2 - \sigma_\varphi^2$. Среднеквадратическое отклонение погрешности входного сигнала нелинейного преобразователя будет равняться $\sigma = \sigma_{own}/|\varphi'(A)|$.

Пусть теперь собственная погрешность нелинейного преобразователя добавляется к его входному сигналу: $\varphi(A+\varepsilon+\varepsilon_\varphi)$.В этом случае погрешность входного сигнала имеет математическое ожидание $M_{\varepsilon+\varepsilon_\varphi} = \int_{-\infty}^{\infty} (\varepsilon+\varepsilon_\varphi)\cdot\rho_\varepsilon d(\varepsilon+\varepsilon_\varphi) = 0$ и дисперсию

$$D_{\varepsilon+\varepsilon_\varphi} = \int_{-\infty}^{\infty} (\varepsilon+\varepsilon_\varphi - M_{\varepsilon+\varepsilon_\varphi})^2 \rho_\varepsilon d(\varepsilon+\varepsilon_\varphi) = \sigma^2 + \sigma_\varphi^2.$$

Вычислим математическое ожидание и дисперсию выходного сигнала нелинейного преобразователя, рассматривая линейное приближение $\varphi(A+\varepsilon+\varepsilon_\varphi)$.

$$M_{\varphi(A+\varepsilon+\varepsilon_\varphi)} \approx \int_{-\infty}^{\infty} (\varphi(A)+\varphi'(A)\cdot(\varepsilon+\varepsilon_\varphi))\cdot\rho_\varepsilon d(\varepsilon+\varepsilon_\varphi) = \varphi(A).$$

$$\sigma_1^2 = D_{\varphi(A+\varepsilon+\varepsilon_\varphi)} \approx \int_{-\infty}^{\infty} (\varphi(A) + \varphi'(A)\cdot(\varepsilon+\varepsilon_\varphi) - \varphi(A))^2 \rho_\varepsilon d(\varepsilon+\varepsilon_\varphi) =$$

$$= \varphi'(A)^2(\sigma^2 + \sigma_\varphi{}^2)$$

Отсюда получаем $(\sigma^2 + \sigma_\varphi{}^2) = \sigma_1^2 / \varphi'(A)^2 \Rightarrow \sigma = \sqrt{\sigma_1^2 / \varphi'(A)^2 - \sigma_\varphi^2}$.

Перейдем к вычислению среднеквадратических отклонений входных сигналов сумматора. Пусть среднеквадратическое отклонение выходного сигнала сумматора равно σ, собственное среднеквадратическое отклонение погрешности сумматора равно σ_Σ.

Собственная погрешность сумматора может добавляться либо к выходному сигналу сумматора: $\sum_{i=1}^{n} \alpha_i \cdot (x_i + \varepsilon_i) + \varepsilon_\Sigma$, либо к каждому входу сумматора: $\sum_{i=1}^{n} \alpha_i \cdot (x_i + \varepsilon_i + \varepsilon_\Sigma^i)$, где $\varepsilon_\Sigma^i = \varepsilon_\Sigma / n$.

Пусть собственная погрешность добавляется к выходному сигналу сумматора. Вычислим среднеквадратическое отклонение погрешностей для входных сигналов сумматора. Рассмотрим для этого дисперсию

$$D(\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i + \varepsilon_\Sigma) = \sigma^2.$$

Для равномерного распределения среднеквадратических отклонений предполагаем, что σ_i равны между собой.

$$D(\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i + \varepsilon_\Sigma) = D(\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i) + D(\varepsilon_\Sigma) =$$

$$= \sum_{i=1}^{n} \alpha_i^2 \cdot \sigma_i^2 + \sigma_\Sigma^2 \Rightarrow \sigma_i = \sqrt{\frac{\sigma^2 - \sigma_\Sigma^2}{\Sigma_{i=1}^n \alpha_i^2}}.$$

Если будем рассматривать пропорциональное распределение среднеквадратических отклонений входных сигналов сумматора, то получим

$$\sigma^2 = D(\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i + \varepsilon_\Sigma) = \sum_{i=1}^{n} \sigma_i^2 + \sigma_\Sigma^2 \Rightarrow \sigma_i = \sqrt{\frac{\sigma^2 - \sigma_\Sigma^2}{n}}.$$

Пусть теперь собственное среднеквадратическое отклонение сумматора добавляется к каждому входу сумматора: $\sum_{i=1}^{n} \alpha_i \cdot (x_i + \varepsilon_i + \varepsilon_\Sigma^i)$.

Вычислим среднеквадратическое отклонение погрешностей для входных сигналов сумматора. Рассмотрим для этого дисперсию $D(\sum_{i=1}^{n} \alpha_i \cdot (\varepsilon_i + \varepsilon_\Sigma^i))$ $= \sigma^2$.

Для равномерного распределения среднеквадратических отклонений предполагаем, что σ_i равны между собой.

$$D(\sum_{i=1}^{n} \alpha_i \cdot (\varepsilon_i + \varepsilon_\Sigma^i)) = D(\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_i) + D(\sum_{i=1}^{n} \alpha_i \cdot \varepsilon_\Sigma^i) = \sum_{i=1}^{n} \alpha_i^2 \cdot \sigma_i^2 +$$

$$+ \sum_{i=1}^{n} \alpha_i^2 \cdot (\sigma_\Sigma^i)^2 \Rightarrow \sigma_i = \sqrt{\frac{\sigma^2 - (\sigma_\Sigma^i)^2 \cdot \Sigma_{i=1}^n \alpha_i^2}{\Sigma_{i=1}^n \alpha_i^2}}.$$

Если будем рассматривать пропорциональное распределение среднеквадратических отклонений входных сигналов сумматора, то получим

$$\sigma^2 = D(\sum_{i=1}^{n} \alpha_i \cdot (\varepsilon_i + \varepsilon_\Sigma^i)) = \sum_{i=1}^{n} \sigma_i^2 + \sum_{i=1}^{n} \alpha_i^2 \cdot (\sigma_\Sigma^i)^2 \Rightarrow$$

$$\sigma_i = \sqrt{\frac{\sigma^2 - (\sigma_\Sigma^i)^2 \cdot \Sigma_{i=1}^n \alpha_i^2}{n}}$$

Зная, как вычисляются среднеквадратические отклонения погрешностей для всех элементов стандартного нейрона, можно вычислить среднеквадратические отклонения погрешностей сигналов для всей сети. Если заданы среднеквадратические отклонения погрешностей для выходных сигналов сети, можно вычислить среднеквадратические отклонения погрешностей для последнего слоя сети. Когда вычислены среднеквадратические отклонения погрешностей всех входных сигналов последнего слоя сети, переходим к вычислению среднеквадратических отклонений погрешностей предпоследнего слоя и так далее.

Рассмотрим пример на рис.10. Пусть дана сеть с тремя нейронами входного слоя, двумя нейронами скрытого слоя и одним выходным нейроном. На рисунке показаны сигналы, проходящие по сети при данном векторе входных сигналов, и веса связей. В данном примере элементы сети не имеют собственных погрешностей. Характеристическая функция нелинейных преобразователей имеет вид: $\varphi(x) = x/(2+|x|)$, где x – входной сигнал нелинейного преобразователя. Среднеквадратическое отклонение вектора выходных сигналов сети σ_{out} равняется 0.01. Среднеквадратические отклонения погрешностей по входам сумматора вычисляются с использованием формулы для равномерного распределения среднеквадратических отклонений.

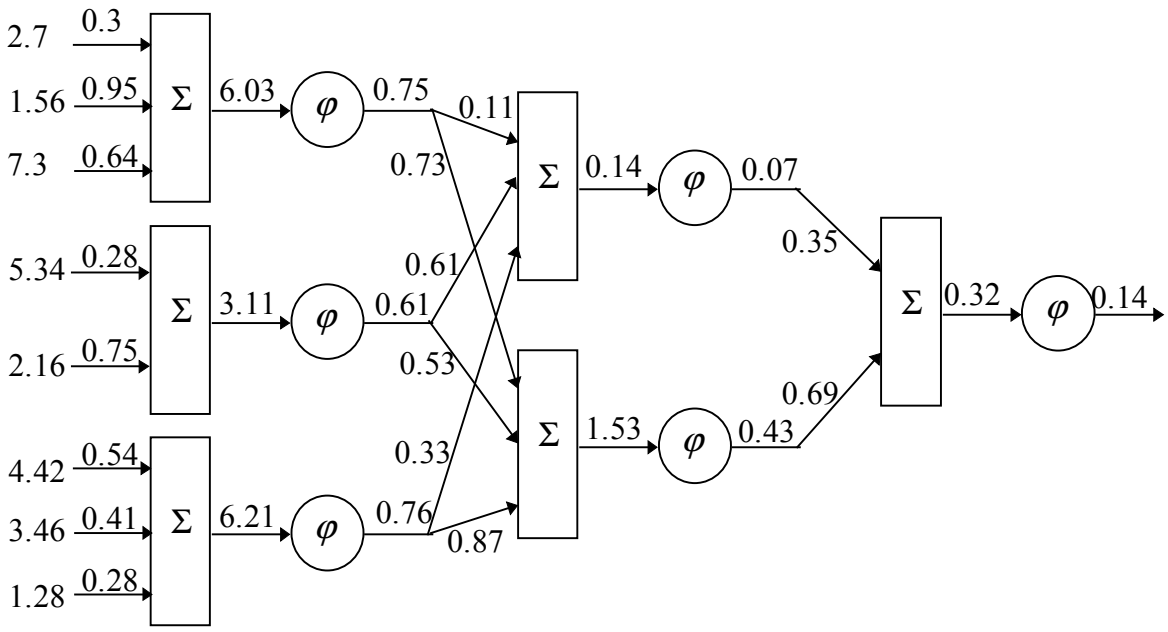


Рис.10

Σ – адаптивные сумматоры, φ – нелинейные преобразователи.

Вычислим среднеквадратические отклонения для всех сигналов сети при данном векторе входных сигналов. Все вычисленные значения в этом примере округляются до двух знаков после запятой. На рис.11 показаны вычисленные среднеквадратические отклонения для данного примера.

Зная, как вычисляются среднеквадратические отклонения погрешностей для всех элементов стандартного нейрона, можно вычислить среднеквадратические отклонения погрешностей сигналов для всей сети. Если заданы среднеквадратические отклонения погрешностей для выходных сигналов сети, можно вычислить среднеквадратические отклонения погрешностей для последнего слоя сети. Когда вычислены среднеквадратические отклонения погрешностей всех входных сигналов последнего слоя сети, переходим к вычислению среднеквадратических отклонений погрешностей предпоследнего слоя и так далее.

Рассмотрим пример на рис.10. Пусть дана сеть с тремя нейронами входного слоя, двумя нейронами скрытого слоя и одним выходным нейроном. На рисунке показаны сигналы, проходящие по сети при данном векторе входных сигналов, и веса связей. В данном примере элементы сети не имеют собственных погрешностей. Характеристическая функция нелинейных преобразователей имеет вид: $\varphi(x) = x/(2+|x|)$, где x – входной сигнал нелинейного преобразователя. Среднеквадратическое отклонение вектора выходных сигналов сети σ_{out} равняется 0.01. Среднеквадратические отклонения погрешностей по входам сумматора вычисляются с использованием формулы для равномерного распределения среднеквадратических отклонений.

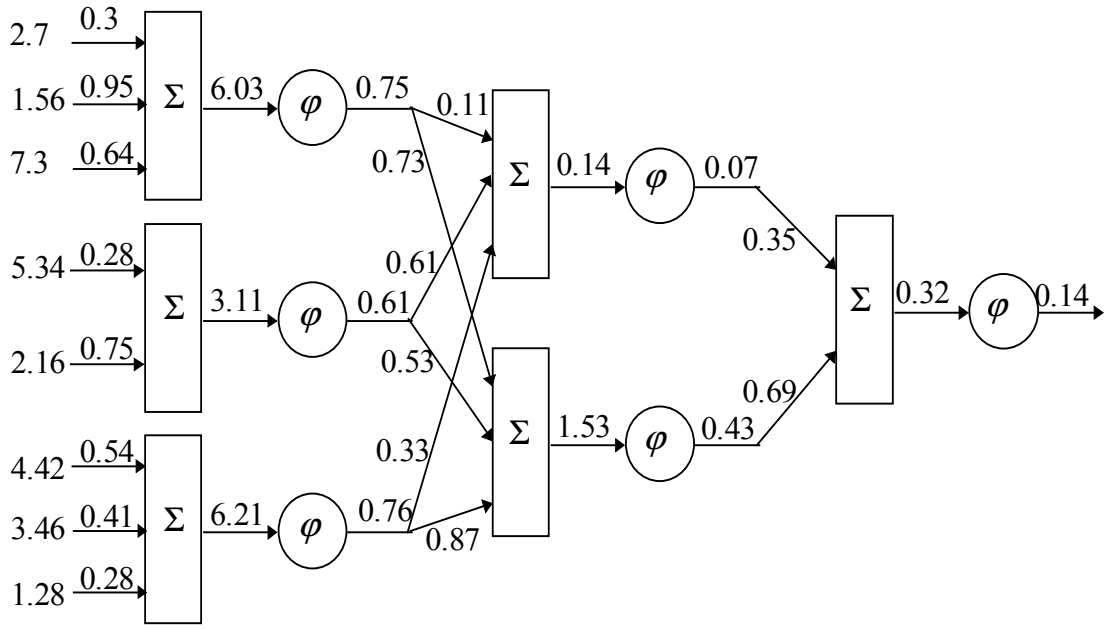


Рис.10

Σ – адаптивные сумматоры, φ – нелинейные преобразователи.

Вычислим среднеквадратические отклонения для всех сигналов сети при данном векторе входных сигналов. Все вычисленные значения в этом примере округляются до двух знаков после запятой. На рис.11 показаны вычисленные среднеквадратические отклонения для данного примера.

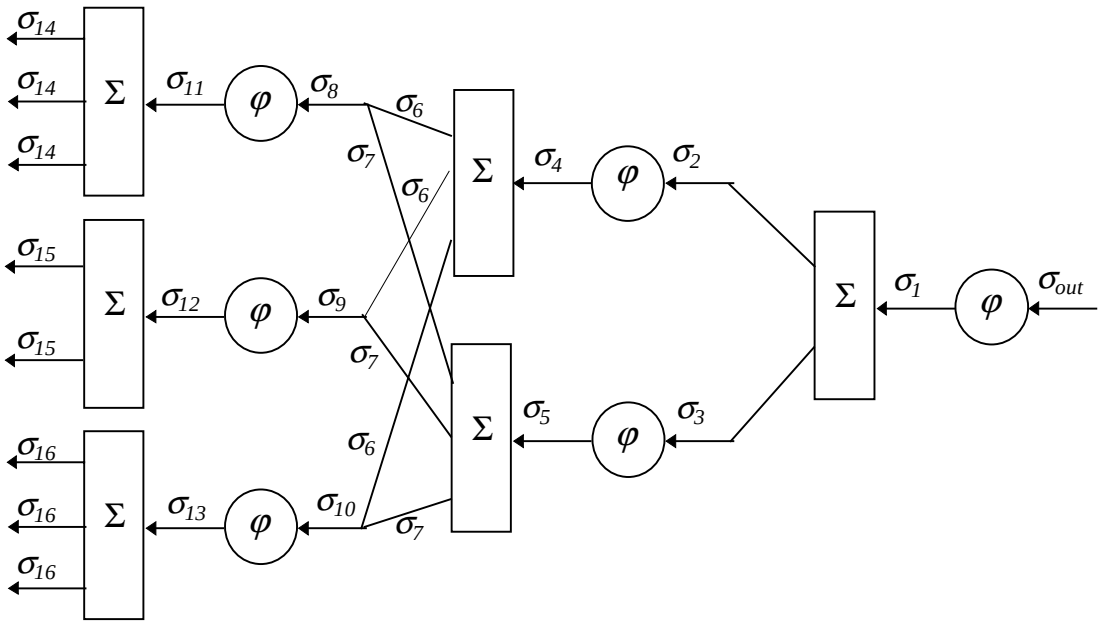


Рис. 11

$$\sigma_1 = \frac{\sigma_{out}}{|\varphi'(A)|} = \frac{0.01}{2/(2+0.32)^2} = 0.03;\ \sigma_2 = \sigma_3 = \frac{0.03}{0.35^2 + 0.69^2} = \frac{0.03}{0.77} = 0.04;$$

$$\sigma_4 = \frac{0.04}{2/(2+0.14)^2} = \frac{0.04}{0.44} = 0.08;\ \sigma_5 = \frac{0.04}{2/(2+1.53)^2} = \frac{0.04}{0.16} = 0.23;$$

$$\sigma_6 = \frac{0.08}{0.11^2 + 0.61^2 + 0.38^2} = \frac{0.08}{0.7} = 0.11;$$

$$\sigma_7 = \frac{0.23}{0.73^2 + 0.53^2 + 0.87^2} = \frac{0.23}{1.25} = 0.18;$$

$$\sigma_8 = \sigma_9 = \sigma_{10} = \min\{\sigma_6, \sigma_7\} = 0.11;\ \sigma_{11} = \frac{0.11}{2/(2+6.03)^2} = \frac{0.11}{0.03} = 3.55;$$

$$\sigma_{12} = \frac{0.11}{2/(2+3.11)^2} = \frac{0.11}{0.08} = 1.43;\ \sigma_{13} = \frac{0.11}{2/(2+6.21)^2} = \frac{0.11}{0.03} = 3.67;$$

$$\sigma_{14} = \frac{3.55}{0.3^2 + 0.95^2 + 0.64^2} = \frac{3.55}{1.49} = 2.38;\ \sigma_{15} = \frac{1.43}{0.28^2 + 0.75^2} = \frac{1.43}{0.8} = 1.79;$$

$$\sigma_{16} = \frac{3.67}{0.54^2 + 0.41^2 + 0.28^2} = \frac{3.67}{0.73} = 5.03;$$

Таким образом, получены формулы для вычисления среднеквадратических отклонений погрешностей сигналов сети, в предположении, что погрешности являются независимыми случайными величинами.

7. Анализ реализуемости сетей с собственными погрешностями элементов методом обратного распространения точности для оценки среднеквадратических отклонений

Все изложенные выше соображения о выполнимости метода обратного распространения точности справедливы и для метода обратного распространения точности для среднеквадратических отклонений погрешностей с учетом собственных погрешностей элементов. Отличие состоит в способе вычисления промежуточных среднеквадратических отклонений погрешностей.

Как и выше, рассмотрим участок сети, изображенный на рис.9. Для этого участка нам необходимо вычислить промежуточное среднеквадратическое отклонение погрешности σ^{part}.

Пусть собственное среднеквадратическое отклонение погрешности сумматора Σ_1 равно σ_{Σ_1}, собственное среднеквадратическое отклонение погрешности нелинейного преобразователя равно σ_φ и σ_{tv} – собственное среднеквадратическое отклонение погрешности точки ветвления.

Рассмотрим сначала вариант, когда собственные погрешности элементов добавляются к выходным сигналам этих элементов. В этом случае среднеквадратическое отклонение погрешности входного сигнала нелинейного преобразователя вычисляется по формуле $\sigma = \sigma_{own}/|\varphi'(A)|$, где $\sigma_{own} = \sqrt{\sigma_1^2 - \sigma_\varphi^2}$, σ_1 – общая погрешность выходного сигнала нелинейного преобразователя. Для точки ветвления среднеквадратическое отклонение погрешности входного сигнала определяется как $\min\{D_i\}_{i=1}^k - \sigma_{tv}^2$.

Среднеквадратическое отклонение погрешности σ^{part}, которое придет к входу сумматора Σ_2 при прямом функционировании сети, начиная от выходного сигнала сумматора Σ_1 (рис.9), равно

$$\sigma^{part} = \sqrt{\sigma_{\Sigma_1}^2 \cdot \varphi'(A)^2 + \sigma_{\varphi}^2 + \sigma_{tv}^2}\ .$$

Среднеквадратические отклонения погрешностей σ_i^{part} придут к каждому входу сумматора Σ_2. Если сумма квадратов среднеквадратических отклонений σ_i^{part} с коэффициентами α_i меньше квадрата среднеквадратического отклонения погрешности выходного сигнала сумматора ($\Sigma_{i=1}^n \alpha_i^2 \cdot (\sigma_i^{part})^2 < \sigma^2$), то вычисляем разность $\Sigma_{i=1}^n \alpha_i^2 \cdot (\sigma_i^{part})^2 - \sigma^2$. Оставшуюся часть квадрата среднеквадратического отклонения погрешности выходного сигнала сумматора σ распределяем равномерно по всем входам, чтобы среднеквадратические отклонения погрешностей входов превышали собственные среднеквадратические отклонения погрешностей элементов на одну и ту же величину ξ. Тогда получаем следующую формулу

$$\Sigma_{i=1}^n \alpha_i^2 \cdot ((\sigma_i^{part})^2 + \xi^2) = \sigma^2 \Rightarrow \xi = \sqrt{(\sigma^2 - \Sigma_{i=1}^n \alpha_i^2 \cdot (\sigma_i^{part})^2) / \Sigma_{i=1}^n \alpha_i^2}\ .$$

Среднеквадратические отклонения погрешностей по входам сумматора будут равны $\sigma_i = \sqrt{(\sigma_i^{part})^2 + \xi^2}$.

Пусть теперь собственные погрешности элементов добавляются к входным сигналам этих элементов. В этом случае среднеквадратическое отклонение погрешности входного сигнала нелинейного преобразователя вычисляется по формуле $\sigma = \sqrt{\sigma_1^2 / \varphi'(A)^2 - \sigma_{\varphi}^2}$, где σ_1 – погрешность выходного сигнала нелинейного преобразователя. Для точки ветвления среднеквадратическое отклонение погрешности входного сигнала определяется как было указано выше.

Среднеквадратическое отклонение погрешности σ^{part} в этом случае равно

$$\sigma^{part} = \sqrt{\left(\Sigma_{i=1}^{n}\alpha_i^2 \cdot (\sigma_{\Sigma_1}^{i})^2 + \sigma_{\varphi}^2\right)\cdot|\varphi'(A)|^2 + \sigma_{tv}^2}\,.$$

Величины ξ для вычисления среднеквадратических отклонений погрешностей входных сигналов σ_i вычисляются как было показано выше.

Промежуточные среднеквадратические отклонения погрешностей σ^{part} можно вычислять как для участков сети, так и для сети в целом.

Таким образом, мы получили формулы для вычисления среднеквадратических отклонений погрешностей сигналов нейронной сети с собственными погрешностями элементов.

Типы погрешностей

В методе обратного распространения точности приведены формулы для вычисления погрешностей сигналов сети. Эти формулы рассчитаны для сигналов, полученных при прямом функционировании сети с одним примером из обучающей выборки в качестве входных сигналов сети. Вообще говоря, допустимые погрешности сигналов зависят от вида входных сигналов сети. Исходя из этого, для метода обратного распространения точности можно выделить четыре типа допустимых погрешностей:

1) погрешности, вычисленные для одного примера;

2) погрешности, вычисленные для всей обучающей выборки;

3) погрешности, вычисленные для примеров, компоненты которых принадлежат области $A_i \le x_i \le B_i, i = 1, n$, где n – размерность области, A_i, B_i – действительные числа;

4) погрешности, вычисленные для примеров, компоненты которых принадлежат области $\Sigma_{i=1}^{n} x_i^2 \leq R^2$, где R – действительное число.

Погрешности первого типа вычисляются по формулам, описанным в методе обратного распространения точности.

Для того, чтобы вычислить погрешности второго типа, вычисляем погрешности для каждого примера из обучающей выборки. Затем в качестве допустимой погрешности для каждого элемента сети выбирается минимум допустимых погрешностей этого элемента, вычисленных для каждого примера из обучающей выборки.

Рассмотрим, как вычисляются допустимые погрешности третьего и четвертого типов. В формулах для вычисления допустимых погрешностей входной сигнал используется только у нелинейного преобразователя. Допустимые погрешности остальных элементов сети от входных сигналов не зависят. Поэтому для вычисления этих типов погрешностей следует выяснить, какие сигналы будут входными для нелинейных преобразователей, если входные сигналы сети принадлежат области, которая является прямоугольником или шаром.

Для начала рассмотрим допустимые погрешности третьего типа, то есть те допустимые погрешности элементов сети, которые получаются при входных сигналах, принадлежащих прямоугольной области. Нам известны интервалы, в которых изменяются входные сигналы сети. Требуется вычислить интервалы для входных сигналов каждого элемента сети. Будем вычислять их следующим образом. При прохождении интервалов через сумматор концы интервалов соответствующих входов умножаются на веса синапсов α_i и затем складываются. Предположим, что функция активации нелинейного преобразователя непрерывна и монотонна. Тогда в качестве концов интервала его выходного сигнала берутся значения характеристической функции нелинейного преобразователя от концов интервала входного сигнала. Точка ветвления посылает приходящий к ней интервал на входы следующих элементов.

Таким образом, для каждого элемента сети мы можем вычислить интервал, в котором изменяются его входные сигналы. Нас интересуют интервалы, в которых изменяются входные сигналы нелинейных преобразователей. Для того, чтобы вычислить допустимые погрешности входного сигнала нелинейного преобразователя, необходимо вычислить максимум производной функции активации нелинейного преобразователя на интервале изменения его входных сигналов и затем разделить на эту величину допустимую погрешность выходного сигнала нелинейного преобразователя.

Таким образом вычисляются допустимые погрешности сигналов сети для прямоугольной области входных сигналов сети.

Рассмотрим пример, в котором будем вычислять погрешности третьего типа. Воспользуемся нейросетью, изображенной на рис.9. Нейросеть имеет такие же веса синапсов, но входные сигналы принадлежат прямоугольной области:

$$I_1=[-2,6]; I_2=[3,11]; I_3=[1.5,4]; I_4=[2,7]; I_5=[4,10]; I_6=[-5,5];$$

$$I_7=[1,8]; I_8=[-4,6].$$

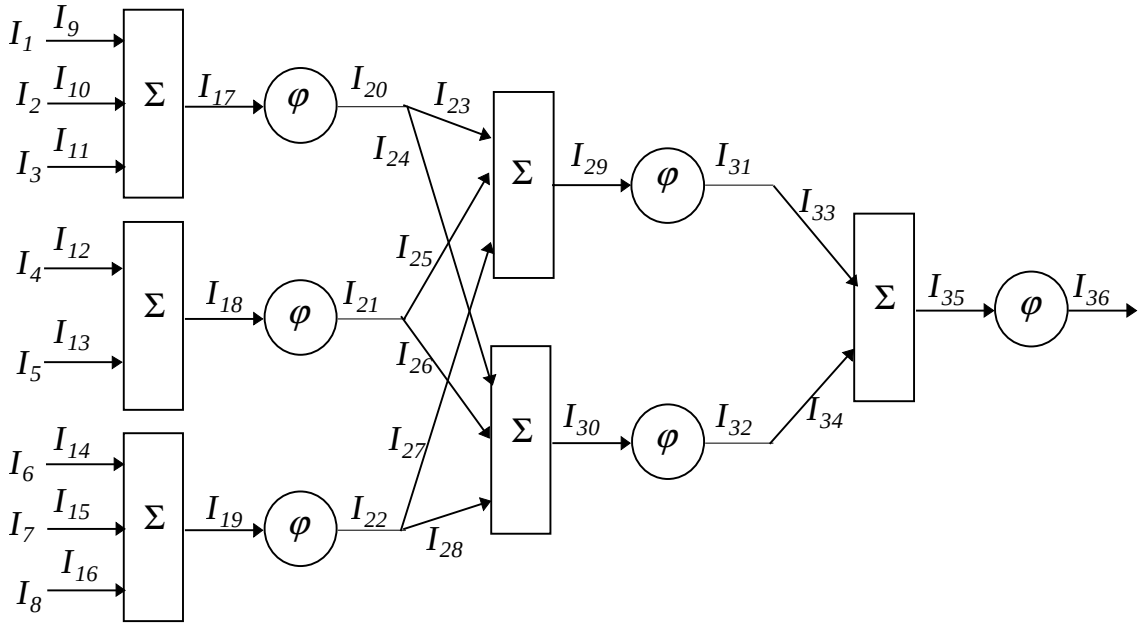


Рис. 12

Сигналы сети изменяются в следующих интервалах:

I_9=[-0.6,3.6]; I_{10}=[2.85,10.45]; I_{11}=[0.96,2.56]; I_{12}=[0.56,1.96];

I_{13}=[3,7.5]; I_{14}=[-2.7,2.7]; I_{15}=[0.41,3.28]; I_{16}=[-1.12,1.68];

I_{17}=[3.21,16.61]; I_{18}=[3.56,9.46]; I_{19}=[-3.41,7.66]; I_{20}=[0.62,0.89];

I_{21}=[0.64,0.83]; I_{22}=[-0.63,0.79]; I_{23}=[0.07,0.1]; I_{24}=[0.45,0.65];

I_{25}=[0.39,0.51]; I_{26}=[0.34,0.44; I_{27}=[-0.2,0.26]; I_{28}=[-0.55,0.69];

I_{29}=[0.26,0.86]; I_{30}=[0.24,1.78]; I_{31}=[0.12,0.3]; I_{32}=[0.1,0.47];

I_{33}=[0.04,0.1]; I_{34}=[0.07,0.32]; I_{35}=[0.11,0.42]; I_{36}=[0.05,0.17].

Мы можем вычислить максимум производной функции активации на интервале изменения входного сигнала нелинейного преобразователя.

$max\ \varphi'_{[3.21,16.61]} = 0.07$; $max\ \varphi'_{[3.56,9.46]} = 0.06$;

$max\ \varphi'_{[-3.41,7.66]} = 0.07$; $max\ \varphi'_{[0..26,0.86]} = 0.39$;

$max\ \varphi'_{[0..24,1.78]} = 0.4$; $max\ \varphi'_{[0.11,0.42]} = 0.45$.

Зная эти величины, можно вычислить допустимые погрешности третьего типа.

Выясним теперь, как вычисляются допустимые погрешности сигналов четвертого типа, то есть погрешности, получающиеся, когда область входных сигналов сети является шаром.

Рассуждения, приведенные выше для допустимых погрешностей третьего типа, справедливы и для допустимых погрешностей четвертого типа. Отличие состоит в том, что нам необходимо "развернуть" шаровую область таким образом, чтобы получить интервалы, в которых изменяются входные сигналы элементов.

Рассмотрим для этого квадраты выходных сигналов сумматоров входного слоя сети. Используя неравенство Коши, получаем $A^2 = (\Sigma_{i=1}^k \alpha_i x_i)^2 \le \Sigma_{i=1}^k \alpha_i^2 \cdot \Sigma_{i=1}^k x_i^2 \le R^2 \cdot \Sigma_{i=1}^k \alpha_i^2 \Rightarrow |A| \le R\sqrt{\Sigma_{i=1}^k \alpha_i^2}$, где k – число входных сигналов сумматора. Получили интервалы, в которых изменяются выходные сигналы сумматоров входного слоя нейронной сети. Используя эти интервалы, можем вычислить интервалы, в которых изменяются входные сигналы элементов сети. Затем, как уже было описано выше, вычисляем допустимые погрешности входных сигналов нелинейных преобразователей.

8. Обсуждение

Как метод обратного распространения точности, так и метод обратного распространения среднеквадратических отклонений погрешностей можно применять к сетям не только слоистой структуры, но также к циклическим и полносвязным сетям. Рассматривая такт функционирования сети как слой, "разворачиваем" циклические и полносвязные сети в сети слоистой структуры. Вычисляем допустимые погрешности (среднеквадратические отклонения погрешностей) для сигналов стандартных нейронов каждого слоя. Затем "сворачиваем" слоистую сеть в исходную. Так как каждый слой полученной сети на самом деле является тактом функционирования, то для каждого сигнала сети на разных тактах получаем разные допустимые погрешности

(среднеквадратические отклонения погрешностей). В качестве допустимой погрешности (среднеквадратического отклонения погрешности) для каждого сигнала сети выбирается минимум этих величин по всем тактам.

Идея этих методов возникла при решении задачи бинаризации нейронной сети. Бинаризация состоит в построении такой сети, которая функционирует так же, как и исходная, но имеет веса синапсов, равные 0 или 1 (вариант: +1 или -1).

Но метод обратного распространения точности и метод обратного распространения среднеквадратических отклонений погрешностей сигналов сети интересен не только и не столько в приложении к задаче бинаризации. Их можно применять при решении ряда других задач. Например, вычислив допустимые погрешности (среднеквадратические отклонения погрешностей) для всей сети, можно выяснить, в каких пределах можно варьировать входные данные и сигналы на любом участке сети, чтобы вектор выходных сигналов при этом изменился не более, чем на заданную величину.

Метод обратного распространения точности для среднеквадратических оценок погрешности позволяет получать формулы для вычисления погрешностей сигналов сети, налагающие менее жесткие ограничения на величину погрешностей по сравнению с гарантированными интервальными оценками погрешностями. Если для гарантированных интервальных оценок при обратном прохождении слоев допустимые погрешности сигналов уменьшаются, то для среднеквадратических оценок погрешностей есть ситуации, когда погрешности увеличиваются от последнего слоя к первому. Если погрешности сигналов являются независимыми случайными величинами, то, как показано в примере, даже при больших погрешностях входных сигналов получаются достаточно точные выходные сигналы.

9. Заключение.

Таким образом, решение задачи вычисления допустимых погрешностей (среднеквадратических отклонений погрешностей) для каждого сигнала сети методом обратного распространения точности удивительно похоже на метод обратного распространения ошибки, но с другими правилами прохождения элементов. Метод позволяет формулировать требования к точности вычисления и реализации технических устройств, если известны требования к точности выходных сигналов сети.

Литература.

1. Горбань А.Н. Обучение нейронных сетей. –М.: СП “ПараГраф”, 1990. – 159 с.
2. Нейроинформатика и ее приложения // Материалы 3 Всероссийского семинара, 6-8 октября 1995 г. Ч. 1 /Под редакцией А.Н.Горбаня. Отв. за выпуск Г.М.Цибульский; Красноярск: изд. КГТУ. – 1995. – 229 с.
3. Kimura T., Shima T. Synapse weight accuracy of analog neuro chip // Proceedings of International Joint Conference on Neural Networks. – Japan, Nagoya, October 25-29, 1993. – Vol.1. – P. 891-894.
4. Anguita D., Ridella S., Rovetta S. Limiting the effects of weight errors in feed forward networks using interval arithmetic // Proceedings of International Conference on Neural Networks (ICNN’96). – USA, Washington, June 3-6, 1996. – Vol.1. – P. 414-417.
5. Edwards P., Murray A. Modelling weight- and input-noise in MLP learning // Proceedings Of International Conference on Neural Networks (ICNN’96). – USA, Washington, June 3-6, 1996. – Vol.1. – P. 78-83.

6. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. Новосибирск: Наука. Сибирская издательская фирма РАН, 1996. – 276 с.
7. Боровков А.А. Теория вероятностей. М.: Наука, 1976. – 352 с.
8. Гнеденко Б.В. Курс теории вероятностей. М.: Наука, 1969.– 400 с.

Глава 7.

Скрытые параметры и транспонированная регрессия

А.Н.Кирдин, А.Ю.Новоходько, В. Г.Царегородцев

Вычислительный центр СО РАН в г. Красноярске[1]

Решается классическая проблема восстановления недостающих данных в следующей постановке: найти для каждого объекта наилучшую формулу, выражающую его признаки через признаки других объектов (которых должно быть по возможности меньше). Эта формула должна быть инвариантна относительно смены шкал измерения. Инвариантность достигается тем, что решение представляется в виде суперпозиции однородных дробно - линейных функций.

Строится отношение "объект - опорная группа объектов". Опорная группа выделена тем, что по признакам ее элементов наилучшим образом восстанавливаются признаки исходного объекта. Решение дается с помощью нейронной сети специальной архитектуры. Предлагается способ минимизации опорной группы, использующий преимущества нейросетевого подхода.

Метод транспонированной регрессии применяется к задаче интерполяции свойств химических элементов. Исследуется точность интерполяции потенциалов ионизации химических элементов при помощи транспонированной линейной регрессии. Достигнутая точность позволяет предсказать отсутствующие в справочной литературе значения высших (с 5-го по 10-й) потенциалов ионизации для элементов с атомными номерами от 59-го до 77-го и рекомендовать метод для интерполяции иных физических и химических свойств элементов и соединений.

1. Гипотеза о скрытых параметрах

Пусть задано некоторое множество объектов и совокупность («номенклатура») признаков этих объектов. Для каждого объекта может быть

[1] 660036, Красноярск-36, ВЦК СО РАН. E-mail: amse@cc.krascience.rssi.ru

определен вектор значений его признаков - полностью или частично. Если эти значения для каких-либо объектов определены не полностью, то возникает классическая проблема восстановления пробелов в таблицах данных [1].

Наиболее распространенный путь ее решения - построение регрессионных зависимостей. Предполагается, что одни свойства каждого из объектов могут быть с достаточной степенью точности описаны как функции других свойств. **Эти функции одинаковы для различных объектов**. Последнее предположение выполняется далеко не всегда.

Что делать, если не удается построить регрессионной зависимости, общей для всех объектов? В этом случае естественно предположить, что существуют неописанные и неизмеренные свойства объектов - и именно в них и заключаются скрытые различия, не дающие построить искомые зависимости. Эти неучтенные и неизмеренные свойства; от которых зависят наблюдаемые параметры, называют «скрытыми параметрами», а предположение о том, что все дело в них - **гипотезой о скрытых параметрах.**

Проблема скрытых параметров стала знаменитой, благодаря квантовой механике. Многолетние попытки свести квантовые неопределенности к различию в значениях скрытых параметров и поиск этих самых параметров не увенчались успехом. В этом случае проблема отсутствия однозначных связей между характеристиками объектов оказалась глубже, а квантовые неопределенности признаны несводимыми к различию в значениях неизмеренных, но в принципе доступных измерению величин - для квантовых объектов микромира скрытых параметров не нашли.

За пределами миров квантовой механики различия между объектами всегда объяснимы наличием скрытых параметров. В нашем обычном макроскопическом мире проблема состоит не в существовании скрытых параметров, а в эффективной процедуре их поиска и учета, а также в разделении ситуаций на те, для которых разумно искать скрытые параметры, и те, для которых больше

подходит представления о неустранимых (в данном контексте) случайных различиях.

Одна из простейших форм предположения о скрытых параметрах - **гипотеза о качественной неоднородности выборки**. Она означает, что скрытые параметры принимают сравнительно небольшое конечное число значений и всю выборку можно разбить на классы, внутри которых скрытые параметры, существенные для решения интересующей нас задачи регрессии, постоянны. Каждой такой выборке будет соответствовать «хорошая» регрессионная зависимость.

Построить классификацию (без учителя), соответствующую данной гипотезе можно только на основе предположении о форме искомой регрессионной зависимости наблюдаемых параметров от наблюдаемых же параметров внутри классов (**задача о мозаичной регрессии**). Если предполагается линейная зависимость, то эта задача классификации решается методом динамических ядер, только место точек - центров тяжести классов (как в сетях Кохонена) - занимают линейные многообразия, каждое из которых соответствует линейному регрессионному закону своего класса [2].

Регрессионные зависимости, которые строятся с помощью нейронных сетей, также образуют вполне определенный класс и для них тоже возможна соответствующая классификация без учителя. Изящный способов решения проблемы скрытых параметров для нейросетевых уравнений регрессии реализован в пакете «MultiNeuron» [2,3]. Достаточно большая нейронная сеть может освоить любую непротиворечивую обучающую выборку, однако, как показывает опыт, если малая нейронная сеть не может обучиться, то из этого можно извлечь полезную информацию. Если не удается построить удовлетворительную регрессионную зависимость при заданном (небольшом) числе нейронов и фиксированной характеристике («крутизне» функции активации) каждого нейрона, то из обучающей выборки исключаются наиболее сложные примеры до тех пор, пока сеть не обучится. Так получается класс,

который предположительно соответствует одному значению скрытых параметров. Далее обучение можно продолжить на отброшенных примерах и т.д.

Пример. В одном из проводимых исследований [3] нейросеть обучали ставить диагноз вторичного иммунодефицита (недостаточности иммунной системы) по иммунологическим и метаболическим параметрам лимфоцитов. В реальной ситуации по сдвигам таких параметров иногда бывает трудно сделать верное заключение (и это хорошо известная в иммунологии проблема соотношения клинической картины и биохимических проявлений иммунодефицитов). Были обследованы здоровые и больные люди, параметры которых использовались для обучения. Однако нейросеть не обучалась, причем хорошо распознавала все до единого примеры здоровых людей, а часть примеров больных путала со здоровыми. Тогда был сделан следующий шаг: каждый раз, когда сеть останавливала работу, из обучающей выборки убирался пример, на данный момент самый трудный для распознавания, и после этого вновь запускался процесс обучения. Постепенно из обучающей выборки были исключена примерно одна треть больных (при этом ни одного здорового!), и только тогда сеть обучилась полностью. Так как ни один здоровый человек не был исключен из обучения, группа здоровых не изменилась, а группа больных оказалась разделена на 2 подгруппы - оставшиеся и исключенные примеры больных. После проведения статистического анализа выяснилось, что группа здоровых и исходная группа больных практически не отличаются друг от друга по показателям метаболизма лимфоцитов. Однако получившиеся 2 подгруппы больных статистически достоверно отличаются от здоровых людей и друг от друга по нескольким показателям внутриклеточного метаболизма лимфоцитов. Причем в одной подгруппе наблюдалось увеличение активности большинства лимфоцитарных ферментов по сравнению со здоровыми, а в другой подгруппе - депрессия (снижение активности).

В научном фольклоре проблема скрытых параметров описывается как задача отделения комаров от мух: на столе сидят вперемежку комары и мухи, требуется провести разделяющую поверхность, отделяющую комаров от мух.

Данные здесь - место на плоскости, скрытый параметр - видовая принадлежность, и он через данные не выражается.

2. Теорема о скрытых параметрах

Ряд алгоритмов решения проблемы скрытых параметров можно построить на основе следующей теоремы. Пусть n - число свойств, N - количество объектов, $\{x^i\}_{i=1}^N$ - множество векторов значений признаков. Скажем, что в данной группе объектов выполняется уравнения регрессии ранга r, если все векторы $\{x^i\}_{i=1}^N$ принадлежат n-r-мерному линейному многообразию. Как правило, в реальных задачах выполняется условие $N>n$. Если же $n \geq N$, то векторы $\{x^i\}_{i=1}^N$ принадлежат N-1-мерному линейному многообразию и нетривиальные регрессионные связи возникают лишь при ранге $r>n-N+1$. Ранг регрессии r измеряет, сколько независимых линейных связей допускают исследуемые свойства объектов. Число r является коразмерностью того линейного подпространства в пространстве векторов признаков, которому принадлежат наблюдаемы векторы признаков объектов. Разумеется, при обработке реальных экспериментальных данных необходимо всюду добавлять «с заданной точностью», однако пока будем вести речь о точных связях.

Следующая теорема о скрытых параметрах позволяет превращать вопрос о связях между различными свойствами одного объекта (одной и той же для разных объектов) в вопрос о связи между одним и тем же свойством различных объектов (одинаковой связи для различных свойств) - транспонировать задачу регрессии. При этом вопрос о качественной неоднородности выборки «транспонируется» в задачу поиска для каждого объекта такой группы объектов (опорной группы), через свойства которых различные свойства данного объекта выражаются одинаково и наилучшим образом.

Теорема. Пусть для некоторого $r>0$ существует такое разбиение $\{x^i\}_{i=1}^N$ на группы $\{x^i\}_{i=1}^N = \bigcup_{j=1}^k Y_j$, что $r>n-N_j+1$ (где N_j - число элементов в Y_j), и для

каждого класса Y_j выполняются уравнения регрессии ранга r. Тогда для каждого объекта x^i из $\{x^i\}_{i=1}^N$ найдется такое множество W_i (опорная группа объекта x^i) из k объектов, что $n\text{-}r+1 \geq k$ и для некоторого набора коэффициентов λ_y

$$x^i = \sum_{y \in W_i} \lambda_y y, \ \sum_{y \in W_i} \lambda_y = 1. \qquad (1)$$

Последнее означает, что значение *каждого* признака объекта x^i является линейной функцией от значений этого признака для объектов опорной группы. Эта линейная функция *одна и та же* для всех признаков.

Линейная зависимость (1) отличается тем, что она инвариантна к изменениям единиц измерения свойств и сдвигам начала отсчета. Действительно, пусть координаты всех векторов признаков подвергнуты неоднородным линейным преобразованиям: $x_j \mapsto a_j x_j + b_j$, где j - номер координаты. Нетрудно убедиться, что при этом линейная связь (1) сохранится. Инвариантность относительно преобразования масштаба обеспечивается линейностью и однородностью связи, а инвариантность относительно сдвига начала отсчета - еще и тем, что сумма коэффициентов λ_y равна 1.

Сформулированная теорема позволяет переходить от обычной задачи регрессии (поиска зависимостей значения признака от значений других признаков того же объекта) к транспонированной задаче регрессии - поиску линейной зависимости признаков объекта от признаков других объектов и отысканию опорных групп, для которых эта зависимость является наилучшей.

Доказательство основано на том, что на каждом k-мерном линейном многообразии для любого набора из q точек $y_1, y_2, ..., y_q$ при $q>k+1$ выполнено соотношение

$$\sum_{j=1}^{q} \lambda_j y_j = 0 \text{ для некоторого набора } \lambda_j, \ \sum_{j=1}^{q} \lambda_j = 0 \text{ и некоторые } \lambda_j \neq 0.$$

С математической точки зрения теорема о скрытых параметрах представляет собой вариант утверждения о равенстве ранга матрицы, вычисляемого по строкам, рангу, вычисляемому по столбцам.

3. Транспонированная задача линейной регрессии

Изложение в этом разделе следует работам [2,5,6]. Постановка обычной задачи регрессии (или мозаичной регрессии) исходит из гипотезы о том, что одни характеристики объектов могут быть функциями других и эти функции одни и те же для всех объектов (или соответственно классов объектов).

Транспонируем таблицу данных (поменяем местами слова "объект" и "признак"). Рассмотрим гипотезу от том, что значения признака одного объекта могут быть функциями значений того же признака других объектов и эти функции одни и те же для всех признаков (или классов признаков). Получаем формально те же задачи регрессии (транспонированные задачи регрессии). Есть, однако, два содержательных отличия транспонированных задач от исходных:

1) инвариантность к смене шкал измерения - кажется маловероятным, чтобы существенные связи между признаками различных объектов зависели от шкалы измерения, поэтому необходимо, чтобы уравнения транспонированной регрессии были инвариантны относительно смены шкалы измерения любого признака (обычно - линейного неоднородного преобразования $x'=ax+b$ однородная часть которого описывает смену единицы измерения, а свободный член - сдвиг начала отсчета);

2) в традиционных задачах регрессии предполагается, что объектов достаточно много (N), по сравнению с числом признаков n, иначе (при $N<n$) точные линейные соотношения возникнут просто из-за малого числа объектов, так как через N точек всегда можно провести линейное многообразие размерности N-1. В противовес этому "транспонированное" предположение о достаточно большом числе признаков ($n>N$) кажется нереалистичным.

Требование инвариантности к смене шкал приводит к специальным ограничениям на вид функций регрессии, а недостаточность количества признаков (в сравнении с числом объектов) для построения транспонированной регрессии вынуждает нас для каждого объекта искать небольшую группу, по свойствам которых можно восстановить характеристики данного.

Задача построения таких групп объектов была чрезвычайно популярна в химии перед открытием Менделеевым периодического закона (1871 г.). С 1817 г. (Деберейнер) были опубликованы десятки работ на эту тему [7]. Именно они поставили исходный материал для систематизации элементов. Деберейнер обнаружил триады, в которых свойства среднего элемента могут быть оценены как средние значения этих свойств для крайних членов триады. Его труды продолжили Гмелин, Гладстон, Дюма и другие. Вот некоторые из таких триад: K-Na-Li, Ba-Sr-Ca, Cl-Br-J, S-Se-Te, P-As-Sb, W-V-Mo, ...

Один из наиболее полных списков триад был опубликован Ленсеном (1857). Он же заметил, что для большей точности иногда полезно брать "эннеады" - девятки, составленные из трех триад.

Менделеев писал: "...между всеми... учеными, которые раньше меня занимались сравнением величин атомных весов элементов, я считаю, что обязан преимущественно двум: Ленсену и Дюма. Я изучил их исследования и они меня побудили искать действительный закон" (цит. по [7], с. 220-222).

Более общим образом задача ставится так: найти для каждого объекта наилучшую линейную формулу, выражающую его вектор признаков через векторы признаков других объектов (которых должно быть по возможности меньше). Эта формула должна быть инвариантна относительно смены шкал.

Итак, требуется построить отношение, связывающее объекты с группами объектов, по которым для него строятся интерполяционные формулы. Проделав эту работу "в лоб" (по базам данных и без обращения к интуиции химиков) для большого числа элементов (объектов) и потенциалов ионизации (признаков), мы получили хорошее согласие с экспериментом и предсказали ряд неизвестных ранее высших потенциалов ионизации. Результаты будут описаны в следующем разделе.

Предположим, что некоторый большой набор свойств - внешних, эмпирических данных об объекте (явление) является сюръекцией небольшого набора внутренних, теоретических переменных (сущности). Эта идея позволяет сделать предположение о том, что размер опорной группы объектов, по которой

наилучшим образом восстанавливаются свойства данного объекта, не только не должен превосходить размер набора свойств (иначе заведомо возникнут точные линейные соотношения), но и быть малым настолько, насколько это позволяет заданная точность [2-5].

Если предположить, что для некоторого множества объектов зависимость между теоретическим и эмпирическим линейна, и векторы теоретических параметров объектов данного множества лежат в линейном многообразии размерности q, то размер опорной группы не будет превосходить q+1.

Другое условие, налагаемое на искомую формулу, требует инвариантности к смене шкал измерений. Разумно считать, что глубинные связи не зависят от единиц, в которых выражены значения свойств объектов:

$$f(a\mathbf{y}^1+b,\ldots,a\mathbf{y}^{\mathbf{q}}+b) = a\,f(\mathbf{y}^1,\ldots,\mathbf{y}^{\mathbf{q}})+b$$

Если в качестве искомой формулы рассматривать линейную комбинацию векторов опорной группы, то требуемой инвариантности можно достичь, наложив некоторое условие на коэффициенты разложения. Таковым условием является равенство суммы коэффициентов единице:

$$\tilde{\mathbf{y}} = \sum_i \alpha_i \mathbf{y}^i, \quad \sum_i \alpha_i = 1.$$

Для нелинейной регрессии естественно использовать однородные рациональные функции [2].

Рассматривались два вида решения. Первый:

$$\tilde{\mathbf{y}} = \mathbf{m}_{\mathbf{y}} + \sum_{i=1}^{q} \beta_i \left(\mathbf{y}^i - \mathbf{m}_{\mathbf{y}}\right), \ \alpha_i = \beta_i + \frac{1}{q} - \frac{1}{q}\sum_{k=1}^{q}\beta_k \qquad (2)$$

где $\tilde{\mathbf{y}}$ - восстановленный вектор свойств, $\mathbf{y}^i$ - вектор свойств i-го объекта опорной группы, q - мощность опорной группы, $\mathbf{m}_{\mathbf{y}} = \frac{1}{q}\sum_{i=1}^{q}\mathbf{y}^i$, - среднее значение

Во втором случае в качестве $\mathbf{m}_{\mathbf{y}}$ выбирался один из векторов опорной группы.

$$\tilde{\mathbf{y}} = \mathbf{y}^t + \sum_{i=1}^{q} \beta_i \left(\mathbf{y}^i - \mathbf{y}^t\right), \ \alpha_i = \beta_i \ , \ \alpha_t = 1 - \sum_{\substack{k=1 \\ k\neq t}}^{q}\beta_k \qquad (3)$$

Заметим, что легко построить нейронную сеть, вычисляющую такие формулы [5,6].

Из-за предположения о малости опорной группы объектов в качестве одного из путей решения предлагается перебор всех наборов заданного размера. Было предложено искать минимум одного из двух критериев:

а) $\|\mathbf{y}-\tilde{\mathbf{y}}\|^2+\varepsilon^2\|\boldsymbol{\alpha}\|^2\to\min$, б) $\|\mathbf{y}-\tilde{\mathbf{y}}\|+\varepsilon\|\boldsymbol{\alpha}\|\to\min$.

В случае а) точное решение находится из системы линейных уравнений. Введем обозначения:

Y - матрица векторов опорной группы, *n* строк, *q* столбцов. *n* - число известных компонент восстанавливаемого вектора **y**.

$\hat{\mathbf{Y}}=(\mathbf{y}^i-\mathbf{m}_\mathbf{y})$ - матрица **Y** в которой из каждого столбца вычтен вектор $\mathbf{m}_\mathbf{y}$ ($\mathbf{y}^t$ в случае 2).

M - матрица, все элементы которой равны 1,

m - вектор, все компоненты которого равны 1,

E - единичная матрица,

$\boldsymbol{\alpha},\boldsymbol{\beta}$ - вектора размерностью *q*.

Для выражения (2)

$$\tilde{\mathbf{y}}=\mathbf{m}_\mathbf{y}+\hat{\mathbf{Y}}\boldsymbol{\beta},\ \boldsymbol{\alpha}=\frac{1}{q}\mathbf{m}+\left(\mathbf{E}-\frac{1}{q}\mathbf{M}\right)\boldsymbol{\beta}.$$

Дифференцируя выражение а) и приравнивая нулю, получаем:

$$\left(\hat{\mathbf{Y}}^\mathsf{T}\hat{\mathbf{Y}}+\varepsilon\left(\mathbf{E}-\frac{1}{q}\mathbf{M}\right)\right)\boldsymbol{\beta}=\hat{\mathbf{Y}}^\mathsf{T}\left(\mathbf{y}-\mathbf{m}_\mathbf{y}\right).$$

Для выражения (3),

$\mathbf{e}^t$ - вектор, *t*-ая компонента которого равна 1, остальные 0.

$\mathbf{L}_t=(\mathbf{e}^t)$ - матрица, столбцы которой равны вектору $\mathbf{e}^t$.

$$\mathbf{L}_t^\mathsf{T}\mathbf{L}_t=\mathbf{M},\ \mathbf{L}_t^\mathsf{T}\mathbf{e}^t=\mathbf{m}$$

Имеем

$$\tilde{\mathbf{y}}=\mathbf{y}^t+\hat{\mathbf{Y}}\boldsymbol{\beta},\ \boldsymbol{\alpha}=\mathbf{e}^t+\left(\mathbf{E}-\mathbf{L}_t\right)\boldsymbol{\beta}$$

$$\left(\hat{\mathbf{Y}}^\mathsf{T}\hat{\mathbf{Y}}+\varepsilon(\mathbf{E}+\mathbf{M})\right)\boldsymbol{\beta}=\hat{\mathbf{Y}}^\mathsf{T}\left(\mathbf{y}-\mathbf{y}^t\right)+\varepsilon\mathbf{m}$$

Система уравнений решается для известных значений компонент вектора **y**, полученное решение используется для предсказания неизвестных значений.

В случае критерия б) в качестве начального приближения для каждого испытуемого набора рассматривались β минимизирующие невязку $\Delta=\|\mathbf{y}-\tilde{\mathbf{y}}\|$. Минимум критерия находился BFGS-методом [8].

Нами рассмотрен вариант нахождения оптимальной опорной группы фиксированного размера в задаче транспонированной линейной регрессии, когда оптимальная опорная группа отбиралась в ходе полного перебора всех возможных опорных групп. Другой предложенный вариант (оптимизационный) предполагает первоначальное задание избыточного числа объектов в опорной группе и последующее сокращение ее размера в результате отбрасывания наименее значимых параметров.

Программная реализация и переборного, и оптимизационного вариантов решения транспонированной задачи линейной регрессии выполнялась в среде MS DOS с использованием транслятора Borland C++. Текст программы соответствует ANSI–стандарту языка C++, что делает возможным перенос программы на другие аппаратные платформы (что и делалось большие базы медицинских данных обрабатывалась на компьютере Alpha Station корпорации DEC). При этом зависимые от операционной системы фрагменты программы подключаются при помощи условных директив препроцессора языка. Так, для обеспечения работы с большими файлами данных в среде MS DOS используется обращение к интерфейсу DPMI (предоставляется DPMI–расширителями и операционными системами OS/2, Windows 3.xx, Windows 95, Windows NT) для переключения в защищенный режим и обхода ограничения в 640K памяти.

Программа позволяет пользователю определять файл данных, обрабатываемые строки (объекты) и столбцы (свойства объектов), выбирать между вариантами решения и видами функции критерия, задавать значения иных параметров метода. Для обработки порядковых признаков возможна спецификация некоторых столбцов, как содержащих значения не из непрерывного, а из дискретного множества значений. Прогнозные значения

отсутствующих данных в этом случае будут приводиться к ближайшему значению из дискретного множества значений.

Результатом работы программы является файл отчета. Для каждого обрабатываемого объекта (строки базы данных) в файле отчета содержится информация об оптимальным образом приближающей объект опорной группе (номера объектов, входящих в опорную группу, и коэффициенты разложения), значение функции критерия, ошибки интерполяции известных свойств объекта и прогнозные значения для неизвестных свойств. В конце файла отчета выводятся максимальные и средние ошибки аппроксимации известных данных для всех обрабатываемых столбцов базы данных (свойств объектов).

Тестирование предлагаемого метода проводилось на модельных данных. При построении модельных данных задаются размерность теоретической проекции (число скрытых переменных), размерность эмпирической проекции (число свойств объекта) , число различных классов, вектор среднего и разброса для генерируемых данных в каждом классе. Для каждого класса случайным образом порождается линейный оператор, отображающий пространство скрытых переменных в пространство свойств объектов. Для каждого объекта случайным образом выбираются значения скрытых переменных и рассчитываются значения свойств. Тестирование проводилось в скользящем режиме по всему задачнику. Полученные результаты (Табл.1) позволяют заключить, что предложенный метод весьма эффективен, критерий вида б) с большей эффективностью определяет опорную группу при избыточном и недостаточном наборах объектов (лучше, чем МНК а)), а решение вида (2) дает лучшие по сравнению с (3) результаты при избыточном наборе объектов.

Таблица 1.

Качество восстановления по модельным данным с теоретической размерностью 3

ε	критерий	вид	средняя относительная ошибка, % при размере опорной группы			
			3	4	5	18
0.01	а	1	5	0	15	66
	а	2	5	0	15	66
	б	1	5	0	13	40
	б	2	5	0	13	66
0.1	а	1	10	16	30	72
	а	2	10	16	30	72
	б	1	6	10	14	40
	б	2	6	10	14	66

При решении задачи заполнения пробелов в таблицах данных для любой таблицы общей рекомендацией является проведение серии пробных прогнозов для определения оптимального сочетания параметров.

4. Интерполяция свойств химических элементов

Идея интерполяции свойств элементов возникла в химии еще до создания периодической системы [7]. В триадах Деберейнера (1817г.) характеристики среднего элемента триады находились как средние арифметические значений характеристик крайних элементов. Были попытки работать с тетрадами, “эннеадами” (составленными из трех триад) и т.п. Периодическая таблица Менделеева позволяет по–разному определять группу ближайших соседей для интерполяции: от двух вертикальных соседей по ряду таблицы до окружения из восьми элементов (два из того же ряда и по три из соседних рядов). Однако интерполяция свойств путем взятия среднего арифметического по ближайшим элементам таблицы не всегда (не для всех свойств и элементов) дает приемлемые результаты – требуется либо иной выбор соседей, либо другая процедура интерполяции.

Более общим образом задачу интерполяции можно поставить так: найти для каждого элемента наилучшую формулу, выражающую его вектор свойств через векторы свойств других элементов. Эту задачу и решает метод транспонированной регрессии.

В работах [9,10] исследовался полуэмпирический метод, близкий по идее к методу транспонированной регрессии. Единственное и главное отличие заключалось в том, что среди параметров сразу фиксировался набор «теоретических» и строились зависимости остальных свойств от них (в частности, зависимости потенциалов ионизации от атомного номера).

Используем метод транспонированной линейной регрессии для интерполяции и прогноза высших потенциалов ионизации (ПИ). Напомним, что *n*–й потенциал ионизации A – энергия, которую необходимо затратить, чтобы оторвать *n*–й электрон от иона $A^{(n-1)+}$ (n–1 раз ионизированного атома A). Зависимость ПИ от атомного номера (рис.1) нелинейна и сложна.

Следуя формальному смыслу, *n*–й ПИ атома A следует относить все к тому же атому. Однако структура энергетических уровней иона определяется зарядом ядра и числом электронов. Для атома оба этих числа совпадают с атомным номером, но для ионов уже различны. Как и в работах [9,10], *n*–й потенциал ионизации атома с атомным номером *m* будем искать как функцию от *m*–*n*+1. Объектами будут служить, строго говоря, не атомы с атомным номером *m*, а *m*-электронные системы. Таким образом, второй ПИ гелия (атомный номер 2), третий ПИ лития (атомный номер 3) и т.д. относятся к одноэлектронной системе при различных зарядах ядра. Осуществляется привязка потенциала ионизации уже ионизированного атома не к этому же атому, а к *m*-электронной системе с *m*, равным имеющемуся числу электронов в ионе.

Рассмотрим результаты пробного прогноза высших потенциалов ионизации. Приведем результаты, полученные при использовании в функции критерия нормы в виде суммы абсолютных значений компонент вектора и значения ε=0.1, поскольку такое сочетание при тестировании показало себя наилучшим образом. Для того, чтобы невязки по каждому свойству равномерно входили в левую часть функции критерия, выполнялось нормирование каждого свойства (приведение к нулевому математическому ожиданию и единичному среднеквадратическому уклонению).

На рис.2 показаны ошибки прогноза ПИ (с 3-го по 10-й) при разных размерах опорных групп (2, 3 и 4 элемента в опорной группе). При этом для каждого ПИ опорные группы строились по предыдущим ПИ. Величины максимальной и средней ошибок показаны в процентах от диапазона изменения величин соответствующего ПИ. На основе приведенных графиков можно рекомендовать использование как можно большего набора однородных свойств для достижения оптимального прогноза.

Для попытки прогноза отсутствующих в справочной литературе [11,12] значений высших ПИ (с 5-го по 10-й ПИ для элементов с атомными номерами от 59-го до 77-го) изучим влияние размера опорной группы на точность прогноза при построении опорной группы по первым четырем ПИ (Рис.3). Удовлетворительная точность достигается при трех и четырех элементах в опорной группе.

Дальнейшее увеличение числа элементов в опорной группе себя не оправдывает. Увеличению точности прогноза мешают и погрешности при экспериментальном определении ПИ, особенно высших. В таблице 1 приводится прогноз отсутствующих значений ПИ.

Литература

1. Загоруйко Н.Г., Ёлкина В.Н., Тимеркаев В.С. Алгоритм заполнения пропусков в эмпирических таблицах (алгоритм “ZET”) // Вычислительные системы. — Новосибирск, 1975. — Вып. 61. Эмпирическое предсказание и распознавание образов. — С. 3-27.
2. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. Новосибирск: Наука (Сиб. отделение), 1996. 276 с.
3. Rossiev D.A., Savchenko A.A., Borisov A.G., Kochenov D.A. The employment of neural-network classifier for diagnostics of different phases of immunodeficiency // Modelling, Measurement & Control.- 1994.- V.42.- N.2. P.55-63.

4. Горбань А.Н. Проблема скрытых параметров и задачи транспонированной регрессии // Нейроинформатика и ее приложения. Тезисы докладов V Всероссийского семинара. Красноярск: изд. КГТУ, 1997.
5. Горбань А.Н., Новоходько А.Ю., Царегородцев В.Г. Нейросетевая реализация транспонированной задачи линейной регрессии // Нероинформатика и ее приложения. Тезисы докладов IV Всероссийского семинара, 5-7 октября 1996 г. Красноярск: изд. КГТУ, 1996. С. 37-39.
6. A.N. Gorban and A.Yu.Novokhodko. Neural Networks In Transposed Regression Problem, Proc. of the World Congress on Neural Networks, Sept. 15-18, 1996, San Diego, CA, Lawrence Erlbaum Associates, 1996, pp. 515-522.
7. Становление химии как науки. Всеобщая история химии / Под ред. Ю.И.Соловьева. — М.: Наука, 1983. — 464с.
8. Гилл Ф., Мюррей У., Райт М. Практическая оптимизация, — М.: Мир, 1985. — 509с.
9. Горбань А.Н., Миркес Е.М., Свитин А.П. Полуэмпирический метод классификации атомов и интерполяции их свойств // Математическое моделирование в биологии и химии. Новые подходы. — Новосибирск: Наука. Сиб. отделение, 1992. — с.204–220.
10. Горбань А.Н., Миркес Е.М., Свитин А.П. Метод мультиплетных покрытий и его использование для предсказания свойств атомов и молекул // Журнал физической химии. — 1992. — Т.66, №6. — с.1503–1510.
11. Физико-химические свойства элементов. — Киев: Наукова думка. 1965 — 808с.
12. Свойства элементов. В 2-х частях. Ч.1. Физические свойства. Справочник. — М.: Металлургия, 1976. - 600с.

Рис. 1. Зависимость 1-го ПИ от атомного номера

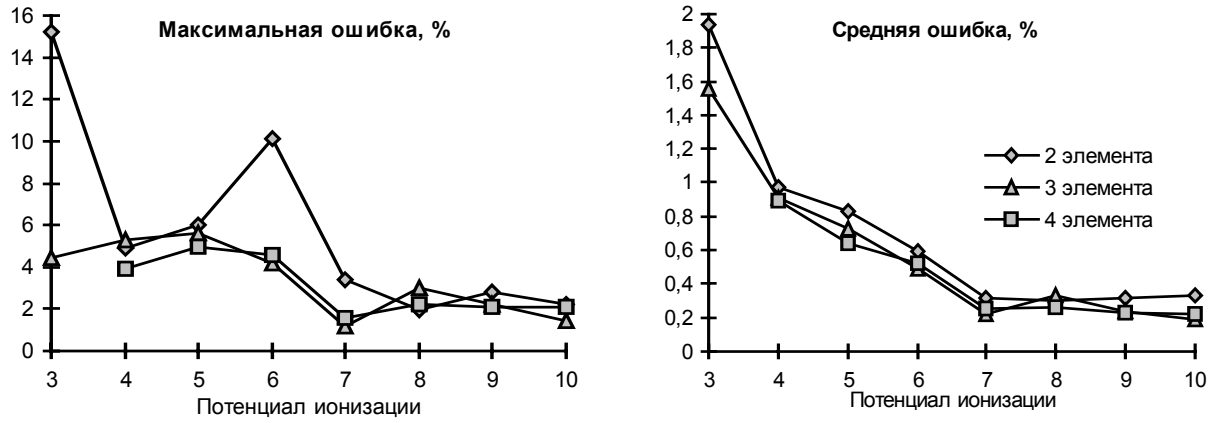


Рис. 2. Зависимость ошибки прогноза 3-10 ПИ от числа элементов в опорной группе. Опорные группы и регрессионные зависимости для каждого ПИ строились по предыдущим ПИ

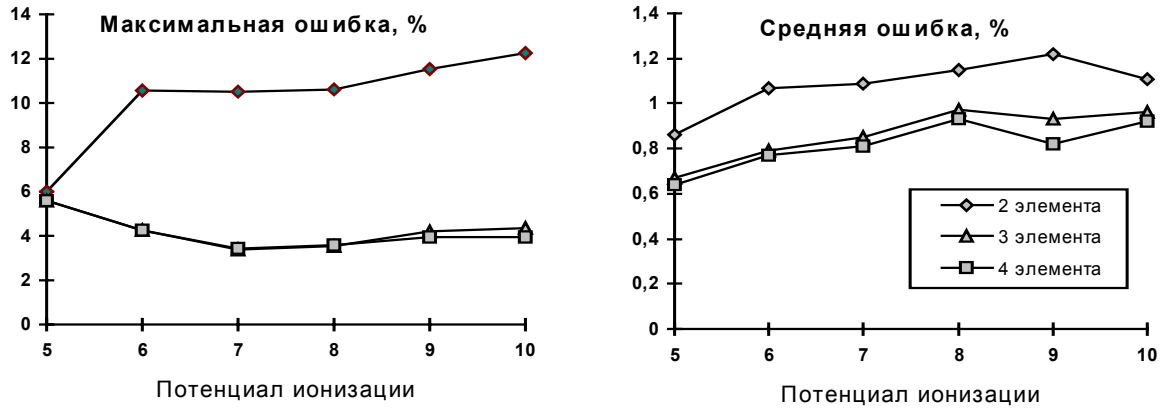


Рис. 3. Зависимость ошибок прогноза 5-10 ПИ от числа элементов в опорной группе. Опорные группы и регрессионные зависимости строились по первым четырем ПИ

Таблица 2.

Прогноз высших потенциалов ионизации отдельных химических элементов

Атомный номер	Элемент	5-й ПИ	6-й ПИ	7-й ПИ	8-й ПИ	9-й ПИ	10-й ПИ
59	Pr	50,7					
60	Nd	49,2	69,6				
61	Pm	53,6	67,7	97,1			
62	Sm	55,9	72,9	87,9	123,7		
63	Eu	56,3	76,3	93,9	110,8	153,6	
64	Gd	61,9	77,2	98,4	117,7	135,9	186,9
65	Tb	67,4	84,9	99,8	123,8	142,9	163,8
66	Dy	48,3	92,2	110,2	125,9	151,5	171,3
67	Ho	52,6	65,8	119,5	138,0	154,5	181,6
68	Er	54,5	72,1	84,6	149,7	169,1	185,7
69	Tm	54,9	74,5	93,4	106,1	182,8	203,5
70	Yb	52,4	74,8	96,1	117,3	128,6	219,9
71	Lu	57,8	71,7	96,2	120,6	143,9	154,0
72	Hf	63,1	79,2	92,2	121,0	147,8	173,2
73	Ta		85,8	102,4	115,2	147,4	177,1
74	W			110,5	128,2	140,7	176,6
75	Re				139,0	156,9	168,7
76	Os					170,1	188,6
77	Ir						203,7

Глава 8.

Нейронные сети ассоциативной памяти

Е.М.Миркес

Вычислительный центр СО РАН в г. Красноярске[1]

Рассматриваются нейронные сети ассоциативной памяти, восстанавливающие по искаженному и/или зашумленному образу ближайший к нему эталонный. Исследована информационная емкость сетей и предложено несколько путей ее повышения, в том числе - ортогональные тензорные (многочастичные) сети. Построены способы предобработки, позволяющие конструировать нейронные сети ассоциативной памяти для обработки образов, инвариантной относительно групп преобразований. Описан численный эксперимент по использованию нейронных сетей для декодирования различных кодов.

1. Введение

Прежде чем заниматься конструированием сетей ассоциативной памяти необходимо ответить на следующие два вопроса: “Как устроена ассоциативная память?” и “Какие задачи она решает?”. Когда мы задаем эти вопросы, имеется в виду не устройство отделов мозга, отвечающих за ассоциативную память, а наше представление о макропроцессах, происходящих при проявлении ассоциативной памяти.

Принято говорить, что у человека возникла ассоциация, если при получении некоторой неполной информации он может подробно описать объект, к которому по его мнению относится эта информация. Достаточно хорошим примером может служить описание малознакомого человека. К примеру, при высказывании: “Слушай, а что за парень, с которым ты вчера разговаривал на вечеринке, такой высокий блондин?”- у собеседника возникает образ

[1] 660036, Красноярск-36, ВЦК СО РАН E-mail: amse@cckr.krasnoyarsk.su

вчерашнего собеседника, не ограничивающийся ростом и цветом волос. В ответ на заданный вопрос он может рассказать об этом человеке довольно много. При этом следует заметить, что содержащейся в вопросе информации явно недостаточно для точной идентификации собеседника. Более того, если вчерашний собеседник был случайным, то без дополнительной информации его и не вспомнят.

В качестве другого примера можно рассмотреть ситуацию, когда ваша однокурсница появляется в институте с совершенно новой прической и в незнакомой вам одежде. При этом вы, тем не менее, чаще всего ее узнаете и сможете определить чем ее новый образ отличается от привычного. Можно предположить, что это происходит следующим образом. При виде ее нового облика в вашей памяти возникает ассоциация с привычным для вас. А далее сравнивая эти два облика вы можете определить отличия.

Исходя из рассмотренных примеров можно сказать, что ассоциативная память позволяет по неполной и даже частично недостоверной информации восстановить достаточно полное описание *знакомого* объекта. Слово *знакомого* является очень важным, поскольку невозможно вызвать ассоциации с незнакомыми объектами. При этом объект должен быть знаком тому, у кого возникают ассоциации.

Одновременно рассмотренные примеры позволяют сформулировать решаемые ассоциативной памятью задачи:

1. Соотнести входную информацию со знакомыми объектами, и дополнить ее до точного описания объекта.
2. Отфильтровать из входной информации недостоверную, а на основании оставшейся решить первую задачу.

Очевидно, что под точным описанием объекта следует понимать всю информацию, которая доступна ассоциативной памяти. Вторая задача решается не поэтапно, а одновременно происходит соотнесение полученной информации с известными образцами и отсев недостоверной информации.

2. Постановка задачи

Пусть задан набор из m **эталонов** – n-мерных векторов $\left\{x^i\right\}$. Требуется построить сеть, которая при предъявлении на вход произвольного образа – вектора x – давала бы на выходе “наиболее похожий” эталон.

Всюду далее образы и, в том числе, эталоны - n-мерные векторы с координатами ± 1. Эталон, “наиболее похожий” на x - ближайший к x вектор x^i. Легко заметить, что это требование эквивалентно требованию максимальности скалярного произведения векторов x и x^i:

$$\left\|x - x^i\right\| = \|x\|^2 + \left\|x^i\right\|^2 - 2\left(x, x^i\right).$$

Первые два слагаемых в правой части совпадают для любых образов x и x^i, так как длины всех векторов-образов равны $\sqrt{n}$. Таким образом, задача поиска ближайшего образа сводится к поиску образа, скалярное произведение с которым максимально. Этот простой факт приводит к тому, что сравнивать придется линейные функции от образов, тогда как расстояние является квадратичной функцией.

3. Сети Хопфилда

Наиболее известной сетью ассоциативной памяти является сеть Хопфилда [1]. В основе сети Хопфилда лежит следующая идея – запишем систему дифференциальных уравнений для градиентной минимизации “энергии” H (функции Ляпунова). Точки равновесия такой системы находятся в точках минимума энергии. Функцию энергии будем строить из следующих соображений:

1. Каждый эталон должен быть точкой минимума.
2. В точке минимума все координаты образа должны иметь значения ± 1.

Функция

$$H = -\frac{1}{2}\sum_{i=1}^{m}\left(x, x^i\right)^2 + \alpha\sum_{j=1}^{n}\left(x_j^2 - 1\right)^2$$

не удовлетворяет этим требованиям строго, но можно предполагать, что первое слагаемое обеспечит притяжение к эталонам (для вектора x фиксированной длины максимум квадрата скалярного произведения $\left(x, x^i\right)^2$ достигается при $x=x_i$), а второе слагаемое $\sum_{j=1}^{n}\left(x_j^2-1\right)^2$ – приблизит к единице абсолютные величины всех координат точки минимума. Величина α характеризует соотношение между этими двумя требованиями и может меняться со временем.

Используя выражение для энергии, можно записать систему уравнений, описывающих функционирование сети Хопфилда:

$$\dot{x}_j = -\partial H / \partial x_j = \sum_{i=1}^{m}\left(x, x^i\right)x_j^i - 4\alpha\left(x_j^2-1\right)x_j . \qquad (1)$$

Сеть Хопфилда в виде (1) является сетью с непрерывным временем. Это, быть может, и удобно для некоторых вариантов аналоговой реализации, но для цифровых компьютеров лучше воспользоваться сетями, функционирующими в дискретном времени - шаг за шагом.

Построим сеть Хопфилда с дискретным временем. Сеть должна осуществлять преобразование входного вектора x так, чтобы выходной вектор x' был ближе к тому эталону, который является правильным ответом. Преобразование сети будем искать в следующем виде:

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} w_i x^i\right), \qquad (2)$$

где w_i – вес i-го эталона, характеризующий его близость к вектору x, Sign - нелинейный оператор, переводящий вектор с координатами y_i в вектор с координатами $\mathrm{sign}\, y_i$.

Функционирование сети. Сеть работает следующим образом:

1. На вход сети подается образ x, а на выходе снимается образ x'.
2. Если $x' \neq x$, то полагаем $x = x'$ и возвращаемся к шагу 1.
3. Полученный вектор x' является ответом.

Таким образом, ответ всегда является неподвижной точкой преобразования сети (2) и именно это условие (неизменность при обработке образа сетью) и является условием остановки.

Пусть i^* – номер эталона, ближайшего к образу x. Тогда, если выбрать веса пропорционально близости эталонов к исходному образу x, то следует ожидать, что образ x' будет ближе к эталону x^{i^*}, чем x, а после нескольких итераций он станет совпадать с эталоном x^{i^*}.

Наиболее простой сетью вида (2) является дискретный вариант сети Хопфилда с весами равными скалярному произведению эталонов на предъявляемый образ:

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} \left(x, x^i \right) x^i \right). \qquad (3)$$

О сетях Хопфилда известно, что они способны запомнить и точно воспроизвести "порядка $0.14n$ слабо скоррелированных образов". В этом высказывании содержится два ограничения:

- число эталонов не превосходит $0.14n$.
- эталоны слабо скоррелированны.

а б в г

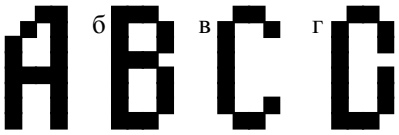


Рис. 1. а, б, с – эталоны, г – ответ сети на предъявление любого эталона

Наиболее существенным является второе ограничение, поскольку образы, которые сеть должна обрабатывать, часто очень похожи. Примером могут служить буквы латинского алфавита. При обучении сети Хопфилда распознаванию трех первых букв (см. рис. 1 а, б, в), при предъявлении на вход сети любого их эталонов в качестве ответа получается образ, приведенный на рис. 1 г.

В связи с такими примерами первый вопрос о качестве работы сети ассоциативной памяти звучит тривиально: будет ли сеть правильно обрабатывать сами эталонные образы (т.е. не искажает их)?

Зависимость работы сети Хопфилда от степени скоррелированности образов можно легко продемонстрировать на следующем примере. Пусть даны три эталона x^1, x^2, x^3 таких, что

$$\begin{aligned} &\left(x^1, x^2\right) + \left(x^1, x^3\right) > n, \\ &\left(x^1, x^2\right) + \left(x^2, x^3\right) > n, \\ &\left(x^1, x^3\right) + \left(x^2, x^3\right) > n, \\ &\left(x^i, x^j\right) > 0 \ (\forall i, j). \end{aligned} \tag{4}$$

Для любой координаты существует одна из четырех возможностей:

$$\begin{aligned} &1) \ \ x_l^i = x_l^j = 1 \ (\forall i, j), \\ &2) x_l^i = -x_l^j = x_l^k = 1, \\ &3) x_l^i = -x_l^j = x_l^k = \text{-}1, \\ &4) \ \ x_l^i = x_l^j = \text{-}1 \ \ (\forall i, j) \ . \end{aligned}$$

В первом случае при предъявлении сети q-го эталона в силу формулы (3) получаем $x_l' = Sign\left(\sum_{i=1}^{3}\left(x^q, x^i\right) \times 1\right) = 1$, так как все скалярные произведения положительны по условию (4). Аналогично получаем в четвертом случае $x_j' = -1$.

Во втором случае рассмотрим отдельно три варианта

$$\begin{aligned} x = x^i, \ \ x_j' &= Sign\left(-\left(x^i, x^i\right) + \left(x^i, x^j\right) + \left(x^i, x^k\right)\right) = 1 \\ x = x^j, \ \ x_l' &= Sign\left(-\left(x^j, x^i\right) + \left(x^j, x^j\right) + \left(x^j, x^k\right)\right) = 1 \\ x = x^k, \ \ x_l' &= Sign\left(-\left(x^k, x^i\right) + \left(x^k, x^j\right) + \left(x^k, x^k\right)\right) = 1 \end{aligned}$$

так как скалярный квадрат любого образа равен n, а сумма двух любых скалярных произведений эталонов больше n, по условию (4). Таким образом, независимо от предъявленного эталона получаем $x_j' = 1$. Аналогично в третьем случае получаем $x_j' = -1$.

Окончательный вывод таков: если эталоны удовлетворяют условиям (4), то при предъявлении любого эталона на выходе всегда будет один образ. Этот образ может быть эталоном или “химерой”, составленной, чаще всего, из узнаваемых фрагментов различных эталонов (примером “химеры” может служить образ, приведенный на рис. 1 г). Рассмотренный ранее пример с буквами детально иллюстрирует такую ситуацию.

Приведенные выше соображения позволяют сформулировать требование, детализирующие понятие “слабо скоррелированных образов”. Для правильного распознавания всех эталонов достаточно (но не необходимо) потребовать, чтобы выполнялось следующее неравенство $\sum_{\substack{i=1 \\ i\neq j}}^{m}\left|\left(x^i,x^j\right)\right|<n,\forall j$. Более простое и наглядное, хотя и более сильное условие можно записать в виде $\left|\left(x^i,x^j\right)\right|<\frac{n}{m},\forall i\neq j$. Из этих условий видно, что чем больше задано эталонов, тем более жесткие требования предъявляются к степени их скоррелированности, тем ближе они должны быть к ортогональным.

Рассмотрим преобразование (3) как суперпозицию двух преобразований:

$$Px=\sum_{i=1}^{m}\left(x,x^i\right)x^i,\qquad x'=\mathrm{Sign}\left(Px\right). \tag{5}$$

Обозначим через $L\left(\left\{x^i\right\}\right)=\left\{x\mid x=\sum_{i=1}^{m}\alpha_i x^i;\ \alpha_i\in \mathrm{R}\right\}$ – линейное пространство, натянутое на множество эталонов. Тогда первое преобразование в (5) переводит векторы из R^n в $L\left(\left\{x^i\right\}\right)$. Второе преобразование в (5) переводит результат первого преобразования Px в одну из вершин гиперкуба образов. Легко показать, что второе преобразование в (5) переводит точку Px в ближайшую вершину гиперкуба. Действительно, пусть a и b две различные вершины гиперкуба такие, что a – ближайшая к Px, а $b=x'$. Из того, что a и b различны следует, что существует множество индексов, в которых

координаты векторов a и b различны. Обозначим это множество через $I=\{i: a_i=-b_i\}$. Из второго преобразования в (5) и того, что $b=x'$, следует, что знаки координат вектора Px всегда совпадают со знаками соответствующих координат вектора b. Учитывая различие знаков i-х координат векторов a и Px при $i\in I$ можно записать $\left|a_i-(Px)_i\right|=\left|a_i\right|+\left|(Px)_i\right|=1+\left|(Px)_i\right|$. Совпадение знаков i-х координат векторов b и Px при $i\in I$ позволяет записать следующее неравенство $\left|b_i-(Px)_i\right|=\left|\left|a_i\right|-\left|(Px)_i\right|\right|<1+\left|(Px)_i\right|$. Сравним расстояния от вершин a и b до точки Px

$$\|b-Px\|=\sum_{i=1}^{n}\left(b_i-(Px)_i\right)^2=\sum_{i\in I}\left(b_i-(Px)_i\right)^2+\sum_{i\notin I}\left(b_i-(Px)_i\right)^2<$$

$$\sum_{i\in I}\left(1+\left|(Px)_i\right|\right)^2+\sum_{i\notin I}\left(a_i-(Px)_i\right)^2=\sum_{i=1}^{n}\left(a_i-(Px)_i\right)^2=\|a-Px\|.$$

Полученное неравенство $\|b-Px\|<\|a-Px\|$ противоречит тому, что a – ближайшая к Px. Таким образом доказано, что второе преобразование в (5) переводит точку Px в ближайшую вершину гиперкуба образов.

4. Ортогональные сети

Для обеспечения правильного воспроизведения эталонов достаточно потребовать, чтобы первое преобразование в (5) было таким, что $x^i=Px^i$. Очевидно, что если проектор является ортогональным, то это требование выполняется, поскольку $x=Px$ при $x\in L\left(\left\{x^i\right\}\right)$, а $x^j\in L\left(\left\{x^i\right\}\right)$ по определению множества $L\left(\left\{x^i\right\}\right)$.

Для обеспечения ортогональности проектора воспользуемся дуальным множеством векторов. Множество векторов $V\left(\left\{x^i\right\}\right)$ называется дуальным к

множеству векторов $\left\{x^i\right\}$ если все вектора этого множества v^j удовлетворяют следующим требованиям:

1. $\left(x^i, v^j\right) = \delta_{ij};\ \delta_{ij} = 0,$ при $i \neq j;\ \delta_{ij} = 1$ при $i = j;$

2. $v^j \in L\left(\left\{x^i\right\}\right)$.

Преобразование $Px = \sum_{i=1}^{m}\left(x, v^i\right)x^i, \quad v^i \in V\left(\left\{x^i\right\}\right)$ является ортогональным проектором на линейное пространство $L\left(\left\{x^i\right\}\right)$.

Ортогональная сеть ассоциативной памяти преобразует образы по формуле

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(x, v^i\right)x^i\right). \tag{6}$$

Дуальное множество векторов существует тогда и только тогда, когда множество векторов $\left\{x^i\right\}$ линейно независимо. Если множество эталонов $\left\{x^i\right\}$ линейно зависимо, то исключим из него линейно зависимые образы и будем рассматривать полученное усеченное множество эталонов как основу для построения дуального множества и преобразования (6). Образы, исключенные из исходного множества эталонов, будут по-прежнему сохраняться сетью в исходном виде (преобразовываться в самих себя). Действительно, пусть эталон x является линейно зависимым от остальных m эталонов. Тогда его можно представить в виде $x = \sum_{i=1}^{m}\alpha_i x^i$. Подставив полученное выражение в преобразование (6) и учитывая свойства дуального множества получим:

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(x, v^i\right)x^i\right) = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(\sum_{j=1}^{m}\alpha_j x^j, v^i\right)x^i\right) =$$
$$= \mathrm{Sign}\left(\sum_{i,j=1}^{m}\alpha_j\left(x^j, v^i\right)x^i\right) = \mathrm{Sign}\left(\sum_{j=1}^{m}\alpha_j x^j\right) = \mathrm{Sign}(x) = x \qquad (7)$$

Рассмотрим свойства сети (6) [2]. Во-первых, количество запоминаемых и точно воспроизводимых эталонов не зависит от степени их скоррелированности. Во-вторых, формально сеть способна работать без искажений при любом возможном числе эталонов (всего их может быть до 2^n). Однако, если число линейно независимых эталонов (т.е. ранг множества эталонов) равно n, сеть становится прозрачной – какой бы образ не предъявили на ее вход, на выходе окажется тот же образ. Действительно, как было показано в (7), все образы, линейно зависимые от эталонов, преобразуются проективной частью преобразования (6) сами в себя. Значит, если в множестве эталонов есть n линейно независимых, то любой образ можно представить в виде линейной комбинации эталонов (точнее n линейно независимых эталонов), а проективная часть преобразования (6) в силу формулы (7) переводит любую линейную комбинацию эталонов в саму себя.

Если число линейно независимых эталонов меньше *n*, то сеть преобразует поступающий образ, отфильтровывая помехи, ортогональные всем эталонам.

Отметим, что результаты работы сетей (3) и (6) эквивалентны, если все эталоны попарно ортогональны.

Остановимся несколько подробнее на алгоритме вычисления дуального множества векторов. Обозначим через $\Gamma\left(\left\{x^i\right\}\right)$ матрицу Грама множества векторов $\left\{x^i\right\}$. Элементы матрицы Грама имеют вид $\gamma_{ij} = \left(x^i, x^j\right)$ (ij-ый элемент матрицы Грама равен скалярному произведению i-го эталона на j-ый). Известно, что векторы дуального множества можно записать в следующем виде:

$$v^i = \sum_{j=1}^{m} \gamma_{ij}^{-1} x^j \,, \tag{8}$$

где γ_{ij}^{-1} – элемент матрицы $\Gamma^{-1}\left(\left\{x^i\right\}\right)$. Поскольку определитель матрицы Грама равен нулю, если множество векторов линейно зависимо, то матрица, обратная к матрице Грама, а следовательно и дуальное множество векторов существует только тогда, когда множество эталонов линейно независимо.

Для работ сети (6) необходимо хранить эталоны и матрицу $\Gamma^{-1}\left(\left\{x^i\right\}\right)$.

Рассмотрим процедуру добавления нового эталона к сети (6). Эта операция часто называется дообучением сети. Важным критерием оценки алгоритма формирования сети является соотношение вычислительных затрат на обучение и дообучение. Затраты на дообучение не должны зависеть от числа освоенных ранее эталонов.

Для сетей Хопфилда это, очевидно, выполняется - добавление еще одного эталона сводится к прибавлению к функции *H* одного слагаемого $\left(x, x^{m+1}\right)^2$, а модификация связей в сети - состоит в прибавлении к весу *ij*-й связи числа $x_i^{m+1} x_j^{m+1}$ - всего n^2 операций.

Для рассматриваемых сетей с ортогональным проектированием также возможно простое дообучение. На первый взгляд, это может показаться странным - если добавляемый эталон линейно независим от старых эталонов, то вообще говоря необходимо пересчитать матрицу Грама и обратить ее. Однако симметричность матрицы Грама позволяет не производить заново процедуру обращения всей матрицы. Действительно, обозначим через G_m – матрицу Грама для множества из m векторов x^i; через E_m – единичную матрицу размерности $m \times m$. При обращении матриц методом Гаусса используется следующая процедура:

1. Запишем матрицу размерности $m \times 2m$ следующего вида: $\left(\mathrm{G}_m \middle| \mathrm{E}_m\right)$.

2. Используя операции сложения строк и умножения строки на ненулевое число преобразуем левую квадратную подматрицу к единичной. В результате получим $\left(\mathrm{E}_m \middle| \mathrm{G}_m^{-1}\right)$.

Пусть известна G_m^{-1} – обратная к матрице Грама для множества из m векторов x^i. Добавим к этому множеству вектор x^{m+1}. Тогда матрица для обращения матрицы G_{m+1} методом Гаусса будет иметь вид:

$$\left(\begin{array}{ccc|c} & & \left(x^1, x^{m+1}\right) & \\ & \mathrm{G}_m & \vdots & \\ & & \left(x^m, x^{m+1}\right) & \mathrm{E}_{m+1} \\ \left(x^1, x^{m+1}\right) & \cdots \left(x^m, x^{m+1}\right) & \left(x^{m+1}, x^{m+1}\right) & \end{array}\right).$$

После приведения к единичной матрице главного минора ранга m получится следующая матрица:

$$\left(\begin{array}{ccc|ccc} & & b_1 & & & 0 \\ & \mathrm{E}_m & \vdots & & \mathrm{G}_m^{-1} & \vdots \\ & & b_m & & & 0 \\ \left(x^1, x^{m+1}\right) & \cdots \left(x^m, x^{m+1}\right) & \left(x^{m+1}, x^{m+1}\right) & 0 & \cdots 0 & 1 \end{array}\right),$$

где b_i – неизвестные величины, полученные в ходе приведения главного минора к единичной матрице. Для завершения обращения матрицы G_{m+1} необходимо привести к нулевому виду первые m элементов последней строки и $(m+1)$-о столбца. Для обращения в ноль i-о элемента последней строки необходимо умножить i-ю строку на $\left(x^i, x^{m+1}\right)$ и вычесть из последней строки. После проведения этого преобразования получим

$$\left(\begin{array}{ccc|ccc} & & b_1 & & & 0 \\ & \mathrm{E}_m & \vdots & & \mathrm{G}_m^{-1} & \vdots \\ & & b_m & & & 0 \\ 0 & \cdots 0 & b_0 & c_1 & \cdots c_m & 1 \end{array}\right),$$

где $b_0 = \left(x^{m+1}, x^{m+1}\right) - \sum_{i=1}^{m} \left(x^i, x^{m+1}\right) b_i$, $c_i = -\sum_{j=1}^{m} \left(x^j, x^{m+1}\right) \mathrm{G}^{-1}_{m,ji}$. $b_0 = 0$ только если новый эталон является линейной комбинацией первых m эталонов. Следовательно $b_0 \neq 0$. Для завершения обращения необходимо разделить последнюю строку на b_0 и затем вычесть из всех предыдущих строк последнюю, умноженную на соответствующее номеру строки b_i. В результате получим следующую матрицу

$$\left(\begin{array}{ccc|ccc} & & 0 & & & -b_1/b_0 \\ & \mathrm{E}_m & \vdots & & \mathrm{F} & \vdots \\ & & 0 & & & -b_m/b_0 \\ 0 & \cdots & 0 \quad 1 & c_1/b_0 & \cdots & c_m/b_0 \quad 1/b_0 \end{array}\right),$$

где $\mathrm{F}_{ij} = \mathrm{G}^{-1}_{m,ij} - b_i c_j / b_0$. Поскольку матрица, обратная к симметричной, всегда симметрична получаем $c_i/b_0 = -b_i/b_0$ при всех i. Так как $b_0 \neq 0$ следовательно $b_i = -c_i$.

Обозначим через d вектор $\left(\left(x^1, x^{m+1}\right), \ldots, \left(x^m, x^{m+1}\right)\right)$, через b – вектор $\left(b_1, \ldots, b_m\right)$. Используя эти обозначения можно записать $\mathrm{b} = \mathrm{G}^{-1}_m \mathrm{d}$, $b_0 = \left(x^{m+1}, x^{m+1}\right) - (\mathrm{d}, \mathrm{b})$. Матрица G^{-1}_{m+1} записывается в виде

$$\mathrm{G}^{-1}_{m+1} = \frac{1}{b_0} \begin{pmatrix} b_0 \mathrm{G}^{-1}_m + \mathrm{b} \otimes \mathrm{b} & -\mathrm{b} \\ -\mathrm{b} & 1 \end{pmatrix}.$$

Таким образом, при добавлении нового эталона требуется произвести следующие операции:

1. Вычислить вектор d (m скалярных произведений – mn операций, $mn \leq n^2$).
2. Вычислить вектор b (умножение вектора на матрицу – m^2 операций).
3. Вычислить b_0 (два скалярных произведения – $m + n$ операций).

4. Умножить матрицу на число и добавить тензорное произведение вектора b на себя ($2m^2$ операций).
5. Записать G_{m+1}^{-1}.

Таким образом, эта процедура требует $m+n+mn+3m^2$ операций. Тогда как стандартная схема полного пересчета потребует:

1. Вычислить всю матрицу Грама ($nm(m+1)/2$ операций).
2. Методом Гаусса привести левую квадратную матрицу к единичному виду ($2m^3$ операций).
3. Записать G_{m+1}^{-1}.

Всего $2m^3+nm(m+1)/2$ операций, что в m раз больше.

Используя ортогональную сеть (6), удалось добиться независимости способности сети к запоминанию и точному воспроизведению эталонов от степени скоррелированности эталонов. Так, например, ортогональная сеть смогла правильно воспроизвести все буквы латинского алфавита в написании, приведенном на рис. 1.

У сети (6) можно выделить два основных недостатка:

1. Число линейно независимых эталонов должно быть меньше размерности системы n.

2. Неинвариантностью - если два визуальных образа отличаются только своим положением в рамке, то в большинстве задач желательно объединять их в один эталон.

Оба этих недостатка можно устранить, изменив выбор весовых коэффициентов в (2).

5. Тензорные сети

Для увеличения числа линейно независимых эталонов, не приводящих к прозрачности сети, используется прием перехода к тензорным или многочастичным сетям [3-7].

Тензорным произведением k n-мерных векторов $y^1,\cdots,y^k$ называется k-индексная величина $b_{i_1\cdots i_k}$, у которой все индексы независимо пробегают весь набор значений от единицы до n, а $b_{i_1\cdots i_k}=y^1_{i_1}y^2_{i_2}\cdots y^k_{i_k}$. k-ой тензорной степенью вектора x будем называть вектор $x^{\otimes k}$, полученный как тензорное произведение k векторов x. Вектор $x^{\otimes k}$ является n^k-мерным вектором. Однако пространство $L\left(\left\{x^{i\otimes k}\right\}\right)$ имеет размерность, не превышающую величину $r_{n,k}=\sum_{i=0}^{k}\mathrm{C}^i_{n-1}$, где $\mathrm{C}^q_p=\frac{p!}{q!(p-q)!}$ – число сочетаний из p по q.

Теорема. При $k<n$ в ранг $r_{n,k}$ множества $\left\{x^{\otimes k}\right\}$ равен:

$$r_{n,k}=\sum_{i=0}^{k}\mathrm{C}^i_{n-1}.$$

Небольшая модернизация треугольника Паскаля, позволяет легко вычислять эту величину. На рис. 1 приведен “тензорный” треугольник Паскаля. При его построении использованы следующие правила:

2									
3	4								
4	7	8							
5	11	15	16						
6	16	26	31	32					
7	22	42	57	63	64				
8	29	64	99	120	127	128			
9	37	93	163	219	247	255	256		
10	46	130	256	382	466	502	511	512	
$r_{1,1}$	$r_{1,2}$	$r_{1,3}$	$r_{1,4}$	$r_{1,5}$	$r_{1,6}$	$r_{1,7}$	$r_{1,8}$	$r_{1,9}$	$r_{1,10}$

Рис. 1. “Тензорный” треугольник Паскаля

1. Первая строка содержит двойку, поскольку при n=2 в множестве X всего два неколлинеарных вектора.

2. При переходе к новой строке, первый элемент получается добавлением единицы к первому элементу предыдущей строки, второй – как сумма первого и второго элементов предыдущей строки, третий – как сумма второго и третьего элементов и т.д. Последний элемент получается удвоением последнего элемента предыдущей строки.

Таблица 1.

n	k	n^k	C_{n+k-1}^{k-1}	$r_{n,k}$
5	2	25	15	11
	3	125	35	15
10	3	1 000	220	130
	6	1 000 000	5005	466
	8	100 000 000	24310	511

В табл. 1 приведено сравнение трех оценок информационной емкости тензорных сетей для некоторых значений n и k. Первая оценка - n^k - заведомо завышена, вторая - C_{n+k-1}^{k-1} - дается формулой Эйлера для размерности пространства симметричных тензоров и третья - точное значение $r_{n,k}$

Как легко видеть из таблицы, уточнение при переходе к оценке r_{nk} является весьма существенным. С другой стороны, предельная информационная емкость тензорной сети (число правильно воспроизводимых образов) может существенно превышать число нейронов, например, для 10 нейронов тензорная сеть валентности 8 имеет предельную информационную емкость 511.

Легко показать, что если множество векторов $\left\{x^i\right\}$ не содержит взаимно обратных, то размерность пространства $L\left(\left\{x^{i\otimes n}\right\}\right)$ равна числу векторов в множестве $\left\{x^i\right\}$. Сеть (2) для случая тензорных сетей имеет вид

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(x^{\otimes k}, x^{i\otimes k}\right)x^i\right) = \mathrm{Sign}\left(\sum_{i=1}^{m}\left(x, x^i\right)^k x^i\right), \qquad (9)$$

а ортогональная тензорная сеть

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} \left(x^{\otimes k}, v^i \right) x^i \right) = \mathrm{Sign}\left(\sum_{i=1}^{m} \sum_{j=1}^{m} \gamma_{ij}^{-1} \left(x, x^j \right)^k x^i \right), \qquad (10)$$

где γ_{ij}^{-1} – элемент матрицы $\Gamma^{-1}\left(\left\{ x^{i\otimes k} \right\} \right)$. Сеть (9) хорошо работает на слабо скоррелированных эталонах, а сеть (10) не чувствительна к степени скоррелированности эталонов.

6. Сети для инвариантной обработки изображений

Для того, чтобы при обработке переводить визуальные образов, отличающиеся только положением в рамке изображения, в один эталон, применяется следующий прием [7]. Преобразуем исходное изображение в некоторый вектор величин, не изменяющихся при сдвиге (вектор инвариантов). Простейший набор инвариантов дают автокорреляторы - скалярные произведения образа на сдвинутый образ, рассматриваемые как функции вектора сдвига.

В качестве примера рассмотрим вычисление сдвигового автокоррелятора для черно-белых изображений. Пусть дан двумерный образ S размером $p \times q = n$. Обозначим точки образа как s_{ij}. Элементами автокоррелятора $Ac(S)$ будут величины $a_{kl} = \sum_{i=1}^{p} \sum_{j=1}^{q} s_{ij} s_{i+k, j+l}$, где $s_{ij=0}$ при выполнении любого из неравенств $i < 1, i > p, j < 1, j > q$. Легко проверить, что автокорреляторы любых двух образов, отличающихся только расположением в рамке, совпадают. Отметим, что $a_{ij} = a_{-i,-j}$ при всех i, j, и $a_{ij} = 0$ при выполнении любого из неравенств $i < 1 - p, i > p - 1, j < 1 - q, j > q - 1$. Таким образом, можно считать, что размер автокоррелятора равен $p \times (2q + 1)$.

Автокорреляторная сеть имеет вид

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} \left(Ac(x), Ac\left(x^i \right) \right) x^i \right). \qquad (11)$$

Сеть (11) позволяет обрабатывать различные визуальные образы, отличающиеся только положением в рамке, как один образ.

Подводя итоги, можно сказать, что все сети ассоциативной памяти типа (2) можно получить, комбинируя следующие преобразования:

1. Произвольное преобразование. Например, переход к автокорреляторам, позволяющий объединять в один выходной образ все образы, отличающиеся только положением в рамке.
2. Тензорное преобразование, позволяющее сильно увеличить способность сети запоминать и точно воспроизводить эталоны.
3. Переход к ортогональному проектору, снимающий зависимость надежности работы сети от степени скоррелированности образов.

Наиболее сложная сеть будет иметь вид:

$$x' = \mathrm{Sign}\left(\sum_{i=1}^{m} \sum_{j=1}^{m} \gamma_{ij}^{-1} \left(F(x), F\left(x^{j}\right) \right)^{k} x^{i} \right), \qquad (12)$$

где γ_{ij}^{-1} – элементы матрицы, обратной матрице Грама системы векторов $\Gamma^{-1}\left(\left\{ F\left(x^{i \otimes k} \right) \right\} \right)$, $F(x)$ – произвольное преобразование.

7. Численный эксперимент

Работа ортогональных тензорных сетей при наличии помех сравнивалась с возможностями линейных кодов, исправляющих ошибки. Линейным кодом, исправляющим k ошибок, называется линейное подпространство в n-мерном пространстве над GF_2, все вектора которого удалены друг от друга не менее чем на $2k+1$ (см., например, [8]). Линейный код называется совершенным, если для любого вектора n-мерного пространства существует кодовый вектор, удаленный от данного не более, чем на k. Тензорной сети в качестве эталонов подавались все кодовые векторы избранного для сравнения кода. Численные эксперименты с совершенными кодами показали, что тензорная сеть минимально необходимой валентности правильно декодирует все векторы. Для несовершенных кодов

картина оказалась хуже – среди устойчивых образов тензорной сети появились “химеры” – векторы, не принадлежащие множеству эталонов.

В случае n=10, k=1 (см. табл. 2 и 3, строка 1) при валентностях 3 и 5 тензорная сеть работала как единичный оператор – все входные вектора передавались на выход сети без изменений. Однако уже при валентности 7 число химер резко сократилось и сеть правильно декодировала более 60% сигналов. При этом были правильно декодированы все векторы, удаленные от ближайшего эталона на расстояние 2, а часть векторов, удаленных от ближайшего эталона на расстояние 1, остались химерами. В случае n=10, k=2 (см. табл. 2 и 3, строки 3, 4, 5) наблюдалось уменьшение числа химер с ростом валентности, однако часть химер, удаленных от ближайшего эталона на расстояние 2 сохранялась. Сеть правильно декодировала более 50% сигналов. Таким образом при малых размерностях и кодах, далеких от совершенных, тензорная сеть работает довольно плохо. Однако, уже при n=15, k=3 и валентности, большей 3 (см. табл. 2 и 3, строки 6, 7), сеть правильно декодировала все сигналы с тремя ошибками. В большинстве экспериментов число эталонов было больше числа нейронов.

Подводя итог, можно сказать, что качество работы сети возрастает с ростом размерности пространства и валентности и по эффективности устранения ошибок сеть приближается к коду, гарантированно исправляющему ошибки.

Работа выполнена при поддержке Красноярского краевого фонда науки, грант 6F0124.

Литература

1. Hopfield J.J. Neural networks and physical systems with emergent collective computational abilities // Proc. Nat. Acad. Sci. USA. 1982. Vol. 79. P.2554-2558.
2. Горбань А.Н. Обучение нейронных сетей. М.: изд-во СССР-США СП “ParaGraph”, 1990. 160 с.
3. Коноплев В.А., Синицын Е.В. “Моделирование нейронных сетей с максимально высокой информационной емкостью” // Тез. докл. III Всероссийского семинара “Нейроинформатика и ее приложения”. Красноярск: изд. КГТУ, 1995 г., с. 66.

4. Golub D.N. Gorban A.N. Multi-Particle Networks for Associative Memory // Proc. of the World Congress on Neural Networks, Sept. 15-18, 1996, San Diego, CA, Lawrence Erlbaum Associates, 1996. P. 772-775.
5. Горбань А.Н., Миркес Е.М. Информационная емкость тензорных сетей // Тез. докл. IV Всероссийского семинара “Нейроинформатика и ее приложения”. Красноярск: изд. КГТУ, 1996 г., с. 22-23.
6. Горбань А.Н., Миркес Е.М. Помехоустойчивость тензорных сетей // Тез. докл. IV Всероссийского семинара “Нейроинформатика и ее приложения”. Красноярск: изд. КГТУ, 1996 г., с. 24-25.
7. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. Новосибирск: Наука, 1996.
8. Блейхут Р. Теория и практика кодов, контролирующих ошибки. М.: Мир. 1986. 576 с.

Глава 9.

Логически прозрачные нейронные сети и производство явных знаний из данных

Е.М.Миркес

Вычислительный центр СО РАН в г. Красноярске[1]

Производство явных знаний из накопленных данных - проблема, которая намного старше чем компьютеры. Обучаемые нейронные сети могут производить из данных скрытые знания: создается навык предсказания, классификации, распознавания образов и т.п., но его логическая структура обычно остается скрытой от пользователя. Проблема проявления (контрастирования) этой скрытой логической структуры решается в работе путем приведения нейронных сетей к специальному “логически прозрачному” разреженному виду.

Исследуются два вопроса, встающие перед каждым исследователем, решившим использовать нейронные сети: “Сколько нейронов необходимо для решения задачи?” и “Какова должна быть структура нейронной сети?” Объединяя эти два вопроса, мы получаем третий: “Как сделать работу нейронной сети понятной для пользователя (логически прозрачной) и какие выгоды может принести такое понимание?” Описаны способы получения логически прозрачных нейронных сетей. Приведен пример из области социально-политических предсказаний. Для определенности рассматриваются только нейронные сети, обучение которых есть минимизация оценок (ошибок) с использованием градиента. Градиент оценки вычисляется методом двойственности (его частный случай - метод обратного распространения ошибки).

1. Сколько нейронов нужно использовать?

При ответе на этот вопрос существует две противоположные точки зрения. Одна из них утверждает, что чем больше нейронов использовать, тем более

[1] 660036, Красноярск-36, ВЦК СО РАН, E-mail: amse@cckr.krasnoyarsk.su

надежная сеть получится. Сторонники этой позиции ссылаются на пример человеческого мозга. Действительно, чем больше нейронов, тем больше число связей между ними, и тем более сложные задачи способна решить нейронная сеть. Кроме того, если использовать заведомо большее число нейронов, чем необходимо для решения задачи, то нейронная сеть точно обучится. Если же начинать с небольшого числа нейронов, то сеть может оказаться неспособной обучиться решению задачи, и весь процесс придется повторять сначала с большим числом нейронов. Эта точка зрения (чем больше - тем лучше) популярна среди разработчиков нейросетевого программного обеспечения. Так, многие из них как одно из основных достоинств своих программ называют возможность использования любого числа нейронов.

Вторая точка зрения опирается на такое “эмпирическое” правило: чем больше подгоночных параметров, тем хуже аппроксимация функции в тех областях, где ее значения были заранее неизвестны. С математической точки зрения задачи обучения нейронных сетей сводятся к продолжению функции заданной в конечном числе точек на всю область определения. При таком подходе входные данные сети считаются аргументами функции, а ответ сети - значением функции. На рис. 1 приведен пример аппроксимации табличной функции полиномами 3-й (рис. 1.а) и 8-й (рис. 1.б) степеней. Очевидно, что аппроксимация, полученная с помощью полинома 3-ей степени больше соответствует внутреннему представлению о “правильной” аппроксимации. Несмотря на свою простоту, этот пример достаточно наглядно демонстрирует суть

X	1	2	3	4
F(X)	5	4	6	3

Рис. 1. Аппроксимация табличной функции

проблемы.

Второй подход определяет нужное число нейронов как минимально необходимое. Основным недостатком является то, что это, минимально необходимое число, заранее неизвестно, а процедура его определения путем постепенного наращивания числа нейронов весьма трудоемка. Опираясь на опыт работы группы НейроКомп в области медицинской диагностики [4,5,9], космической навигации и психологии [10] можно отметить, что во всех этих задачах ни разу не потребовалось более нескольких десятков нейронов.

Подводя итог анализу двух крайних позиций, можно сказать следующее: сеть с минимальным числом нейронов должна лучше (“правильнее”, более гладко) аппроксимировать функцию, но выяснение этого минимального числа нейронов требует больших интеллектуальных затрат и экспериментов по обучению сетей. Если число нейронов избыточно, то можно получить результат с первой попытки, но существует риск построить “плохую” аппроксимацию. Истина, как всегда бывает в таких случаях, лежит посередине: нужно выбирать число нейронов большим чем необходимо, но не намного. Это можно осуществить путем удвоения числа нейронов в сети после каждой неудачной попытки обучения. Однако существует более надежный способ оценки минимального числа нейронов - использование процедуры контрастирования [1]. Кроме того, процедура контрастирования позволяет ответить и на второй вопрос: какова должна быть структура сети.

2. Процедура контрастирования

Процедура контрастирования основана на оценке значимости весов связей в сети. Впервые процедура контрастирования нейронных сетей на основе показателей чувствительности описана одновременно в [1] и (существенно более частный вариант) в [2]. В книге [1] указаны основные цели контрастирования: упростить техническую реализацию сети и сделать навык сети более понятным - явизовать (сделать явным) знание, полученное сетью в ходе обучения.

Результаты экспериментов по контрастированию нейронных сетей опубликованы в [7,8]. Существуют также подходы, не использующие показатели чувствительности [3]. Уже в [1] описано несколько способов вычисления показателей чувствительности. Приведем два наиболее широко используемых.

2.1. Контрастирование на основе оценки

Рассмотрим сеть, правильно решающую все примеры обучающего множества. Обозначим через w_p, $p = 1,\text{K}\ ,n$ веса всех связей. При обратном функционировании сети по принципу двойственности или методу обратного распространения ошибки сеть вычисляет вектор градиента функции оценки H по весам связей - $\text{Grad}(H) = \left\{\partial H / \partial w_p\right\}_{p=1,\text{K}\ ,n}$. Пусть w^0 - текущий набор весов связей, а оценка текущего примера равна H^0. Тогда в линейном приближении можно записать функцию оценки в точке w как $H(w) = H^0 + \sum_{p=1}^{n} \frac{\partial H}{\partial w_p}(w_p - w_p^0)$. Используя это приближение можно оценить изменение оценки при замене w_p^0 на как $\chi(p,q) = \left|\frac{\partial H}{\partial w_p}\right| \cdot \left|w_p^* - w_p^0\right|$, где q - номер примера обучающего множества, для которого были вычислены оценка и градиент. Величину $\chi(p,q)$ будем называть показателем чувствительности к замене w_p на w_p^* для примера q. Далее необходимо вычислить показатель чувствительности, не зависящий от номера примера. Для этого можно воспользоваться любой нормой. Обычно используется равномерная норма (максимум модуля): $\chi(p) = \max_q \chi(p,q)$. Умея вычислять показатели чувствительности, можно приступать к процедуре контрастирования. Приведем простейший вариант этой процедуры:

1. Вычисляем показатели чувствительности.
2. Находим минимальный среди показателей чувствительности - χ_{p^*}.

3. Заменим соответствующий этому показателю чувствительности вес $w^0_{p^*}$ на $w^*_{p^*}$, и исключаем его из процедуры обучения.
4. Предъявим сети все примеры обучающего множества. Если сеть не допустила ни одной ошибки, то переходим ко второму шагу процедуры.
5. Пытаемся обучить отконтрастированную сеть. Если сеть обучилась безошибочному решению задачи, то переходим к первому шагу процедуры, в противном случае переходим к шестому шагу.
6. Восстанавливаем сеть в состояние до последнего выполнения третьего шага. Если в ходе выполнения шагов со второго по пятый был отконтрастирован хотя бы один вес, (число обучаемых весов изменилось), то переходим к первому шагу. Если ни один вес не был отконтрастирован, то получена минимальная сеть.

Возможно использование различных обобщений этой процедуры. Например, контрастировать за один шаг процедуры не один вес, а заданное пользователем число весов. Наиболее радикальная процедура состоит в контрастировании половины весов связей. Если половину весов отконтрастировать не удается, то пытаемся отконтрастировать четверть и т.д. Отметим, что при описанном методе вычисления показателей чувствительности, предполагается возможным вычисление функции оценки и проведения процедуры обучения сети, а также предполагается известным обучающее множество. Возможен и другой путь.

2.2. Контрастирование без ухудшения

Пусть нам дана только обученная нейронная сеть и обучающее множество. Допустим, что вид функции оценки и процедура обучения нейронной сети неизвестны. В этом случае так же возможно контрастирование сети. Предположим, что данная сеть идеально решает задачу. Тогда нам необходимо так отконтрастировать веса связей, чтобы выходные сигналы сети при решении всех задач изменились не более чем на заданную величину. В этом случае

контрастирование весов производится понейронно. На входе каждого нейрона стоит адаптивный сумматор, который суммирует входные сигналы нейрона, умноженные на соответствующие веса связей. Для нейрона наименее чувствительным будет тот вес, который при решении примера даст наименьший вклад в сумму. Обозначив через x_p^q входные сигналы рассматриваемого нейрона при решении *q*-го примера получаем формулу для показателя чувствительности весов: $\chi(p,q) = \left|\left(w_p - w_p^*\right) \cdot x_p^q\right|$. Аналогично ранее рассмотренному получаем $\chi(p) = \left|\left(w_p - w_p^*\right)\right| \cdot \max_q \left|x_p^q\right|$. В самой процедуре контрастирования есть только одно отличие - вместо проверки на наличие ошибок при предъявлении всех примеров проверяется, что новые выходные сигналы сети отличаются от первоначальных не более чем на заданную величину.

3. Логически прозрачные нейронные сети

Одним из основных недостатков нейронных сетей, с точки зрения многих пользователей, является то, что нейронная сеть решает задачу, но не может рассказать как. Иными словами из обученной нейронной сети нельзя извлечь алгоритм решения задачи. Однако специальным образом построенная процедура контрастирования позволяет решить и эту задачу.

Зададимся классом сетей, которые будем считать логически прозрачными (то есть такими, которые решают задачу понятным для нас способом, для которого легко сформулировать словесное описания в виде явного алгоритма). Например потребуем, чтобы все нейроны имели не более трех входных сигналов.

Зададимся нейронной сетью у которой все входные сигналы подаются на все нейроны входного слоя, а все нейроны каждого следующего слоя принимают выходные сигналы всех нейронов предыдущего слоя. Обучим сеть безошибочному решению задачи.

После этого будем производить контрастирование в несколько этапов. На первом этапе будем контрастировать только веса связей нейронов входного слоя.

Если после контрастирования у некоторых нейронов осталось больше трех входных сигналов, то увеличим число входных нейронов. Затем аналогичную процедуру произведем поочередно для всех остальных слоев. После завершения описанной процедуры будет получена логически прозрачная сеть. Можно произвести дополнительное контрастирование сети, чтобы получить минимальную сеть. На рис. 2 приведены восемь минимальных сетей. Если под логически прозрачными сетями понимать сети, у которых каждый нейрон имеет не более трех входов, то все сети кроме пятой и седьмой являются логически прозрачными. Пятая и седьмая сети демонстрируют тот факт, что минимальность сети не влечет за собой логической прозрачности.

В качестве примера приведем интерпретацию алгоритма рассуждений, полученного по второй сети приведенной на рис. 2. Постановка задачи: по ответам на 12 вопросов необходимо предсказать победу правящей или оппозиционной партии. Ниже приведен список вопросов.

1. Правящая партия была у власти более одного срока?
2. Правящая партия получила больше 50% голосов на прошлых выборах?
3. В год выборов была активна третья партия?
4. Была серьезная конкуренция при выдвижении от правящей партии?
5. Кандидат от правящей партии был президентом в год выборов?
6. Был ли год выборов временем спада или депрессии?
7. Был ли рост среднего национального валового продукта на душу населения больше 2.1%?
8. Произвел ли правящий президент существенные изменения в политике?
9. Во время правления были существенные социальные волнения?
10. Администрация правящей партии виновна в серьезной ошибке или скандале?
11. Кандидат от правящей партии - национальный герой?
12. Кандидат от оппозиционной партии - национальный герой?

Ответы на вопросы описывают ситуацию на момент, предшествующий выборам. Ответы кодировались следующим образом: “да” - единица, “нет” -

минус единица. Отрицательный сигнал на выходе сети интерпретируется как предсказание победы правящей партии. В противном случае ответом считается победа оппозиционной партии. Все нейроны реализовывали пороговую функцию, равную 1, если алгебраическая сумма входных сигналов нейрона больше либо равна 0, и -1 при сумме меньшей 0. Ответ сети базируется на проявлениях двух синдромов: синдрома политической нестабильности (сумма ответов на вопросы 3, 4 и 9) и синдрома плохой политики (ответы на вопросы 4, 8 и 6). Заметим что симптом несогласия в правящей партии вошел в оба синдрома. Таким образом, для победы правящей партии необходимо отсутствие (-1) обоих синдромов.

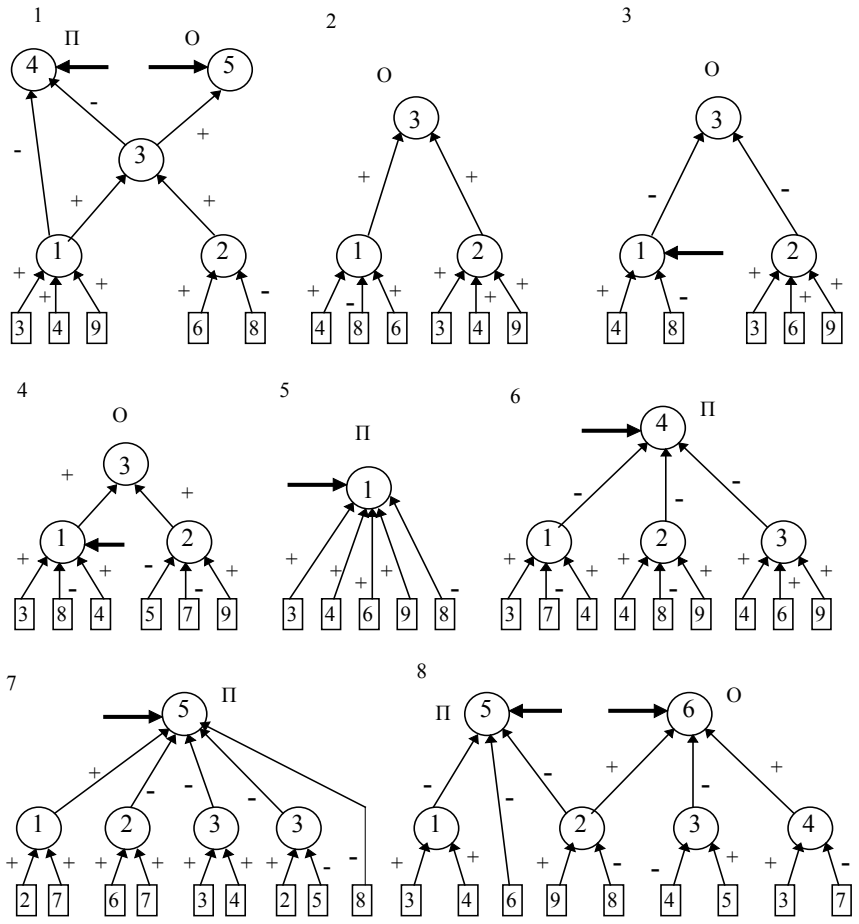


Рис. 2. Набор минимальных сетей для решения задачи о предсказании результатов выборов президента США. В рисунке использованы следующие обозначения: буквы “П” и “О” - обозначают вид ответа, выдаваемый нейроном: “П” - положительный сигнал означает победу правящей партии, а отрицательный - оппозиционной; “О” - положительный сигнал означает победу оппозиционной партии, а отрицательный - правящей.

На рис. 2 приведены структуры шести логически прозрачных нейронных сетей,

решающих задачу о предсказании результатов выборов президента США [6,11]. Все сети, приведенные на этом рисунке минимальны в том смысле, что из них нельзя удалить ни одной связи так, чтобы сеть могла обучиться правильно решать задачу. По числу нейронов минимальна пятая сеть.

Заметим, что все попытки авторов обучить нейронные сети со структурами, изображенными на рис. 2, и случайно сгенерированными начальными весами связей закончились провалом. Все сети, приведенные на рис. 2, были получены из существенно больших сетей с помощью процедуры контрастирования. Сети 1, 2, 3 и 4 были получены из трехслойных сетей с десятью нейронами во входном и скрытом слоях. Сети 5, 6, 7 и 8 были получены из двухслойных сетей с десятью нейронами во входном слое. Легко заметить, что в сетях 2, 3, 4 и 5 изменилось не только число нейронов в слоях, но и число слоев. Кроме того, почти все веса связей во всех восьми сетях равны либо 1, либо -1.

4. Заключение

Технология получения явных знаний из данных с помощью обучаемых нейронных сетей выглядит довольно просто и вроде бы не вызывает проблем - необходимо ее просто реализовывать и пользоваться.

Первый этап: обучаем нейронную сеть решать базовую задачу. Обычно базовой является задача распознавания, предсказания (как в предыдущем разделе) и т.п. В большинстве случаев ее можно трактовать как ***задачу о восполнении пробелов в данных***. Такими пробелами являются и имя образа при распознавании, и номер класса, и результат прогноза, и др.

Второй этап: с помощью анализа показателей значимости, контрастирования и доучивания (все это применяется, чаще всего, неоднократно) приводим нейронную сеть к логически прозрачному виду - так, чтобы полученный навык можно было “прочитать”.

Полученный результат неоднозначен - если стартовать с другой начальной карты, то можно получить другую логически прозрачную структуру. ***Каждой базе данных отвечает несколько вариантов явных знаний.*** Можно считать это недостатком технологии, но мы полагаем, что, наоборот, технология, дающая единственный вариант явных знаний, недостоверна, а ***неединственность результата является фундаментальным свойством производства явных знаний из данных.***

Работа выполнена при поддержке Красноярского краевого фонда науки, грант 6F0124.

ЛИТЕРАТУРА

1. Горбань А.Н. Обучение нейронных сетей. М.": изд. СССР-США СП "ParaGraph", 1990. 160 с. (English Translation: AMSE Transaction, Scientific Siberian, A, 1993, Vol. 6. Neurocomputing, PP. 1-134).
2. Le Cun Y., Denker J.S., Solla S.A. Optimal Brain Damage // Advances in Neural Information Processing Systems II (Denver 1989). San Mateo, Morgan Kaufman, pp. 598-605 (1990)
3. Prechelt L. Comparing Adaptive and Non-Adaptive Connection Pruning With Pure Early Stopping // Progress in Neural Information Processing (Hong Kong, September 24-27, 1996), Springer, Vol. 1 pp. 46-52.
4. Gilev S.E., Gorban A.N., Kochenov D.A., Mirkes Ye.M., Golovenkin S.E., Dogadin S.A., Maslennikova E.V., Matyushin G.V., Nozdrachev K.G., Rossiev D.A., Shulman V.A., Savchenko A.A. "MULTINEURON" neural simulator and its medical applications // Proceedings of the International Conference on Neural Information Processing (Oct. 17-20, Seoul, Korea). V. 2. PP. 1261-1266.
5. Rossiev et al, The Employment of Neural-Network Classifier for Diagnostics of Different phases of Immunodeficiency // Modelling, Measurment & Control, C. Vol.42, No.2, 1994. PP. 55-63.

6. Gorban A.N., Waxman C. How many neurons are sufficient to elect the U.S.A. President? // AMSE Transaction, Scientific Siberian, A, 1993. Vol. 6. Neurocomputing. PP. 168-188
7. Gordienko P. Construction of efficient neural networks: Algorithms and tests // Proceedings of the International joint Conference on Neural Networks IJCNN'93, Nagoya, Japan, 1993. PP. 313-316.
8. Еремин Д.И. Контрастирование // Нейропрограммы/ под. ред. А.Н.Горбаня. Красноярск: изд. КГТУ, 1994. С. 88-108
9. Gorban A.N., Rossiyev D.A., Butakova E.V., Gilev S.E., Golovenkin S.E., Dogadin S.A., Kochenov D.A., Maslennikova E.V., Matyushin G.V., Mirkes Ye.M., Nazarov B.V., Nozdrachev K.G., Savchenko A.A., Smirnova S.V., Shulman V.A. Medical and Physiological Applications of MultiNeuron Neural Simulator. Proceedings of the WCNN'95 (World Congress on Neural Networks'95, Washington DC, July 1995). PP. 170-175.
10. Dorrer M.G., Gorban A.N., Kopytov A.G., Zenkin V.I. Psychological Intuition of Neural Networks. Proceedings of the WCNN'95 (World Congress on Neural Networks'95, Washington DC, July 1995). PP. 193-196.
11. Gorban A.N., Waxman Cory, Neural Networks For Political Forecast. Proceedings of the WCNN'95 (World Congress on Neural Networks'95, Washington DC, July 1995). PP. 176-178.
12. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. Новосибирск: Наука, 1996. 276 с.