Ю.М. НЕФЁДОВ

# ОСНОВЫ МОДЕЛИРОВАНИЯ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

ЛУГАНСК 2012

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ,
МОЛОДЕЖИ И СПОРТА УКРАИНЫ
ВОСТОЧНОУКРАИНСКИЙ НАЦИОНАЛЬНЫЙ
УНИВЕРСИТЕТ ИМЕНИ ВЛАДИМИРА ДАЛЯ**

***Ю.М. НЕФЁДОВ***

# ОСНОВЫ МОДЕЛИРОВАНИЯ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

***Учебное пособие***

Луганск 2012

УДК 519.7
ББК 004.338.8
Н 58

Рекомендовано Вченою Радою СНУ ім. В.Даля
як навчальний посібник

Відповідальний редактор:
*Кочевський А.О.,* доцент, завідувач кафедри прикладної математики Східноукраїнського національного університету імені Володимира Даля

Рецензенти:
*Малий В.В.*, к.т.н., доцент кафедри прикладної математики Східноукраїнського національного університету імені Володимира Даля.
*Гальченко В.Я.,* д.т.н., професор, завідувач кафедри медбіофізики, медінформатики, біостатистики Луганського державного медичного університету.

**Нефьодов Ю.М.**
Н 58 **Основи моделювання інтелектуальних систем:**
Навчальний посібник. – Луганськ: Вид-во СНУ ім. В.Даля, 2012. 140 с.

У навчальному посібнику викладено основні теоретичні положення та принципи побудови штучних нейронних мереж як самостійного напрямку в теорії інтелектуальних систем. Дається введення в теорію нечітких множин і нечіткої логіки, які являють собою сучасний апарат формалізації різних видів невизначеностей, що виникають при моделюванні широкого класу реальних об'єктів будь-якої природи. Теорія нечітких множин являється стратегічним інструментом як керування складними системами, так і інформаційною основою інтелектуальних систем. Окремий розділ присвячений питанням навчання штучних нейронних мереж та раціонального вибору їх архітектури.

Для студентів математичних та інженерно-технічних спеціальностей вузів.

ISBN 966-590-157-7

УДК 519.7
ББК 004.338.8

## ОГЛАВЛЕНИЕ

## ПРЕДИСЛОВИЕ

Предлагаемое учебное пособие является введением в теорию нечетких множеств, нечетких отношений и нечеткой логики. Этот раздел современной математики находит широкое применение во всех сферах человеческой деятельности, в том числе и в области моделирования систем искусственного интеллекта. Учебное пособие написано на основе курса лекций «Моделирование интеллектуальных систем», который в течение ряда лет читался автором в Восточноукраинском национальном университете им. В. Даля для студентов специальностей «Прикладная математика» и «Информатика».

В первой главе описаны исторические и философские аспекты построения систем искусственного интеллекта. Рассмотрены основные проблемы моделирования элементов искусственного интеллекта (распознавания образов, использования естественного языка, компьютерное творчество, экспертные системы и др.).

Вторая глава содержит теоретические основы, связанные с нечеткими множествами и их свойствами. Разобраны подходы к решению задач с нечетко определенной информацией.

Третья глава посвящена нечетким отношениям на нечетких множествах. Рассмотрены вопросы композиции и декомпозиции нечетких отношений, а также транзитивного замыкания нечетких бинарных отношений.

В четвертой главе введено понятие нечеткой (или многозначной) логики. Нечеткая булева логика существенно расширяет возможности моделирования логических функций, анализа и синтеза этих функций. А это позволяет строить математические модели, близкие к рассуждениям человека, т.е. к естественному интеллекту.

Пятая глава содержит материал, связанный с нейронными сетями. При этом не имеет значения, нейронная сеть реализована аппаратурно или является компьютерной эмуляцией. Рассмотрены подходы к выбору архитектуры сети, методы обучения и тестирования сети, а также применение нейронных сетей.

Для усвоения материала книги достаточно знаний в пределах курсов математического анализа и линейной алгебры университета.

# Глава 1

# ОСНОВНЫЕ ПОНЯТИЯ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА (ИИ)

## 1.1. Философские аспекты проблемы ИИ

Термин *интеллект* происходит от латинского intellectus – что означает ум, рассудок, разум; мыслительные способности человека. Соответственно *искусственный интеллект* (artificial intelligence) обычно определяется, как свойство автоматических систем брать на себя отдельные функции интеллекта человека, например, выбирать и принимать оптимальные решения на основе ранее полученного опыта и рационального анализа внешних воздействий.

Под интеллектом принято понимать способность мозга решать *интеллектуальные* задачи путем приобретения, запоминания и целенаправленного преобразования знаний в процессе обучения на опыте и адаптации к разнообразным обстоятельствам.

Для того чтобы пояснить, чем отличается интеллектуальная задача от просто задачи, необходимо ввести понятие алгоритма – одиного из основных терминов кибернетики.

Под *алгоритмом* понимают точное предписание о выполнении в определенном порядке системы операций для решения любой задачи из некоторого данного класса (множества) задач.

«*Алгоритм* – это конечный набор правил, который определяет последовательность операций для решения конкретного множества задач и обладает пятью важными чертами: конечность, определённость, ввод, вывод, эффективность». (Д. Э. Кнут)

Различные определения алгоритма в явной или неявной форме содержат следующий ряд общих требований. Эти требования можно сформулировать следующим образом:

*Дискретность* – алгоритм должен представлять процесс решения задачи как последовательное выполнение некоторых простых шагов. При этом для выполнения каждого шага алгоритма требуется конечный

отрезок времени, то есть преобразование исходных данных в результат осуществляется во времени дискретно.

*Детерминированность* – определённость. В каждый момент времени следующий шаг работы однозначно определяется состоянием системы. Таким образом, алгоритм выдаёт один и тот же результат (ответ) для одних и тех же исходных данных. В современной трактовке у разных реализаций одного и того же алгоритма должен быть изоморфный граф. С другой стороны, существуют вероятностные алгоритмы, в которых следующий шаг работы зависит от текущего состояния системы и генерируемого случайного числа. Однако при включении метода генерации случайных чисел в список «исходных данных», вероятностный алгоритм становится подвидом обычного.

*Понятность* – алгоритм для исполнителя должен включать только те команды, которые ему (исполнителю) доступны, которые входят в его систему команд.

*Конечность* – при корректно заданных исходных данных алгоритм должен завершать работу и выдавать результат за конечное число шагов. С другой стороны, вероятностный алгоритм может и никогда не выдать результат, но вероятность этого равна 0.

*Массовость* – алгоритм должен быть применим к разным наборам исходных данных.

*Результативность* – завершение алгоритма определёнными результатами.

Алгоритм содержит ошибки, если приводит к получению неправильных результатов, либо не даёт результатов вообще.

Алгоритм не содержит ошибок, если он даёт правильные результаты для любых допустимых исходных данных

В математике и кибернетике класс задач определенного типа считается решенным, когда для их решения установлен алгоритм. Нахождение алгоритмов является естественной целью человека при решении им разнообразных классов задач. Отыскание алгоритма связано с тонкими и сложными рассуждениями, требующими большой изобретательности и высокой квалификации. Принято считать, что подобного рода деятельность требует участия интеллекта человека. Поэтому задачи, связанные с отысканием алгоритма решения класса задач определенного типа, называются *интеллектуальными*.

Задачи, для которых существуют стандартные методы решения, исключаются из класса интеллектуальных. Примерами таких задач могут служить чисто вычислительные задачи: решение системы линейных алгебраических уравнений, численное интегрирование дифференциальных уравнений и т.д. Для решения подобного рода задач имеются стандартные алгоритмы, представляющие собой определенную последователь-

ность элементарных операций, которая может быть легко реализована в виде программы для вычислительной машины. В противоположность этому для широкого класса *интеллектуальных задач*, таких, как распознавание образов, игра в шахматы, доказательство теорем и т.п., напротив это формальное разбиение процесса поиска решения на отдельные элементарные шаги часто оказывается весьма затруднительным, даже если само их решение несложно.

Таким образом, можно перефразировать определение интеллекта как *универсальный сверхалгоритм*, который способен создавать алгоритмы решения конкретных задач.

Деятельность мозга, обладающего интеллектом, направленную на решение интеллектуальных задач, принято называть *мышлением*, или *интеллектуальной деятельностью*. Интеллект и мышление органически связаны с решением таких задач, как доказательство теорем, логический анализ, распознавание ситуаций, планирование поведения, игры и управление в условиях неопределенности. Характерными чертами интеллекта, проявляющимися в процессе решения задач, являются *способность к обучению*, *обобщению*, *накоплению опыта* (*знаний и навыков*) и *адаптации* к изменяющимся условиям в процессе решения задач. Благодаря этим качествам интеллекта мозг может решать разнообразные задачи, а также легко перестраиваться с решения одной задачи на другую. Таким образом, мозг, наделенный интеллектом, является универсальным средством решения широкого круга задач (в том числе неформализованных) для которых нет стандартных, заранее известных методов решения.

Следует отметить, что существуют и другие, чисто поведенческие (функциональные) определения интеллекта (А. Н. Колмогорова, А. Тьюринга и др.) [8].

Основной философской проблемой в области искусственного интеллекта является проблема *возможности или невозможности моделирования мышления человека.*

То, что раньше казалось вершиной человеческого творчества – игра в шахматы, распознавание зрительных и звуковых образов, синтез новых технических решений, на практике оказалось не сложным делом (теперь работы ведутся на уровне нахождения наиболее *оптимальных алгоритмов*). Теперь часто данные проблемы даже не относят к проблемам искусственного интеллекта. Есть надежда, что и полное моделирование мышления человека окажется возможным.

С проблемой моделирование мышления человека тесно связана проблема возможности *самовоспроизведения*.

Способность к самовоспроизведению долгое время считалась прерогативой живых организмов. Однако некоторые явления, происходящие в неживой природе (например, рост кристаллов, синтез сложных

молекул копированием т.п.), очень похожи на самовоспроизведение. В начале 50-х годов Дж. фон Нейман занялся основательным изучением самовоспроизведения и заложил основы математической теории *самовоспроизводящихся автоматов.* Так же он доказал теоретически возможность их создания.

Существуют также различные неформальные доказательства возможности самовоспроизведения. Для программистов, например, самым ярким доказательством является существование и самовоспроизведения компьютерных вирусов.

Принципиальная возможность автоматизации решения интеллектуальных задач с помощью ЭВМ обеспечивается свойством алгоритмической универсальности. *Алгоритмическая универсальность* ЭВМ означает, что на них можно программно реализовывать любые алгоритмы преобразования информации, – будь то вычислительные алгоритмы, алгоритмы управления, поиска доказательства теорем или композиции мелодий. При этом мы имеем в виду, что процессы, порождаемые этими алгоритмами, являются потенциально осуществимыми, т.е. что они осуществимы в результате конечного числа элементарных операций. Практическая осуществимость алгоритмов зависит от имеющихся в нашем распоряжении средств, которые могут меняться с развитием техники. Так, в связи с появлением быстродействующих ЭВМ стали практически осуществимыми и такие алгоритмы, которые ранее были только потенциально осуществимыми.

Однако анализ разнообразных задач привел математиков к замечательному открытию. Было строго доказано [8] существование таких типов задач, для которых невозможен единый эффективный алгоритм, решающий все задачи данного типа; в этом смысле невозможно решение задач такого типа и с помощью вычислительных машин. Этот факт способствует лучшему пониманию того, что могут делать машины и чего они не могут сделать. В самом деле, утверждение об алгоритмической неразрешимости некоторого класса задач является не просто признанием того, что такой алгоритм нам не известен и никем еще не найден. Такое утверждение представляет собой одновременно и прогноз на все будущие времена о том, что подобного рода алгоритм не существует.

При решении таких задач человек просто-напросто пользуется методом *случайного поиска* и тем самым расширяет множество доступных для себя элементарных операций (например, создает новые материалы, открывает новые месторождения или типы ядерных реакций).

Следующим философским вопросом искусственного интеллекта является *цель создания*. В принципе все, что делается в практической жизни, обычно определено *поисковыми инстинктами.* И, допустим, что человек сумел создать интеллект, превосходящий свой собственный

(пусть не качеством, так количеством). Какую роль в этом случае будет играть человек?

По-видимому, самым приемлемым ответом на этот вопрос является концепция «*усилителя интеллекта*». Например, человек не мог бы предсказать погоду без компьютеров, а при полетах космических кораблей с самого начала использовались бортовые счетно-решающие устройства. Тем более что человек уже давно использует «усилители силы» – понятие, во многом аналогичное «усилителю интеллекта». В качестве усилителей силы ему служат автомобили, краны, электродвигатели, прессы, пушки, самолеты и многое-многое другое.

Основным отличием «усилителя интеллекта» от «усилителя силы» является наличие *воли*. Если «усилитель силы» в принципе не может иметь желаний, то интеллектуальная система, вполне могла бы иметь свои желания, и поступать не так, как хотелось бы человеку. Таким образом, перед человеком встает еще одна проблема – *проблема безопасности*.

Данная проблема обсуждается и учеными, и писателями фантастами (еще со времен Карела Чапека, впервые употребившего термин «робот»). Например, у писателя-фантаста и ученого Айзека Азимова в его широко известных рассказах можно найти самое проработанное решение проблемы безопасности. Речь идет о так называемых *трех законах роботехники*:

1) Робот не может причинить вред человеку или своим бездействием допустить, чтобы человеку был причинен вред.

2) Робот должен повиноваться командам, которые ему дает человек, кроме тех случаев, когда эти команды противоречат первому закону.

3) Робот должен заботиться о своей безопасности, насколько это не противоречит первому и второму закону.

На первый взгляд подобные законы, при их полном соблюдении, должны обеспечить безопасность человечества. Однако при внимательном рассмотрении возникают некоторые вопросы.

Во-первых, законы сформулированы на человеческом языке, который не допускает простого их перевода в алгоритмическую форму, т.е. на какой-либо язык программирования. Как, например, перевести такой термин, как «причинить вред» или «допустить».

Далее предположим, что человек сумели переформулировать, данные законы на язык, который понимает автоматизированная система. Тогда интересно, что будет подразумевать «система искусственного интеллекта» под термином «вред» после долгих логических размышлений? Не решит ли она, что существование человека это сплошной вред (он курит, пьет, с годами стареет и теряет здоровье, страдает)? Таким образом, потребуется ввести некоторые дополнения, связанные с ценностью

жизни, свободой волеизъявления. Но это уже будут не те простые три закона, которые были сформулированы для робота.

Следующим вопросом будет такой. Что решит система искусственного интеллекта в ситуации, когда спасение одной жизни возможно только за счет другой? Особенно интересны те случаи, когда система не имеет полной информации о том, кто есть кто.

Однако, несмотря на перечисленные проблемы, данные законы являются довольно неплохим неформальным базисом проверки надежности системы безопасности для систем искусственного интеллекта.

Так что же, неужели нет надежной системы безопасности? Если отталкиваться от концепции усилителя интеллекта, то можно предложить следующий вариант.

Согласно многочисленным опытам, несмотря на то, что мы не знаем точно, за что отвечает каждый отдельный нейрон в человеческом мозге, многим из наших эмоций обычно соответствует возбуждение группы нейронов (нейронный ансамбль) во вполне предсказуемой области. Были также проведены обратные эксперименты, когда раздражение определенной области вызывало желаемый результат. Это могли быть эмоции радости, угнетения, страха, агрессивности. Это наводит на мысль, что в принципе мы вполне могли бы вывести степень «удовлетворенности» организма наружу. В то же время, практически все известные механизмы адаптации и самонастройки (в первую очередь имеются в виду технические системы), базируются на принципах типа «хорошо» – «плохо». В математической интерпретации это сведение какой-либо функции к максимуму или к минимуму. Теперь представим себе, что наш усилитель интеллекта в качестве такой функции использует измеренную прямо или косвенно, степень удовольствия мозга человека-хозяина. Если принять меры, чтобы исключить самодеструктивную деятельность в состоянии депрессии, а так же предусмотреть другие особые состояния психики, то получим следующее.

Поскольку предполагается, что нормальный человек, не будет наносить вред самому себе, и, без особой на то причины, другим, а усилитель интеллекта теперь является частью данного индивидуума (не обязательно физическая общность), то автоматически выполняются все 3 закона роботехники. При этом вопросы безопасности смещаются в область *психологии*, поскольку обученная система не будет делать ничего такого, чего бы ни хотел ее владелец.

## 1.2. История развития систем ИИ

Началом развития научного направления, связанного с «искусственным интеллектом», считается появление статьи А. Тьюринга «Com-

puting Machinery and Intelligence» (published in 1950 in «Mind») с вопросом «Могут ли машины мыслить?» (под таким названием статья вышла в русском переводе). Алан Матисон Тьюринг – выдающийся английский математик, логик, криптограф, оказавший существенное влияние на развитие информатики, считается основоположником не только вычислительной техники, но и искусственного интеллекта. Он полагал, что компьютеры, в конце концов, смогут мыслить как человек, и предложил проверку, оценивающую способность машины мыслить, известную как «тест Тьюринга»: «Компьютер можно считать разумным, если он способен заставить нас поверить, что мы имеем дело не с машиной, а с человеком». Однако, как показали последующие исследования, не всё так просто и однозначно. Проблем, связанных с искусственным интеллектом, становилось всё больше. Есть основания утверждать, что человек будет постоянно решать проблему искусственного интеллекта, постоянно сталкиваясь все с новыми проблемами. И, видимо, процесс этот бесконечен.

Исторически сложились *три основных направления* в моделировании искусственного интеллекта.

В рамках *первого направления* объектом исследований являются структура и механизмы работы мозга человека, а конечная цель заключается в раскрытии тайн мышления. Необходимыми этапами исследований в этом направлении являются построение моделей на основе психофизиологических данных, проведение экспериментов с ними, выдвижение новых гипотез относительно механизмов интеллектуальной деятельности, совершенствование моделей и т. д.

*Второй подход* в качестве объекта исследования рассматривает искусственный интеллект. Здесь речь идет о моделировании интеллектуальной деятельности с помощью вычислительных машин. Целью работ в этом направлении является создание алгоритмического и программного обеспечения вычислительных машин, позволяющего решать интеллектуальные задачи не хуже человека.

Наконец, *третье направление* ориентировано на создание смешанных человеко-машинных, или, как еще говорят, интерактивных интеллектуальных систем, на симбиоз возможностей естественного и искусственного интеллекта. Важнейшими проблемами в этих исследованиях является оптимальное распределение функций между естественным и искусственным интеллектом и организация диалога между человеком и машиной.

Самыми первыми интеллектуальными задачами, которые стали решаться при помощи ЭВМ, были логические игры (шашки, шахматы), доказательство теорем. Хотя, правда здесь надо отметить еще кибернетические игрушки типа «электронной мыши» Клода Шеннона, которая

управлялась сложной релейной схемой. Эта мышка могла «исследовать» лабиринт, и находить выход из него. Кроме того, помещенная в уже известный ей лабиринт, она не искала выход, а сразу же, не заглядывая в тупиковые ходы, выходила из лабиринта.

Американский кибернетик А. Самуэль составил для вычислительной машины программу, которая позволяет ей играть в шашки, причем в ходе игры машина обучается или, по крайней мере, создает впечатление, что обучается, улучшая свою игру на основе накопленного опыта. В 1962 г. эта программа сразилась с Р. Нили, сильнейшим шашистом в США и победила.

Каким образом машине удалось достичь столь высокого класса игры?

Естественно, что в машину были программно заложены правила игры так, что выбор очередного хода был подчинен этим правилам. На каждой стадии игры машина выбирала очередной ход из множества возможных ходов согласно некоторому критерию качества игры. В шашках (как и в шахматах) обычно невыгодно терять свои фигуры, и, напротив, выгодно брать фигуры противника. Игрок (будь он человек или машина), который сохраняет подвижность своих фигур и право выбора ходов и в то же время держит под боем большое число полей на доске, обычно играет лучше своего противника, не придающего значения этим элементам игры. Описанные критерии хорошей игры сохраняют свою силу на протяжении всей игры, но есть и другие критерии, которые относятся к отдельным ее стадиям – дебюту, миттельшпилю, эндшпилю.

Разумно сочетая такие критерии (например, в виде линейной комбинации с экспериментально подбираемыми коэффициентами или более сложным образом), можно для оценки очередного хода машины получить некоторый числовой показатель эффективности – оценочную функцию. Тогда машина, сравнив между собой показатели эффективности очередных ходов, выберет ход, соответствующий наибольшему показателю. Подобная автоматизация выбора очередного хода не обязательно обеспечивает оптимальный выбор, но это какой-то выбор, и на его основе машина может продолжать игру, совершенствуя свою стратегию (образ действия) в процессе обучения на прошлом опыте. Формально обучение состоит в подстройке параметров (коэффициентов) оценочной функции на основе анализа проведенных ходов и игр с учетом их исхода.

По мнению А. Самуэля, машина, использующая этот вид обучения, может научиться играть лучше, чем средний игрок, за относительно короткий период времени.

Можно сказать, что все эти элементы интеллекта, продемонстрированные машиной в процессе игры в шашки, сообщены ей автором про-

граммы. Отчасти это так. Но не следует забывать, что эта программа не является «жесткой», заранее продуманной во всех деталях. Она совершенствует свою стратегию игры в процессе самообучения. И хотя процесс «мышления» у машины существенно отличен оттого, что происходит в мозгу, играющего в шашки человека, она способна у него выиграть.

Ярким примером сложной интеллектуальной игры до недавнего времени являлись шахматы. В 1974 г. состоялся международный шахматный турнир машин, снабженных соответствующими программами. Как известно, победу на этом турнире одержала советская машина с шахматной программой «Каисса».

Почему здесь употреблено «до недавнего времени»? Дело в том, что недавние события показали, что, несмотря на довольно большую сложность шахмат, и невозможность, в связи с этим произвести полный перебор ходов, возможность перебора их на большую глубину, чем обычно, очень увеличивает шансы на победу. К примеру, по сообщениям в печати, компьютер фирмы IBM, победивший гроссмейстера Г. Каспарова, имел 256 процессоров, каждый из которых имел 4 Гб дисковой памяти и 128 Мб оперативной. Весь этот комплекс мог просчитывать более 100 млн. ходов в секунду. До недавнего времени редкостью был компьютер, могущий делать такое количество целочисленных операций в секунду, а здесь мы говорим о ходах, которые должны быть сгенерированы и для которых просчитаны оценочные функции. Хотя с другой стороны, этот пример говорит о мощности и универсальности переборных алгоритмов.

За последние годы прошли знаменательные шахматные матчи Человек-Компьютер, они получили большую огласку. Чемпион мира В.Крамник и чемпион мира прошлых лет Г.Каспаров сыграли матчи с различными шахматными программами (Fritz, Junior, Deep Blue). В них компьютер одержал 5 побед, 4 раза проиграл и 11 партий свёл вничью. Однако, с учетом роста мощности современных компьютеров, нельзя считать, что машины становятся умнее. Ведь нас не удивляет мощность экскаватора по сравнению с человеком с лопатой.

В настоящее время существуют и успешно применяются программы, позволяющие машинам играть в деловые или военные игры, имеющие большое прикладное значение. Здесь также чрезвычайно важно придать программам присущие человеку способность к обучению и адаптации. Одной из наиболее интересных интеллектуальных задач, также имеющей огромное прикладное значение, является задача обучения распознавания образов и ситуаций. Решением ее занимались и продолжают заниматься представители различных наук – физиологи, психологи, математики, инженеры. Такой интерес к задаче стимулировался

фантастическими перспективами широкого практического использования результатов теоретических исследований: читающие автоматы, системы искусственного интеллекта, ставящие медицинские диагнозы, проводящие криминалистическую экспертизу и т. п., а также роботы, способные распознавать и анализировать сложные ситуации.

В 1957 г. американский физиолог Ф. Розенблатт предложил модель зрительного восприятия и распознавания – *персептрон*. Появление машины, способной обучаться понятиям и распознавать предъявляемые объекты, оказалось чрезвычайно интересным не только физиологам, но и представителям других областей знания и породило большой поток теоретических и экспериментальных исследований.

Персептрон или любая программа, имитирующая процесс распознавания, работают в двух режимах: в режиме *обучения* и в режиме *распознавания*. В режиме обучения некто (человек, машина, робот или природа), играющий роль учителя, предъявляет машине объекты и о каждом их них сообщает, к какому понятию (классу) он принадлежит. По этим данным строится решающее правило, являющееся, по существу, формальным описанием понятий. В режиме распознавания машине предъявляются новые объекты (вообще говоря, отличные от ранее предъявленных объектов), и она должна их классифицировать, по возможности, правильно.

Проблема обучения распознаванию тесно связана с другой интеллектуальной задачей – проблемой перевода с одного языка на другой, а также обучения машины языку. При достаточно формальной обработке и классификации основных грамматических правил и приемов пользования словарем можно создать вполне удовлетворительный алгоритм для перевода, скажем научного или делового текста. Для некоторых языков такие системы были созданы еще в конце 60-г. Однако для того, чтобы связно перевести достаточно большой разговорный текст, необходимо понимать его смысл. Работы над такими программами ведутся уже давно, но до полного успеха еще далеко. Имеются также программы, обеспечивающие диалог между человеком и машиной на урезанном естественном языке.

Что же касается моделирования логического мышления, то хорошей модельной задачей здесь может служить задача автоматизации доказательства теорем. Начиная с 1960 г., был разработан ряд программ, способных находить доказательства теорем в исчислении предикатов первого порядка. Эти программы обладают, по словам американского специалиста в области искусственного интеллекта Дж. Маккатти, «здравым смыслом», т.е. способностью делать дедуктивные заключения.

В программе К. Грина и др., реализующей вопросно-ответную систему, знания записываются на языке логики предикатов в виде набора

аксиом, а вопросы, задаваемые машине, формулируются как подлежащие доказательству теоремы. Большой интерес представляет «интеллектуальная» программа американского математика Хао Ванга. Эта программа за 3 минуты работы IBM-704 вывела 220 относительно простых лемм и теорем из фундаментальной математической монографии, а затем за 8.5 мин. выдала доказательства еще 130 более сложных теорем, часть их которых еще не была выведена математиками. Правда, до сих пор ни одна программа не вывела и не доказала ни одной теоремы, которая бы, что называется «позарез» была бы нужна математикам и была бы принципиально новой.

Очень важным направлением систем искусственного интеллекта является робототехника. В чем основное отличие интеллекта робота от интеллекта универсальных вычислительных машин?

Для ответа на этот вопрос уместно вспомнить принадлежащее великому русскому физиологу И. М. Сеченову высказывание: «… все бесконечное разнообразие внешних проявлений мозговой деятельности сводится окончательно лишь к одному явлению – мышечному движению». Другими словами, вся интеллектуальная деятельность человека направлена, в конечном счете, на активное взаимодействие с внешним миром посредством движений. Точно так же элементы интеллекта робота служат, прежде всего, для организации его целенаправленных движений. В то же время основное назначение чисто компьютерных систем искусственного интеллекта состоит в решении интеллектуальных задач, носящих абстрактный или вспомогательный характер, которые обычно не связаны ни с восприятием окружающей среды с помощью искусственных органов чувств, ни с организацией движений исполнительных механизмов.

Первые роботы трудно назвать интеллектуальными. Только в 60-х годах появились очуствленные роботы, которые управлялись универсальными компьютерами. К примеру, в 1969 г. в Электротехнической лаборатории (Япония) началась разработка проекта «промышленный интеллектуальный робот». Цель этой разработки – создание очуствленного манипуляционного робота с элементами искусственного интеллекта для выполнения сборочно-монтажных работ с визуальным контролем.

Манипулятор робота имеет шесть степеней свободы и управляется мини-ЭВМ NEAC-3100 (объем оперативной памяти 32 тыс. слов, объем внешней памяти на магнитных дисках 273 тыс. слов), формирующей требуемое программное движение, которое отрабатывается следящей электрогидравлической системой. При этом схватывающее устройство манипулятора оснащено *тактильными датчиками*.

В качестве системы зрительного восприятия используются две телевизионные камеры, снабженные красно-зелено-синими фильтрами для

распознавания цвета предметов. Поле зрения телевизионной камеры разбито на $(64 \times 64)$ ячеек. В результате обработки полученной информации грубо определяется область, занимаемая интересующим робота предметом. Далее, с целью детального изучения этого предмета выявленная область вновь делится на 4096 ячеек. В том случае, когда предмет не помещается в выбранное «окошко», оно автоматически перемещается, подобно тому, как человек скользит взглядом по предмету. Робот Электротехнической лаборатории был способен распознавать простые предметы, ограниченные плоскостями и цилиндрическими поверхностями при специальном освещении. Стоимость данного экспериментального образца составляла примерно 400 тыс. долларов.

Постепенно характеристики роботов монотонно улучшались, но до сих пор они еще далеки по понятливости от человека, хотя некоторые операции уже выполняются на уровне лучших жонглеров. К примеру, робот удерживает на лезвии ножа шарик от настольного тенниса.

Можно также выделить работы киевского Института кибернетики, где под руководством Н. М. Амосова и В. М. Глушкова велся комплекс исследований, направленных на разработку элементов интеллекта роботов. Особое внимание в этих исследованиях уделялось проблемам распознавания изображений и речи, логического вывода (автоматического доказательства теорем) и управления с помощью сетей, подобных нейронным сетям.

К примеру, можно рассмотреть созданный еще в 70-х годах макет транспортного автономного интегрального робота (ТАИР). Конструктивно ТАИР представляет собой трехколесное шасси, на котором смонтирована сенсорная система и блок управления. Сенсорная система включает в себя следующие «органы чувств»: оптический дальномер, навигационная система с двумя радиомаяками и компасом, контактные датчики, датчики углов наклона тележки, таймер и др.

Особенность, которая отличает ТАИР от многих других систем, созданных у нас и за рубежом, это то, что в его составе нет компьютера в том виде, к которому мы привыкли. Основу системы управления составляет бортовая нейронная (подобная) сеть, на которой реализуются различные алгоритмы обработки сенсорной информации, планирования поведения и управления движением робота.

В конце данного очень краткого обзора рассмотрим примеры крупномасштабных экспертных систем.

*MICIN* – экспертная система для медицинской диагностики. Система разработана группой по инфекционным заболеваниям Стенфордского университета. Ставит соответствующий диагноз, исходя из представленных ей симптомов, и рекомендует курс медикаментозного лечения лю-

бой из диагностированных инфекций. База данных состоит из 450 правил.

*PUFF* – система анализа нарушения дыхания. Данная система представляет собой MICIN, из которой удалили данные по инфекциям и вставили данные о легочных заболеваниях.

*DENDRAL* – система распознавания химических структур. Данная система старейшая, из имеющихся экспертных систем. Первые версии данной системы появились еще в 1965 году во все том же Стенфордском университете. Пользователь дает системе DENDRAL некоторую информацию о веществе, а также данные спектрометрии (инфракрасной, ядерного магнитного резонанса и масс-спектрометрии), и та в свою очередь выдает диагноз в виде соответствующей химической структуры.

*PROSPECTOR* – экспертная система, созданная для содействия поиску коммерчески оправданных месторождений полезных ископаемых.

## 1.3. Различные подходы к построению систем ИИ

Существуют различные подходы к построению систем искусственного интеллекта. Это разделение не является историческим, когда одно мнение постепенно сменяет другое, и различные подходы существуют и сейчас. Какой подход является правильным, а какой ошибочным сказать невозможно, поскольку полных систем искусственного интеллекта в настоящее время не существует.

Для начала кратко рассмотрим логический подход. Почему он возник? Ведь человек занимается отнюдь не только логическими измышлениями. Это высказывание конечно верно, но именно способность к логическому мышлению очень сильно отличает человека от животных.

Основой для данного логического подхода служит Булева алгебра. Каждый программист знаком с нею и с логическими операторами с тех пор, когда он осваивал оператор IF. Свое дальнейшее развитие Булева алгебра получила в виде исчисления предикатов, в котором она расширена за счет введения предметных символов, отношений между ними, кванторов существования и всеобщности. Практически каждая система искусственного интеллекта, построенная на логическом принципе, представляет собой машину доказательства теорем. При этом исходные данные хранятся в базе данных в виде аксиом и правил логического вывода как отношения между ними. Кроме того, каждая такая машина имеет блок генерации цели, и система вывода пытается доказать данную цель как теорему. Если цель доказана, то трассировка примененных правил позволяет получить цепочку действий, необходимых для реализации поставленной цели. Мощность такой системы определяется возможностями генератора целей и машиной доказательства теорем.

Конечно, можно сказать, что выразительности алгебры высказываний не хватит для полноценной реализации искусственного интеллекта, но стоит вспомнить, что основой всех существующих ЭВМ является бит – ячейка памяти, которая может принимать значения только 0 и 1. Таким образом, было бы логично предположить, что все, что возможно реализовать на ЭВМ, можно было бы реализовать и в виде логики предикатов. Хотя здесь ничего не говорится о том, за какое время.

Добиться большей выразительности логическому подходу позволяет такое сравнительно новое направление, как *нечеткая логика*. Основным ее отличием является то, что правдивость высказывания может принимать в ней кроме «да» и «нет» (1 и 0) еще и промежуточные значения. Данный подход больше похож на мышление человека, поскольку он на вопросы редко отвечает только «да» или «нет». Хотя, правда, на экзамене будут приниматься только ответы из разряда классической булевой алгебры.

Для большинства логических методов характерна большая трудоемкость, поскольку во время поиска доказательства возможен полный перебор вариантов. Поэтому данный подход требует эффективной реализации вычислительного процесса, и хорошая работа обычно гарантируется при сравнительно небольшом размере базы данных.

Под *структурным подходом* подразумеваются попытки построения искусственного интеллекта путем моделирования структуры человеческого мозга. Одной из первых таких попыток был персептрон Френка Розенблатта. Основной моделируемой структурной единицей в персептронах (как и в большинстве других вариантов моделирования мозга) является *нейрон*.

Позднее возникли и другие модели, которые известны под термином «нейронные сети». Эти модели различаются по строению отдельных нейронов, по топологии связей между ними и по алгоритмам обучения. Среди наиболее известных сейчас вариантов нейронных сетей можно назвать нейронные сети с обратным распространением ошибки, сети Хопфилда, стохастические нейронные сети.

Нейронные сети наиболее успешно применяются в задачах распознавания образов, в том числе сильно зашумленных, однако имеются и примеры успешного применения их для построения собственно систем искусственного интеллекта, это уже упоминавшаяся система ТАИР.

Для моделей, построенных по мотивам человеческого мозга характерна не слишком большая выразительность, легкое распараллеливание алгоритмов, и связанная с этим высокая производительность параллельно реализованных нейронных сетей. Также для таких сетей характерно одно свойство, которое очень сближает их с человеческим мозгом – нейронные сети работают даже при условии неполной информации об ок-

ружающей среде, то есть, как и человек, они на вопросы могут отвечать не только «да» и «нет» но и «не знаю точно, но скорее да».

Довольно большое распространение получил и эволюционный подход. При построении систем искусственного интеллекта по данному подходу основное внимание уделяется построению начальной модели, и правилам, по которым она может изменяться (*эволюционировать*). Причем модель может быть составлена по самым различным методам, это может быть и нейронная сеть и набор логических правил и любая другая модель. После этого включается компьютер и он, на основании проверки моделей отбирает самые лучшие из них, на основании которых по самым различным правилам генерируются новые модели, из которых опять выбираются самые лучшие и т. д.

В принципе можно сказать, что эволюционных моделей как таковых не существует, существует только эволюционные алгоритмы обучения, но модели, полученные при эволюционном подходе, имеют некоторые характерные особенности, что позволяет выделить их в отдельный класс.

Такими особенностями являются перенесение основной работы разработчика с построения модели на алгоритм ее модификации и то, что полученные модели практически не сопутствуют извлечению новых знаний о среде, окружающей систему искусственного интеллекта, то есть она становится как бы вещью в себе.

Еще один широко используемый подход к построению систем искусственного интеллекта – *имитационный*. Данный подход является классическим для кибернетики с одним из ее базовых понятий – «черным ящиком». *Черный ящик* – устройство, программный модуль или набор данных, информация о внутренней структуре и содержании которых отсутствуют полностью, но известны спецификации входных и выходных данных. Объект, поведение которого имитируется, как раз и представляет собой такой «черный ящик». Не важно, что у него и у модели внутри и как он функционирует, главное, чтобы построенная модель в аналогичных ситуациях вела себя точно так же.

Таким образом, здесь моделируется другое свойство человека – способность копировать то, что делают другие, не вдаваясь в подробности, зачем это нужно. Зачастую эта способность экономит ему массу времени, особенно в начале его жизни.

Основным недостатком имитационного подхода также является низкая информационная способность большинства моделей, построенных с его помощью.

В заключение беглого рассмотрения различных методов и подходов к построению систем искусственного интеллекта, следует отметить, что на практике четкой границы между ними нет. Очень часто встреча-

ются смешанные системы, где часть работы выполняется по одному типу, а часть по другому.

## 1.4. Основные проблемы ИИ (решенные и нерешенные)

Работы в области искусственного интеллекта ведутся во многих направлениях. Следует упомянуть основные направления.

*Распознавание образов.* Это направление является одним из ранних направлений искусственного интеллекта. Оно связано с моделированием особенностей восприятия внешнего мира, узнавания объектов. В основе этого лежит тот факт, что все объекты могут быть проклассифицированы по определенным признакам и, следовательно, умение различать их проявление, и позволяет идентифицировать соответствующий объект.

Эта проблема касается распознавания зрительных, слуховых образов, а также образов других (смешанных) модальностей. Медицинская диагностика и предсказание погоды являются примерами задач распознавания образов, не связанных с какой-то конкретной модальностью. В последнее время большая часть работ в этой области ориентирована на *анализ сцен* (задача разделения дискретных объектов, восприятие трехмерного пространства ит.п.), а не только на распознавание отдельных объектов (например, печатных знаков), что важно для робототехники. Распознавание образов относится к высокоинтеллектуальной задаче, т.к. искусственное распознавание образов вводит компьютер в контакт с реальным миром. В этом направлении встречается множество как технических, так и смысловых трудностей.

*Использование естественного языка.* В этом направлении можно выделить самостоятельные направления.

Первое направление – создание систем «*вопрос-ответ*», т.е. сделать естественные языки средством общения человека с машиной. Попутно исследуются вопросы о природе мышления, о роли лингвистики в процессе мышления.

Второе направление – создание систем *автоматического перевода* с одного языка на другой. В этом направлении решаются синтаксические и семантические проблемы. Качество перевода зависит от семантики текста (интерпретация фраз зависит от знания предмета разговора, содержания текста, смысловой неоднозначности слов, идиомы и т.п.).

*Игры и машинное творчество.* Машинное творчество охватывает сочинение компьютерной музыки, стихов, автоматизацию изобретения новых объектов. Компьютерные игры являются той сферой искусственного интеллекта, которая наиболее знакома большинству пользователей. При реализации интеллектуальных игр (в том числе и при игре в шахматы) используются, как классические разделы математики (напри-

мер, теория графов), так и *эвристические* принципы. «*Эвристика* – это искусство нахождения истины» [2]. В области искусственного интеллекта понятие *эвристика* противопоставляется понятию *алгоритм*. В компьютерных играх постоянно совершенствуют их интеллектуальная составляющая.

Многие игры относятся к *играм с полной информацией*. Иначе говоря, это игры, в которых ничто (за исключением мыслей противника) не скрыто от игрока. В этих играх отсутствует элемент случайности. По мнению некоторых ведущих ученых [3] игры с полной информацией являются «тривиальными», но поиск эффективных игровых стратегий, тем не менее, отнюдь не тривиальная задача.

Так называемые «случайные игры» требуют принципиально иного подхода к их решению.

*Доказательство теорем.* Это направление перекрывается с определенными областями математики и решением проблем в ряде других областей (например, в робототехнике).

Что же касается моделирования логического мышления, то хорошей модельной задачей здесь может служить задача автоматизации доказательства теорем. Начиная с 1960 г., был разработан ряд программ, способных находить доказательства теорем в исчислении предикатов первого порядка. Эти программы обладают способностью делать дедуктивные заключения.

Например, существуют «интеллектуальные» программы, которые несколько минут доказывают десятки сложных теорем, часть их которых еще не была выведена математиками. Однако, до сих пор ни одна программа не вывела и не доказала ни одной теоремы, которая бы, что называется «позарез» была бы нужна математикам и была бы принципиально новой.

*Интеллектуальные роботы.* Их создание связано с объединением технологий искусственного интеллекта и методов кибернетики, робототехники. В настоящее время их производство ограничивается манипуляторами с жесткой схемой управления, а также роботами развлекательного и бытового назначения с узкой областью применения и ограниченными функциями. Сдерживающим фактором при разработке более совершенных кибернетических систем являются нерешенные проблемы в области машинного зрения, адаптивного поведения, накопления и обработки трехмерной визуальной информации. Эта область имеет непосредственную практическую ценность.

*Экспертные системы.*. В них воплощаются большие объемы знаний и навыков, присущих эксперту-человеку. Создано множество программ для ЭВМ, обеспечивающих экспертизу в различных областях знаний, которые и получили название *экспертные системы*. Эти системы пред-

ставляют большую ценность в медицинской диагностике, в проблемах управления производством, при обнаружении газовых и нефтяных месторождений, в налоговой политике и в решении других технологических проблем.

Следует отметить, что область искусственного интеллекта, пожалуй, как ни какая другая из областей научных исследований, пострадала от чрезмерного оптимизма на первых этапах. Считалось, например, что в обозримом будущем диапазон проблем, которыми будет заниматься машины, «сравняется с диапазоном проблем, подвластных человеческому уму».

В некоторых весьма серьезных областях действия машины разительно отличаются от того, что способен делать мозг человека. В связи с этим можно сформулировать некоторые нерешенные проблемы.

*Различие между машиной и мозгом*. Различие в действиях мозга и машины, пожалуй, наиболее ярко проявилось в методах, которые они используют при игре в шахматы. Безусловно, при игре в шахматы машины достигли сейчас очень высокого «мастерства». Однако успех игровых программ основывается совсем на иных факторах нежели игра человека. Машинные методы шахматной игры базируются на громадной вычислительной способности машины. Программы не способны порождать свои собственные эвристики.

*Параллельная обработка информации*. Это присуще мозгу. Человек способен охватить в целом структуру той или иной проблемной ситуации таким образом, что он может выделить существенные аспекты проблемы, удерживая менее существенное в «краевом сознании». Такой результат обеспечивается исключительно параллельной работой мозга, но ее не удается смоделировать в искусственных системах.

Благодаря параллельной работе мозга процессу мышления одновременно доступно громадное число единиц информации и при этом возможно совершать огромное количество действии по их комбинированию и преобразованию.

*Дискретная логика и непрерывность*. Мозг удивительно легко комбинирует методы, соответствующие непрерывной среде, с обработкой логической информации. В терминах дискретных понятий может быть выражен целый ряд важных принципов управления, но они имеют *элементарную иллюстрированность*, выраженную в непрерывных переменных. Легкость, с которой мозг объединяет дискретные и непрерывные подходы, вероятно, связан с эволюции интеллекта. В этом проявляется гибкость мозга, которую трудно имитировать на машинах.

*Обучение.* Одной из многих особенностей искусственной интеллектуальной системы является *обучение*, как самая существенная особенность. Дать определение понятию «обучение» столь же трудно, как и определить понятие «интеллект». То, что под этим обычно подразумева-

ется в контексте искусственного интеллекта или кибернетики, это не столько накопление данных (информации), сколько приобретение и накопление на опыте необходимых навыков.

*Самоорганизующиеся системы.* Многие исследователи понимали, что изменения, связанные с обучением системы, необходимо вводить более фундаментальным и универсальным способом. Первые попытки создать *самоорганизующиеся* сети на основе искусственных нейронных сетей пока не оправдались. Знания о пластичности реальных нейронов весьма неполны и неоднозначны. Тем не менее, они заключают в себе важные идеи.

*Обработка нечеткой информации.* Использование теории нечетких множеств, нечеткой логики и в целом основ нечеткой информации позволяет формализовать многие функции естественного интеллекта. И на этой основе развивать компьютерные технологии в реализации искусственного интеллекта.

*Морально-этические проблемы искусственного интеллекта.* Потребность в «чистом» искусственном интеллекте высокого уровня остается. Исследования, связанные с искусственным интеллектом, привлекательны по самой своей сущности, в частности, потому, что вопрос о том, как много можно добиться от вычислительной машины, интересен с философской, общемировоззренческой, точки зрения. Эти исследования, разумеется, сулят двоякую выгоду. *Во-первых*, как и в прошлом, они положат начало новым вычислительным методам, которые в дальнейшем найдут новые приложения. *Во-вторых*, они помогут в понимании сути исследуемых задач, связанных с естественным интеллектом. Что можно сформулировать одним вопросом: «Существуют ли такие аспекты интеллекта человека, которые *в принципе* нельзя смоделировать на вычислительной машине?»

По мнению Тьюринга, если человек при беседе с машиной не сможет понять, что ему отвечает не человек, а машина, то такая машина будет «мыслящей». Это и есть критерий Тьюринга или тест Тьюринга на разумность. Тьюринг пишет о машине, но очевидно, что подразумевает программу. Сейчас понятно, что никаких препятствий к созданию таких программ для компьютеров нет. Программа, в основном, просто ищет в памяти подходящие фразы среди тысяч или миллионов уже готовых и выдает ответы, которые не отличаются от человеческих. Но в этом процессе отсутствует мысль. Математическая формула или программа не мыслят. У набора формул и знаков нет ума, сознания и интеллекта.

Тьюринг не дал определения главным для понимания терминам «мыслить», мышление, сознание. Такой бихевиористический (направление экспериментальной психологии) подход, когда неважно, что происходит в человеческом мозге, фиксируется только внешнее поведение и

реакция на некоторый стимул. В результате этот слишком простой критерий (тест) приводит к ошибкам и преувеличениям.

Правильные или правдоподобные ответы – это необходимое, но недостаточное условие для признания того, что машина мыслит. Поэтому критерий Тьюринга представляется недостаточным.

Если применить критерий Тьюринга к животным, то получается, что обезьяны, дельфины, лошади, вороны и все другие животные не обладают мышлением. Однако биологи, изучающие поведение животных и их организмы (в том числе нервную систему и мозг) последние сто лет, с помощью различных тестов доказали, что многие животные обладают мышлением. Абсурдная картина. Программы, имитирующие ответы, согласно критерию Тьюринга мыслят, а животные – не мыслят.

Вернемся к программам. Запрограммированная машина не осуществляет процессы, аналогичные процессам, проходящим в мозге человека (животного). Поэтому, даже если вычислительная машина с программой хорошо имитирует человеческие ответы, признавая ее мыслящей, мы просто приписываем ей «мышление», что крайне ненаучно.

Поэтому правильно было бы использовать термин «имитация интеллекта» вместо «искусственного интеллекта», а точнее – имитация и моделирование отдельных функций интеллекта. Однако термин «искусственный интеллект» прижился, но содержание его и оттенки со временем меняются.

В заключение хотелось бы обратить внимание на одну полезную книгу [10] по основам искусственного интеллекта. Первое издание этой книги стало классическим образцом литературы по искусственному интеллекту. Оно было принято в качестве учебного пособия больше чем в 600 университетах 60 стран мира и получило высокую оценку как окончательный итог обобщения результатов, достигнутых в этой области науки.

## Глава 2

# НЕЧЕТКИЕ МНОЖЕСТВА И ИХ СВОЙСТВА

## 2.1. Понятие нечеткого множества

Следуя Лотфи Заде [8], введем понятие нечеткого множества следующим образом.

Пусть $X$ – некоторое *универсальное* множество элементов $x$, и $\mu_A : X \to [0;1]$. Универсальное множество $X$ не может быть нечетким.

***Определение***. *Нечетким подмножеством* $A$ в $X$ называется множество вида $\{(x, \mu_A(x)) : x \in X\}$; при этом $\mu_A(x)$ называется *степенью (или функцией) принадлежности* элемента $x$ множеству $X$, принимающая значения в некотором вполне упорядоченном множестве $M$ (например, $M = [0;1]$).

Таким образом, задание нечеткого подмножества $A$ в $X$ эквивалентно заданию его *функции принадлежности* $\mu_A(x)$ и принято обозначать $A \subset X$. Следуя сложившейся традиции, вместо корректного термина «нечеткое подмножество» будем употреблять термин «нечеткое множество». Если $M = \{0;1\}$, то нечеткое множество $A$ можно рассматривать как обычное или четкое подмножество множества $X$. Для непрерывных множеств соответствующие функции принадлежности непрерывны, для дискретных множеств – дискретны.

Приведенное определение нечеткого множества является довольно общим. Поэтому при анализе и синтезе нечетких систем используются различные его частные случаи.

С учетом того, что в нечетком множестве всегда перечисляются все элементы универсального множества, при известном универсальном множестве $X$ для определения нечеткого множества достаточно представить его функцию принадлежности.

**Пример 2.1**

Пусть на универсальном конечном множестве

$$X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$$

задано нечеткое множество

$$A = \{x_1|0,\ x_2|0.2,\ x_3|0.8,\ x_4|0.5,\ x_5|0.1,\ x_6|0\},$$

где $\mu_A(x) = \{0,\ 0.2,\ 0.8,\ 0.5,\ 0.1,\ 0\}$ – функция принадлежности множества $A$. Таким образом, для задания нечеткого множества при известном универсальном множестве достаточно задать его функцию принадлежности.

**Пример 2.2**

Пусть $X$ – множество натуральных чисел:

$$X = \{0,\ 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ ...\}.$$

Тогда нечеткое множество $A$ «небольших» натуральных чисел может быть задано своей функцией принадлежности:

$$\mu_A(x) = \{1,\ 0.8,\ 0.6,\ 0.4,\ 0.2,\ 0,\ 0,\ 0,\ ...\}.$$

**Пример 2.3**

Пусть $X$ является подмножеством числовой прямой. В этом случае часто используются так называемые нечеткие множества $(L-R)$-типа [8]. Функции принадлежности для таких множеств задаются с помощью функций $L$ и $R$, удовлетворяющих следующим условиям:

1) $L(0) = R(0) = 1$;

2) $L$ и $R$ – невозрастающие функции на множестве неотрицательных действительных чисел. Например,

$$L(x) = \exp(-|x|^p),\ \ p \geq 0; \qquad R(x) = (1+|x|^p)^{-1},\ \ p \geq 0.$$

Тогда функция принадлежности нечеткого множества $A$, имеющая $(L-R)$– тип, задается так:

$$\mu_A(x) = \begin{cases} L\left((a'-x)/a_L\right) & \text{при } x \leq a',\ 0 < a_L < a'; \\ R\left((x-a'')/a_R\right) & \text{при } x \geq a'',\ 0 < a_R < a''; \\ 1 & \text{при } x \in [a',\ a'']. \end{cases}$$

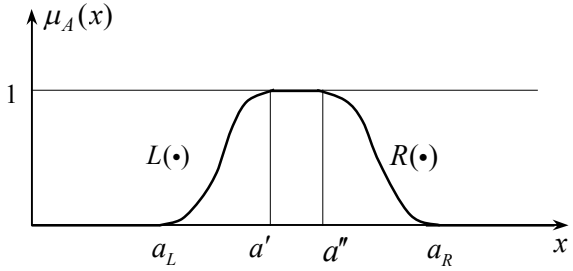



Рис. 2.1

Иногда отрезок $[a', a'']$ называют интервалом толерантности, а $a_L$ и $a_R$ – левым и правым коэффициентами нечеткости соответственно.

## 2.2. Отличие нечетких и вероятностных систем

Рассмотрим два аспекта концепции неопределенности: *нечеткость* (*лингвистическую неопределенность*) и *вероятность* (*стохастическую неопределенность*).

*Лингвистическая неопределенность*. Этот тип неопределенности связан с неточностью обычного человеческого языка и мышления, с которой приходится сталкиваться в повседневной жизни. Очевидно, что понятию «высокие люди» невозможно дать точное количественное определение: 1) высокие и низкие люди имеют свои собственные представления о том, каких людей считать высокими; 2) если формально считать высокими людей выше 180 см, то будет ли человек ростом 179,999 см высоким? 3)как тогда быть с колебаниями роста человека в течение дня? 4) следует также учесть, что количественная оценка понятия «высокий человек» у разных наций различна.

Для изучения подобных субъективных оценок создана наука *психолингвистика*, где принято считать, что люди используют слова в качестве субъективных категорий. По ним классифицируются объекты с характеристическими свойствами «высота», «вес», «температура», «цвет» и т.д. и, хотя они точно не определены, люди успешно используют их для оценок и решений с учетом многих факторов.

Чтобы адекватно использовать логику, присутствующую в человеческих суждениях, для решения технических проблем, необходимо разработать соответствующую математическую модель. Именно с этой целью была разработана *нечеткая логика*, которая позволяет представить процессы принятия решений и оценки ситуаций человеком в алгоритми-

ческой форме. Хотя возможности человеческого мышления и фантазии безграничны, пределы того, что позволяет моделировать нечеткая логика, существуют.

Основное достоинство теории нечетких множеств заключается в возможности использовать лингвистические переменные вместо количественных, нечеткую логику вместо бинарной логики для формального представления неточных *субъективных* категорий. Здесь оказывается возможным учесть опыт и интуицию человека.

*Стохастическая неопределенность*. Стохастическая неопределенность имеет место в ситуациях, когда событие может произойти, а может и не произойти. При этом с течением времени эта неопределенность может измениться. Дополнительно событие должно отвечать определенным условиям статистической устойчивости – сохранение частоты события в течение достаточно длительного времени при повторных испытаниях.

В высказывании «Вероятность того, что при бросании монеты выпадет «орел», равна 0,5» предполагается, что: 1) монета и поверхность падения имеют идеальную форму; 2) процесс бросания монеты идеален с точки зрения субъектов эксперимента; 3) вертикальное положение монеты после бросания исключается. Причем неопределенность пропадает сразу после того, как монета упадет, поскольку она оказывается в одном из двух положений: «орлом» кверху или «орлом» книзу.

Таким образом, рассматриваемое высказывание имеет смысл только по отношению к событию в будущем. Изменение условий эксперимента может привести к изменению высказывания. Обеспечить идеальные условия на практике невозможно. Предельные теоремы теории вероятностей как раз и предназначены для оценки идеальности условий в длинной серии испытаний.

Исторически теория вероятностей была первой математической дисциплиной, моделирующей неопределенность. По этой причине любая неопределенность могла быть выдана за стохастическую с приписыванием ей свойства статистической устойчивости.

Что же касается вероятностного процесса, то предсказать последовательность событий просто невозможно. Могут быть точно описаны только статистические оценки некоторых усредненных характеристик такого процесса.

*Нечеткая логика и теория вероятностей.* Рассмотренные выше примеры высказываний иллюстрируют тот факт, что стохастическая и лингвистическая неопределенности имеют различный характер.

Стохастическая неопределенность имеет дело с неопределенностью того, произойдет ли некоторое хорошо описанное событие в будущем. Теория вероятностей позволяет дать ответ на этот вопрос.

Напротив, лингвистическая неопределенность связана с неточностью описания самой ситуации или события независимо от времени их рассмотрения. Теория вероятностей не может использоваться для решения подобных проблем, поскольку представления о субъективных категориях, присутствующих в процессе мышления человека, не согласуются с ее аксиомами.

Приведем один наглядный пример, иллюстрирующий семантическое различие между стохастической и лингвистической неопределенностями. Представим себе ситуацию, когда для утоления жажды необходимо выбрать одну из двух бутылок с прозрачной жидкостью. Содержимое бутылок неизвестно, одна из них помечена буквой $A$, а вторая буквой $B$.

Предположим, что известна *степень принадлежности* содержимого бутылки $A$ к жидкостям, пригодным для питья: $\mu(A) = 0.9$. Также известна *вероятность* того, что содержимое бутылки $B$ пригодно для питья: $p(B) = 0.9$. Какую бутылку предпочесть?

Можно провести следующие рассуждения. Анализируя информацию о содержимом бутылки $A$, можем предположить, что в ней находится не совсем (на 0.9) пригодная для питья жидкость. Например, болотная вода. Чистая вода имела бы степень принадлежности 1. В то же время в бутылке $A$ никак не может находиться ядовитая жидкость, скажем, серная кислота, поскольку в этом случае степень принадлежности была бы равна 0. Анализируя информацию о содержимом бутылки $B$, можно предположить (после многократно выбранной бутылки $B$), что в 9 случаях из 10 жидкость в этой бутылке питьевая (близкая к чистой воде). Но в единственном случае из 10 в бутылке может оказаться ядовитая жидкость. Поэтому в нашем случае напрашивается очевидный вывод (чтобы избежать тяжелых последствий) – выбрать бутылку $A$.

Но этот выбор может измениться, если изменится информация о содержимом бутылок. Так, совсем не очевидно, какую из бутылок предпочесть, если степень принадлежности для бутылки $A$ и вероятность для бутылки $B$ станут равными, скажем, 0.5.

Таким образом, понятие нечеткого множества способно обеспечить нас адекватной информацией относительно неточного описания тех или иных ситуаций. По существу, этот подход наиболее применим для решения таких проблем, в которых неопределенность характеризуется отсутствием хорошо определенных критериев, позволяющих однозначно судить о принадлежности элементов тому или иному классу. *Именно в этом проявляется различие между нечеткостью и случайностью*.

В то же время нечеткие модели не являются заменой моделей, разработанных в теории вероятностей.

Четкие множества, по сути, являются частным случаем нечетких множеств, поэтому нечеткие модели обобщают традиционные более изученные четкие математические модели. Иногда они работают лучше, а иногда нет. Эффективность той или иной модели проявляется в ее способности адекватно решить конкретную проблему.

## 2.3. Операции над нечеткими множествами

Обозначим через $P(X)$ множество всех нечетких множеств $X$. Опишем некоторые свойства $P(X)$.

Прежде всего заметим, что мощность $P(X)$ выше мощности множества подмножеств $X$, Действительно, если мощности множества $X$ и множества значений функции принадлежности $M$ конечны, то есть $|X| = n$, $|M| = m$, то $|P(X)| = m^n$. При $m = 2$ (обычные множества) $|P(X)| = 2^n$ – число подмножеств $X$.

Основные теоретико-множественные операции в $P(X)$ вводятся следующим образом.

Пусть $X$ – универсальное множество, $A$ и $B$ – нечеткие множества на множестве $X$ с функциями принадлежности $\mu_A(x)$ и $\mu_B(x)$ соответственно.

***Определение.*** Нечеткие множества $A$ и $B$ *равны* тогда и только, когда

$$\forall x \in X: \ \mu_A(x) = \mu_B(x),$$

и обозначается $A = B$

***Определение***. Нечеткое множество $A$ *включает* (содержит) нечеткое множество $B$, если

$$\forall x \in X: \ \mu_A(x) \geq \mu_B(x),$$

и обозначается $A \supseteq B$.

***Определение.*** Нечеткое множество $A$ является *дополнением* нечеткого множества $B$, если

$$\forall x \in X: \ \mu_A(x) = 1 - \mu_B(x),$$

и обозначается $\overline{A} = B$.

***Определение.*** *Объединением* нечетких множеств $A$ и $B$ в $X$ называется нечеткое множество $C = A \cup B$ с функцией принадлежности вида

$$\forall x \in X: \ \mu_C(x) = \mu_{A \cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}.$$

***Определение.*** *Пересечением* нечетких множеств $A$ и $B$ в $X$ называется нечеткое множество $C = A \cap B$ с функцией принадлежности вида

$$\forall x \in X:\ \mu_C(x) = \mu_{A\cap B}(x) = \min\{\mu_A(x),\ \mu_B(x)\}.$$

***Определение.*** *Дизъюнктивная сумма* нечетких множеств $A$ и $B$ в $X$ определяется в терминах объединений и пересечений следующим образом:

$$A \oplus B = \left(A \cap \overline{B}\right) \cup \left(\overline{A} \cap B\right).$$

***Определение.*** *Разность* нечетких множеств $A$ и $B$ в $X$ определяется следующим образом

$$\mu_{A\setminus B}(x) = \begin{cases} \mu_A(x) - \mu_B(x), & \text{если } \mu_A(x) \geq \mu_B(x); \\ 0, & \text{если } \mu_A(x) < \mu_B(x). \end{cases}$$

***Примечание.*** Основным критерием корректности введения операций над нечеткими множествами считается выполнение проверки этих операций над обычными (четкими) подмножествами. Поэтому некоторые операции в теории нечетких множеств определены неоднозначно [8]. Например, для разности нечетких множеств $A$ и $B$ наряду с $A \setminus B$, используется выражение $A - B = A \cap \overline{B}$. При этом для обычных подмножеств $A \setminus B = A \cap \overline{B}$, а для нечетких множеств $A \setminus B \neq A \cap \overline{B}$ (это легко проверить).

**Пример 2.4**

Рассмотрим примеры, которые иллюстрируют введенные операции.

Пусть $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ – универсальное множество; нечеткие множества $A$ и $B$ заданы своими функциями принадлежности:

$\mu_A(x) = \{0, 0.3, 0.7, 0.8, 0.2, 0\}$ и $\mu_B(x) = \{0.9, 0.6, 0.4, 0.2, 0.1, 0\}$.

Тогда

$$\mu_{\overline{A}}(x) = \{1, 0.7, 0.3, 0.2, 0.8, 1\} \quad \text{для } \overline{A};$$

$$\mu_{\overline{B}}(x) = \{0.1, 0.4, 0.6, 0.8, 0.9, 1\} \quad \text{для } \overline{B};$$

$$\mu_{A\cup B}(x) = \{0.9, 0.6, 0.7, 0.8, 0.2, 0\} \quad \text{для } A \cup B;$$

$$\mu_{A\cap B}(x) = \{0, 0.3, 0.4, 0.2, 0.1, 0\} \quad \text{для } A \cap B;$$

$$\mu_{A\cap\overline{B}}(x) = \{0, 0.3, 0.6, 0.8, 0.2, 0\} \quad \text{для } A \cap \overline{B};$$

$$\mu_{\overline{A}\cap B}(x) = \{0.9, 0.7, 0.3, 0.2, 0.1, 0\} \quad \text{для } \overline{A} \cap B;$$

$$\mu_{A\oplus B}(x) = \{0.9,\ 0.7,\ 0.6,\ 0.8,\ 0.2,\ 0\} \quad \text{для} \quad A \oplus B;$$
$$\mu_{A\setminus B}(x) = \{0,\ 0,\ 0.3,\ 0.6,\ 0.1,\ 0\} \quad \text{для} \quad A \setminus B.$$

Для алгебры нечетких множеств $A$, $B$ и $C$ в $X$ с определенными операциями дополнения, объединения и пересечения справедливы следующие свойства:

– коммутативность $A \cup B = B \cup A$, $A \cap B = B \cap A$;

– ассоциативность $(A \cup B) \cup C = A \cup (B \cup C)$,

$$(A \cap B) \cap C = A \cap (B \cap C);$$

– идемпотентность $A \cap A = A$, $A \cup A = A$;

– инволюция $\left(\overline{\overline{A}}\right) = A$.

Перечисленные свойства очевидны. При этом в отличие от обычных подмножеств *не выполняются следующие свойства*

$$A \cap \overline{A} \neq \varnothing, \quad A \cup \overline{A} \neq X.$$

Этот факт отражает особенность нечетких множеств – размытость границ. В прикладных задачах эта особенность иногда затрудняет определяться с выводами.

Свойства *дистрибутивности* и *двойственности* (теоремы де Моргана) требуют доказательств

**Утверждение 2.1**

Для нечетких множеств $A$, $B$ и $C$ выполняется свойство дистрибутивности

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C), \tag{2.1}$$
$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C). \tag{2.2}$$

*Доказательство.* Докажем первое равенство. В соответствии с определениями равенства, пересечения и объединения нечетких множеств, необходимо доказать, что для любого $x \in X$ выполнено

$$\min\left\{\mu_A(x),\ \max\{\mu_B(x),\ \mu_C(x)\}\right\} =$$
$$= \max\left\{\min\{\mu_A(x),\ \mu_B(x)\},\ \min\{\mu_A(x),\ \mu_C(x)\}\right\}. \tag{2.3}$$

Зафиксируем некоторую точку $x \in X$.

Пусть $a = \mu_A(x)$, $b = \mu_B(x)$, $c = \mu_C(x)$. Возможны 15 вариантов соотношения $a$, $b$ и $c$. Все варианты необходимо проверить.

1) $a > b > c$,

$$\min\left\{a,\ \max\{b,\ c\}\right\} = b; \quad \max\left\{\min\{a,\ b\},\ \min\{a,\ c\}\right\} = b.$$

Таким образом, равенство (2. 3) справедливо.

2) $a > b = c$,

$\min\{a, \max\{b, c\}\} = b = c$; $\max\{\min\{a, b\}, \min\{a, c\}\} = b = c$.

Таким образом, равенство (2. 3) справедливо.

3) $a = b > c$

$\min\{a, \max\{b, c\}\} = b = a$; $\max\{\min\{a, b\}, \min\{a, c\}\} = b = a$.

Таким образом, равенство (2.3) справедливо.

4) $a = b = c$,

$$\min\{a, \max\{b, c\}\} = a = b = c;$$

$$\max\{\min\{a, b\}, \min\{a, c\}\} = a = b = c.$$

Таким образом, равенство (2. 3) справедливо.

5) $a > c > b$,

$\min\{a, \max\{b, c\}\} = c$; $\max\{\min\{a, b\}, \min\{a, c\}\} = c$.

Таким образом, равенство (2. 3) справедливо.

6) $a > c = b$,

$\min\{a, \max\{b, c\}\} = b = c$; $\max\{\min\{a, b\}, \min\{a, c\}\} = b = c$.

Таким образом, равенство (2. 3) справедливо.

7) $a = c > b$,

$\min\{a, \max\{b, c\}\} = a = c$; $\max\{\min\{a, b\}, \min\{a, c\}\} = a = c$.

Таким образом, равенство (2. 3) справедливо.

8) $b > a > c$,

$\min\{a, \max\{b, c\}\} = a$; $\max\{\min\{a, b\}, \min\{a, c\}\} = a$.

Таким образом, равенство (2. 3) справедливо.

9) $b > a = c$,

$\min\{a, \max\{b, c\}\} = a$; $\max\{\min\{a, b\}, \min\{a, c\}\} = a$.

Таким образом, равенство (2. 3) справедливо.

10) $b > c > a$,

$\min\{a, \max\{b, c\}\} = a$; $\max\{\min\{a, b\}, \min\{a, c\}\} = a$.

Таким образом, равенство (2. 3) справедливо.

11) $b > c = a$,

$\min\{a, \max\{b, c\}\} = a$; $\max\{\min\{a, b\}, \min\{a, c\}\} = a$.

Таким образом, равенство (2. 3) справедливо.

12) $b = c > a$,

$\min\{a, \max\{b, c\}\} = a$; $\max\{\min\{a, b\}, \min\{a, c\}\} = a$.

Таким образом, равенство (2. 3) справедливо.

13) $c > a > b$,

$$\min\{a, \max\{b, c\}\} = a; \quad \max\{\min\{a, b\}, \min\{a, c\}\} = a.$$

Таким образом, равенство (2. 3) справедливо.

14) $c > a = b$,

$$\min\{a, \max\{b, c\}\} = a = b; \quad \max\{\min\{a, b\}, \min\{a, c\}\} = a = b.$$

Таким образом, равенство (2. 3) справедливо.

15) $c > b > a$,

$$\min\{a, \max\{b, c\}\} = a; \quad \max\{\min\{a, b\}, \min\{a, c\}\} = a.$$

Таким образом, равенство (2. 3) справедливо.

Второе равенство (2.2) доказывается аналогично.

Утверждение доказано.

**Утверждение 2.2**

Для нечетких множеств $A$, $B$ выполняется свойство двойственности (теорема де Моргана)

$$\overline{A \cap B} = \overline{A} \cup \overline{B}, \tag{2.4}$$

$$\overline{A \cup B} = \overline{A} \cap \overline{B}. \tag{2.5}$$

***Доказательство.*** Докажем первое равенство. В соответствии с определениями равенства, пересечения и объединения нечетких множеств, необходимо доказать, что для любого $x \in X$ выполнено

$$1 - \min\{\mu_A(x), \mu_B(x)\} = \max\{1 - \mu_A(x), 1 - \mu_B(x)\}. \tag{2.6}$$

Пусть $a = \mu_A(x)$, $b = \mu_B(x)$. Возможны 3 варианта соотношений $a$ и $b$. Все варианты необходимо проверить.

1) $a > b$, $1 - \min\{a, b\} = 1 - b$, $\max\{1 - a, 1 - b\} = 1 - b$. Таким образом, равенство (2.6) выполнено.

2) $a = b$, $1 - \min\{a, b\} = 1 - a = 1 - b$, $\max\{1 - a, 1 - b\} = 1 - a = 1 - b$. Таким образом, равенство (2.6) выполнено.

3) $b > a$, $1 - \min\{a, b\} = 1 - a$, $\max\{1 - a, 1 - b\} = 1 - a$. Таким образом, равенство (2.6) выполнено.

Второе равенство (2.5) доказывается аналогично.

Утверждение доказано.

*Для нечетких множеств (в отличие от обычных подмножеств) вводятся операции алгебраической суммы и алгебраического произведения.*

***Определение.*** *Алгебраическая сумма* $A \hat{+} B$ двух нечетких множеств $A$ и $B$ в $X$ определяется следующим образом

$$\forall x \in X: \ \mu_{A \hat{+} B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x).$$

***Определение.*** *Алгебраическое произведение* $A \bullet B$ двух нечетких множеств $A$ и $B$ в $X$ определяется следующим образом

$$\forall x \in X:\ \mu_{A \bullet B}(x) = \mu_A(x) \cdot \mu_B(x).$$

Свойства коммутативности, инволюции и ассоциативности для алгебраического произведения является прямым следствием определения и соответствующих свойств операций умножения и сложения для чисел.

В некоторой дополнительной проверке могут нуждаться свойства ассоциативности для алгебраической суммы и теоремы де Моргана для указанных операций. Докажем первое из них.

**Утверждение 2.3**

Для нечетких множеств $A$, $B$ и $C$ в $X$ выполняется

$$(A \hat{+} B) \hat{+} C = A \hat{+} (B \hat{+} C).$$

***Доказательство.*** В соответствии с определениями равенства и алгебраической суммы нечетких множеств, необходимо доказать, что для любого $\forall x \in X$ выполнено

$$\begin{aligned}
&\left(\mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x)\right) + \mu_C(x) - \\
&\qquad -\left(\mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x)\right) \cdot \mu_C(x) = \\
&= \mu_A(x) + \left(\mu_B(x) + \mu_C(x) - \mu_B(x) \cdot \mu_C(x)\right) - \\
&\qquad -\left(\mu_A(x) \cdot \mu_B(x) + \mu_C(x) - \mu_B(x) \cdot \mu_C(x)\right).
\end{aligned}$$

(2.7)

Зафиксируем некоторую точку $x \in X$.

Пусть $a = \mu_A(x)$, $b = \mu_B(x)$, $c = \mu_C(x)$.

Тогда левая часть (2.7) равна

$$(a + b - a \cdot b) + c - (a + b - a \cdot b) \cdot c = a + b + c - a \cdot b - a \cdot c - b \cdot c + a \cdot b \cdot c$$

.

Левая часть (2.7) равна

$$a + (b + c - b \cdot c) - a \cdot (b + c - b \cdot c) = a + b + c - b \cdot c - a \cdot b - a \cdot c + a \cdot b \cdot c$$

.

Сравнивая правые части последних двух равенств, получаем выполнение (2.7).

Утверждение доказано.

Доказательство теорем де Моргана для рассматриваемой алгебры проводится по схеме доказательства утверждения 2.3.

Новое в этой алгебре – невыполнение свойств идемпотентности и дистрибутивности. Первое является прямым следствием определения и свойств арифметических операций для чисел. Докажем невыполнение

дистрибутивности алгебраической суммы относительно алгебраического произведения.

**Утверждение 2.4**

Для нечетких множеств $A$, $B$ и $C$ в $X$ выполняется

$$A\bullet(B \hat{+} C) \neq (A\bullet B) \hat{+} (A\bullet C). \qquad (2.8)$$

*Доказательство.* Выберем некоторую точку $x \in X$, и, полагая $a = \mu_A(x)$, $b = \mu_B(x)$, $c = \mu_C(x)$, получим для, соответственно, левой части $a(b + c - bc) = ab + ac - abc$ и правой части (2.8) $ab + ac - (ab)(ac)$.

Таким образом, правая и левая части (2.8) равны, если только $a = a^2$.

Утверждение доказано.

Недистрибутивность алгебраического произведения относительно алгебраической суммы доказывается аналогично.

Справедливы также следующие свойства

$$A\bullet(B \cap C) = (A\bullet B) \cap (A\bullet C);$$
$$A\bullet(B \cup C) = (A\bullet B) \cup (A\bullet C);$$
$$A \hat{+} (B \cap C) = (A \hat{+} B) \cap (A \hat{+} C);$$
$$A \hat{+} (B \cup C) = (A \hat{+} B) \cup (A \hat{+} C).$$

Доказательство этих свойств выполняется по схеме доказательства, например, дистрибутивности (2.1).

## 2.4. Расстояние между нечеткими множествами

Одной из характеристик множества $P(X)$ нечетких множеств является *метрика*, или расстояние между нечеткими множествами (расстояние между функциями принадлежности).

Число $d(x, y)$ называется *расстоянием* на некотором множестве $V$, если при $x, y, z \in V$ выполняются аксиомы Фреше:

1) $d(x, x) = 0$,
2) $d(x, y) \geq 0$ – неотрицательность,
3) $d(x, y) = d(y, x)$ – симметричность,
4) $d(x, z) \leq d(x, y) + d(y, z)$ – транзитивность.

Приведем примеры известных расстояний между нечеткими множествами.

*Расстояние Хемминга.* Пусть $A, B \subset X$ нечеткие множества, заданные своими функциями принадлежности $\mu_A(x_i)$ и $\mu_B(x_i)$:

– *расстояние Хемминга для конечного* универсального множества $X: |X| = n$:

$$d(A,B) = \sum_{i=1}^{n} \left| \mu_A(x_i) - \mu_B(x_i) \right|;$$

– *относительное расстояние Хемминга для конечного* универсального множества $X: |X| = n$:

$$\delta(A,B) = \frac{1}{n} \sum_{i=1}^{n} \left| \mu_A(x_i) - \mu_B(x_i) \right|, \quad 0 \le \delta(A,B) \le 1;$$

– *расстояние Хемминга для бесконечного* универсального множества $X \subseteq R^1$:

$$d(A,B) = \int_X \left| \mu_A(x_i) - \mu_B(x_i) \right| dx \text{, если интеграл сходящийся;}$$

– *относительное расстояние Хемминга для бесконечного* универсального множества $X \subseteq R^1$:

$$\delta(A,B) = \frac{1}{|X|} \int_X \left| \mu_A(x_i) - \mu_B(x_i) \right| dx, \text{ если интеграл сходящийся;}$$

*Расстояние Евклида.* Пусть $A, B \subset X$ нечеткие множества, заданные своими функциями принадлежности $\mu_A(x_i)$ и $\mu_B(x_i)$:

– *расстояние Евклида для конечного* универсального множества $X$: $|X| = n$:

$$e(A,B) = \sqrt{\sum_{i=1}^{n} \left( \mu_A(x_i) - \mu_B(x_i) \right)^2};$$

– *относительное расстояние Евклида для конечного* универсального множества $X: |X| = n$:

$$\varepsilon(A,B) = \frac{1}{\sqrt{n}} \sqrt{\sum_{i=1}^{n} \left( \mu_A(x_i) - \mu_B(x_i) \right)^2}, \quad 0 \le \varepsilon(A,B) \le 1;$$

– *расстояние Евклида для бесконечного* универсального множества $X \subseteq R^1$:

$$e(A,B) = \sqrt{\int_X \left( \mu_A(x_i) - \mu_B(x_i) \right)^2 dx} \text{, если интеграл сходящийся;}$$

– *относительное расстояние Евклида для бесконечного* универсального множества $X \subseteq R^1$:

$$e(A,B) = \frac{1}{\sqrt{|X|}} \sqrt{\int_X \left(\mu_A(x_i) - \mu_B(x_i)\right)^2 dx}\,.$$

Легко проверить, что приведенные примеры расстояния удовлетворяют аксиомам Фреше.

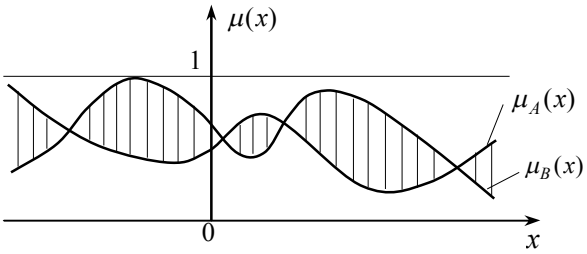



Рис. 2.1.

Очевидно, что можно придумать и определить и другие расстояния. Выбор того или иного расстояния зависит от природы рассматриваемой проблемы. Каждое из них обладает своими преимуществами и недостатками, которые становятся очевидными в приложениях.

**Пример 2.5**

Пусть $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ – универсальное множество; нечеткие множества $A$ и $B$ заданы своими функциями принадлежности:

$$\mu_A(x) = \{0,\ 0.3,\ 0.7,\ 0.8,\ 0.2,\ 0\}$$
$$\mu_B(x) = \{0.9,\ 0.6,\ 0.4,\ 0.2,\ 0.1,\ 0\}\,.$$

Имеем

$$d(A,B) = \sum_{i=1}^{6} \left|\mu_A(x_i) - \mu_B(x_i)\right| = |0-0.9| + |0.3-0.6| + |0.7-0.4| +$$
$$+ |0.8-0.2| + |0.2-0.1| + |0-0| = 2.2\,;$$

$$\delta(A,B) = \frac{1}{6}\sum_{i=1}^{n} \left|\mu_A(x_i) - \mu_B(x_i)\right| = 2.2/6 \cong 0.37$$

$$e(A,B) = \sqrt{\sum_{i=1}^{6} \left(\mu_A(x_i) - \mu_B(x_i)\right)^2} =$$

$$= \sqrt{(0-0.9)^2 + (0.3-0.6)^2 + (0.7-0.4)^2 + (0.8-0.2)^2 + (0.2-0.1)^2 + (0-0)^2} \cong 1.17$$

.

$$\varepsilon(A,B) = \frac{1}{\sqrt{6}} \sqrt{\sum_{i=1}^{6} \left(\mu_A(x_i) - \mu_B(x_i)\right)^2} \cong 0.48 .$$

***Примечание.*** Как правило, $\delta(A,B)$ и $\varepsilon(A,B)$ используют в случае бесконечных множеств при условии, что $X$ ограничено сверху и снизу; тогда расстояния $d(A,B)$ и $e(A,B)$ всегда конечны.

В этом случае можно определить $\delta(A,B)$ и $\varepsilon(A,B)$ (рис. 2. ):

$$\delta(A,B) = \frac{d(A,B)}{\beta - \alpha} \quad \text{и} \quad \varepsilon(A,B) = \frac{e(A,B)}{\sqrt{\beta - \alpha}} .$$

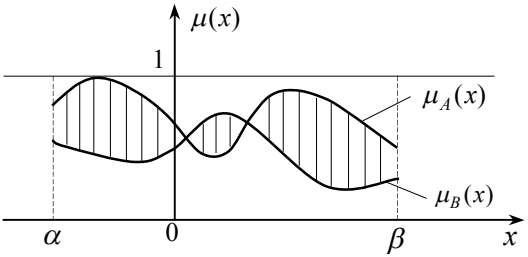



Рис. 2.2

## 2.5. Измерение степени нечеткости множества

Нечеткие множества могут иметь разную степень нечеткости. Меры нечеткости важны в приложениях теории нечетких множеств. Этот показатель является параметром оценки качества различных процедур и алгоритмов в распознавании образов, принятии решений, различных моделях преобразования информации [6,10,14,15] и т.п.

Работы по измерению степени нечеткости начались с 1972 г.. Исторически первыми были разработаны методы оценки нечеткости через *энтропию*. К настоящему времени можно выделить два основных подхода к оценке степени нечеткости множества: *метрический* и *аксиоматический*.

Оценка степени нечеткости множества через энтропию базируется на использовании понятии *энтропии в физике*. Энтропия системы измеряет степень беспорядка компонентов системы относительно вероятностей состояния. Поэтому использование понятия энтропии для вычисле-

ния степени нечеткости (неопределенности) множества являлось естественным в силу очевидной аналогии.

Степень неопределенности компонент физической системы относительно вероятности ее состояния составляет содержание понятия энтропии. Напомним определение энтропии.

Пусть $E_1$, $E_2$,…, $E_N$ – состояния системы и $p_1$, $p_2$,…, $p_N$ – вероятности состояний. Тогда *энтропия системы* определяется следующим выражением:

$$H(p_1, p_2, ..., p_N) = -\sum_{i=1}^{N} p_i \ln p_i . \qquad (2.9)$$

Непосредственно из данного определения вытекают следующие свойства энтропии:

*a*) $H = 0$ (минимально), если $\exists i \ \ (1 \le i \le N)$, $p_i = 1$;

*b*) $H = \ln N$ (максимально), если $\forall i \ \ (1 \le i \le N)$, $p_i = 1/N$.

Если в формуле (2.9) перед знаком суммы поставить нормировочный коэффициент $1/\ln p_N$, то значение энтропии будет меняться в пределах от 0 до 1:

$$H(p_1, p_2, ..., p_N) = -\frac{1}{\ln N}\sum_{i=1}^{N} p_i \ln p_i .$$

Оценка степени нечеткости множества $A$ через энтропию определяется по следующей формуле:

$$\xi\left(\nu_A(x_1), \nu_A(x_2), ..., \nu_A(x_N)\right) =$$

$$= -\frac{1}{\ln N \sum_{i=1}^{N} \mu_A(x_i)} \left[ \sum_{i=1}^{N} \mu_A(x_i) \cdot \left( \ln \sum_{i=1}^{N} \mu_A(x_i) \right) - \sum_{i=1}^{N} \mu_A(x_i) \cdot \ln \mu_A(x_i) \right]. \qquad (2.10)$$

**Пример 2.6**

Пусть $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ – универсальное множество; нечеткое множество $A$ задано функцией принадлежности:

$$\mu_A(x) = \{0.7, 0.9, 0, 0.6, 0.5, 1\} .$$

Положив $$\nu_A(x_i) = \frac{\mu_A(x_i)}{\sum_{i=1}^{6} \mu_A(x_i)},$$

получим $$\nu_A(x) = \left\{ \frac{7}{37}, \frac{9}{37}, 0, \frac{6}{37}, \frac{5}{37}, \frac{10}{37} \right\}.$$

Тогда

$$\xi\left(\nu_A(x_1),\nu_A(x_2),...,\nu_A(x_N)\right) = -\frac{1}{\ln 6}\sum_{i=1}^{6}\nu_A(x_i)\cdot\ln\nu_A(x_i) =$$

$$= -\frac{1}{\ln 6}\left(\frac{7}{37}\ln\frac{7}{37}+\frac{9}{37}\ln\frac{9}{37}+\frac{6}{37}\ln\frac{6}{37}+\frac{5}{37}\ln\frac{5}{37}+\frac{10}{37}\ln\frac{10}{37}\right) = 0.89 .$$

***Замечание***

Анализ формулы (2.10) позволяет заметить, что значение степени нечеткости зависит не от собственно значений функции принадлежности, а от их относительных значений. Это приводит к следующим «парадоксам»:

– степень нечеткости обычных множеств $\mu_A(x)=0$, $\forall x\in X$ и $\mu_A(x)=1$, $\forall x\in X$ максимальна;

– минимальна только степень нечеткости множеств с единственным ненулевым элементом – как «четких», так и «нечетких»

Из сказанного вытекает, что энтропийная мера нечеткости обладает недостатками. Поэтому важными являются другие подходы к оценке степени нечеткости.

*Метрический подход.*

Основная идея этого подхода к измерению степени нечеткости множеств базируется на использовании понятия расстояния между нечеткими множествами. Идея метрического подхода заключается в оценке степени нечеткости как расстояния между оцениваемым множеством и некоторым множеством с известной степенью нечеткости.

В качестве *базисного множества* с известной степенью нечеткости, расположенного на *наименьшем евклидовом расстоянии* от данного нечеткого множества $A$. Это множество называется *ближайшим* к нечеткому множеству $A$. Легко доказать, что это будет обычное множество (обозначим его $\underline{A}$), такое, что

$$\mu_{\underline{A}}(x) = \begin{cases} 0, & \text{если } \mu_A(x) < 0.5; \\ 1, & \text{если } \mu_A(x) > 0.5; \\ 0 \text{ (или 1)}, & \text{если } \mu_A(x) = 0.5. \end{cases}$$

Множество $\underline{A}$ с известной степенью нечеткости, равной нулю, принимается *базисным* множеством. Таким образом, *чем больше расстояние от некоторого множества $A$ до его ближайшего четкого множества $\underline{A}$, тем больше степень его нечеткости.*

Обычные ближайшие множества обладают следующими свойствами, которые легко проверить:

$$\underline{A \cap B} = \underline{A} \cap \underline{B}, \qquad \underline{A \cup B} = \underline{A} \cup \underline{B}.$$

Как правило, в прикладных задачах используется два индекса нечеткости: *линейный индекс нечетко*сти, определяемый посредством относительного расстояния Хемминга, и *квадратичный индекс нечеткости*, определяемый посредством относительного евклидова расстояния. Обозначим их $\nu(A)$ и $\eta(A)$ соответственно:

$$\nu(A) = \frac{2}{n} \cdot d(A, \underline{A}), \quad 0 \le \nu(A) \le 1;$$

$$\eta(A) = \frac{2}{\sqrt{n}} \cdot e(A, \underline{A}), \quad 0 \le \eta(A) \le 1.$$

Понятие подмножества, ближайшего к данному нечеткому множеству и понятие индекса нечеткости можно распространить на случай бесконечного универсального множества с оговоркой (например, относительно индекса нечеткости), что все рассматриваемые ряды (и интегралы) сходящиеся. Рассмотрим случай, когда множество $X = [\alpha, \beta] \subset R$. Тогда

$$\nu(A) = \frac{2}{\beta - \alpha} \cdot \int_{\alpha}^{\beta} \left| \mu_A(x_i) - \mu_{\underline{A}}(x_i) \right| dx;$$

$$\eta(A) = \frac{2}{\sqrt{\beta - \alpha}} \cdot \sqrt{\int_{\alpha}^{\beta} \left( \mu_A(x) - \mu_{\underline{A}}(x) \right)^2 dx} \quad .$$

Теперь можно дать общее определение степени нечеткости множества.

***Определение.*** Пусть $f$ – некоторая монотонная функция, $\rho(x, y)$ – метрика в $P(X)$, $A^*$ (например $\underline{A}$) – базисное множество, тогда степенью нечеткости $\xi(A)$ нечеткого множества $A$ называется значение $\xi(A) == f\left[ \rho(A, A^*) \right]$.

Функция $f$ подбирается для удовлетворения некоторым естественным требованиям для степени нечеткости, которые определяются прикладной задачей.

**Пример 2.7**

Пусть $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ – универсальное множество; нечеткое множество $A$ задано функцией принадлежности:

$$\mu_A(x) = \{0.7, 0.9, 0, 0.6, 0.5, 1\}.$$

Определим линейный индекс нечеткости множества $A$:

$$\nu(A) = \frac{2}{n} \cdot d(A, \underline{A})$$

Базисное множество $\underline{A}$:

$$\mu_{\underline{A}}(x) = \{1, 1, 0, 1, 0, 1\}.$$

Тогда

$$\nu(A) = \frac{2}{6} \cdot d(A, \underline{A}) = \frac{2}{6} \cdot \sum_{i=1}^{6} \left| \mu_A(x_i) - \mu_{\underline{A}}(x_i) \right| = 0.43.$$

## 2.6. Декомпозиция нечетких множеств

Между нечетким множеством универсального множества $X$ можно установить связь с определенным образом устроенным семейством обычных подмножества множества $X$. Эта связь вводится при помощи понятия подмножества $\alpha$ – уровня нечеткого множества.

Пусть $\alpha \in [0, 1]$. *Подмножеством $\alpha$ – уровня* нечеткого множества $A$ называется множество

$$A_\alpha = \{x \in X : \mu_A(x) \geq \alpha\}.$$

**Утверждение 2.4** (О декомпозиции)

Любое нечеткое множество $A$ можно следующим образом разложить на произведения обычных подмножеств по коэффициентам $\alpha_i$:

$$A = \max_{\alpha_i} \left\{ \alpha_1 \cdot A_{\alpha_1},\ \alpha_2 \cdot A_{\alpha_2}, ...,\ \alpha_n \cdot A_{\alpha_n} \right\},\ 0 < \alpha_i \leq 1,\ i = 1, 2, ..., n.$$

*Доказательство* следует непосредственно из определения:

$$\mu_{A_{\alpha_i}}(x) = \begin{cases} 1, & \text{если } \mu_A(x) \geq \alpha_i; \\ 0, & \text{если } \mu_A(x) < \alpha_i. \end{cases}$$

Таким образом, функцию принадлежности множества $A$ можно записать в виде

$$\mu_{\max\limits_{\alpha_i} \{\alpha_i \cdot A_{\alpha_i}\}}(x) = \max_{\alpha_i} \left\{ \alpha_i \cdot A_{\alpha_i} \right\} = \max_{\alpha_i \leq \mu_A(x)} \{\alpha_i\} = \mu_A(x).$$

Утверждение доказано.

**Пример 2.8**

Пусть $X = \{x_1, x_2, x_3, x_4, x_5\}$ – универсальное множество; нечеткое множество $A$ задано функцией принадлежности:

$$\mu_A(x) = \{0.7, 0.9, 0, 0.5, 1\}.$$

Согласно определению множества $\alpha$ – уровня имеем обычные подмножества соответствующего уровня:

$\mu_{A_{0.7}}(x) = \{1, 1, 0, 0, 1\}$; $\mu_{A_{0.9}}(x) = \{0, 1, 0, 0, 1\}$; $\mu_{A_0}(x) = \{1, 1, 1, 1, 1\}$;

$\mu_{A_{0.5}}(x) = \{1, 1, 0, 1, 1\}$; $\mu_{A_1}(x) = \{0, 0, 0, 0, 1\}$.

Тогда

$$\mu_A(x) = \{0.7, 0.9, 0, 0.5, 1\} = \max\{(0.7)\cdot\{1, 1, 0, 0, 1\}, (0.9)\cdot\{0, 1, 0, 0, 1\},$$
$$(0)\cdot\{1, 1, 1, 1, 1\}, (0.5)\cdot\{1, 1, 0, 1, 1\}, (1)\cdot\{0, 0, 0, 0, 1\}\}.$$

Если имеется последовательность множеств

$$A_1 \subset A_2 \subset ... \subset A_n$$

и последовательность чисел

$$1 \geq \alpha_1 > \alpha_2 > ... > \alpha_n > 0,$$

то с помощью утверждения о декомпозиции мы можем синтезировать нечеткое множество $A$.

Функция принадлежности данного множества $A$ будет иметь следующий вид:

$$\mu_A(x) = \begin{cases} \alpha_i, & \text{если } \mu_{A_i} = 1 \text{ и } \mu_{A_{i-1}} = 0; \\ 0, & \text{если } \mu_{A_n}(x) = 0, \end{cases}$$

где $\mu_A(x)$ – характеристическая функция множества $A$.

Это очень важное свойство, поскольку оно позволяет наряду с определением нечеткого множества как отображения $\mu_A(x): X \to [0, 1]$ (с. 9) ввести понятие нечеткого множества как отображения $\mu_A(x): 2^X \to [0, 1]$. В некоторых случаях последнее определение оказывается более удобным, т.е. имеется возможность синтезировать нечеткое множество посредством объединения обычных подмножеств.

## Глава 3

# НЕЧЕТКИЕ ОТНОШЕНИЯ

## 3.1. Нечеткие отношения и операции над ними

Понятие отношения играет важную роль в математике и ее приложениях. Это понятие активно используется в *теории автоматов*; *распознавании образов*; при *моделировании структуры сложных систе*м; при анализе процессов *принятия решений* и многих других областях. Понятие отношения также можно обобщить на нечеткий случай. При этом обнаруживаются некоторые новые интересные свойства [11,15]. Так, например, понятие класса эквивалентности заменяется понятием *подобия*. Подобие является не таким жестким, и, поэтому, более подходящим для представления менее определенных и довольно часто встречающихся ситуаций. В рамках нечетких отношений можно ввести некоторые новые типы отношений, например, *сходства* и *несходства*, более адекватно описывающих многие практические ситуации. Возникает новая теория, которая строится на нечетких отношениях, которая позволяет получить, по крайней мере, хорошие описания сложных явлений, до сих пор не поддававшихся формальному описанию.

Пусть $X_1 = \{x\}$ и $X_2 = \{y\}$ – обычные множества. *Прямое произведение* $X_1 \times X_2$ множеств $X_1$ и $X_2$ есть множество упорядоченных пар вида $(x, y)$, то есть

$$X_1 \times X_2 = \{(x, y) : x \in X_1,\ y \in X_2\}$$

***Определение.*** Пусть $M = [0, 1]$ – множество принадлежностей. Тогда нечеткое множество $R$ такое, что

$$\forall (x, y) \in X_1 \times X_2,\ \ \mu_R(x, y) \in M$$

называется *нечетким бинарным отношением* $R$ в $X_1 \times X_2$.

Можно привести следующий пример нечеткого бинарного отношения.

**Пример 3.1**

Пусть $X_1 = \{x_1, x_2, x_3\}$, $X_2 = \{y_1, y_2, y_3, y_4, y_5\}$, $M = [0, 1]$. Тогда нечеткое бинарное отношение может быть задано следующей таблицей:

| $R$ | $y_1$ | $y_2$ | $y_3$ | $y_4$ | $y_5$ |
|---|---|---|---|---|---|
| $x_1$ | 0 | 0.3 | 0.5 | 0.9 | 0.3 |
| $x_2$ | 0.7 | 0.2 | 1 | 0 | 0.1 |
| $x_3$ | 0.9 | 0 | 1 | 0.7 | 0.4 |

Естественным обобщением нечеткого бинарного отношения является нечеткое $n-$ арное отношение. Оно определяется следующим образом. Пусть $R_n = X_1 \times X_2 \times ... \times X_n$ – прямое произведение $n$ множеств, $M = [0, 1]$ – множество принадлежностей. *Нечетким $n$ –арным отношением* называется нечеткое множество в $R_n$, принимающее свои значения в $M$.

Ниже рассматриваются только нечеткие бинарные отношения. Вводимые понятия естественным образом обобщаются на случай $n-$ арных отношений.

Многие понятия и свойства нечетких отношений в данной главе описаны схематично, поскольку подробный анализ их аналогов проведен в предыдущей главе для нечетких множеств.

Нечеткое отношение $L$ *содержит* нечеткое отношение $R$ ($R$ *содержится* в $L$), если для всех пар $(x, y)$ из $X_1 \times X_2$ выполнено $\mu_R(x, y) \le \mu_L(x, y)$, т.е.

$$\forall (x, y) \in X_1 \times X_2 : \mu_R(x, y) \le \mu_L(x, y).$$

Нечеткое отношение $\overline{R}$ является *дополнением* нечеткого отношения $R$, если для всех пар $(x, y)$ из $X_1 \times X_2$ выполнено $\mu_{\overline{R}}(x, y) = 1 - \mu_R(x, y)$.

Нечеткое отношение $Q$ является *объединением* нечетких отношений $R$ и $L$ (т.е. $Q = R \cup L$), если для всех пар $(x, y)$ из $X_1 \times X_2$ выполнено

$$\mu_Q(x, y) = \mu_{R \cup L}(x, y) = \mu_R(x, y) \vee \mu_L(x, y) = \max[\mu_R(x, y), \mu_L(x, y)].$$

Если $R_1$, $R_2$,…, $R_n$ – отношения, то

$$\mu_{R_1 \cup R_2 \cup ... \cup R_n}(x,y) = \underset{R_i}{\vee} \mu_{R_i}(x,y) = \max[\mu_{R_1}(x,y),..., \mu_{R_n}(x,y)]$$

Нечеткое отношение $Q$ является *пересечением* нечетких отношений $R$ и $L$ (т.е. $Q = R \cap L$), если для всех пар $(x,y)$ из $X_1 \times X_2$ выполнено

$$\mu_Q(x,y) = \mu_{R \cap L}(x,y) = \mu_R(x,y) \wedge \mu_L(x,y) = \min[\mu_R(x,y),\ \mu_L(x,y)]$$

Если $R_1$, $R_2$,…, $R_n$ – отношения, то

$$\mu_{R_1 \cap R_2 \cap ... \cap R_n}(x,y) = \underset{R_i}{\wedge} \mu_{R_i}(x,y) = \min[\mu_{R_1}(x,y),..., \mu_{R_n}(x,y)]\,.$$

Нечеткое отношение $Q$ является *алгебраическим произведением* нечетких отношений $R$ и $L$ (т.е. $Q = R \bullet L$), если для всех пар $(x,y)$ из $X_1 \times X_2$ выполнено $\mu_Q(x,y) = \mu_{R \bullet L}(x,y) = \mu_R(x,y) \cdot \mu_L(x,y)$.

Нечеткое отношение $Q$ является *алгебраической суммой* нечетких отношений $R$ и $L$ (т.е. $Q = R \hat{+} L$), если для всех пар $(x,y)$ из $X_1 \times X_2$ выполнено

$$\mu_Q(x,y) = \mu_{R \hat{+} L}(x,y) = \mu_R(x,y) + \mu_L(x,y) - \mu_R(x,y) \cdot \mu_L(x,y)\,.$$

Нечеткое отношение $Q$ является *дизъюнктивной суммой* нечетких отношений $R$ и $L$ (т.е. $Q = R \oplus L$), если для всех пар $(x,y)$ из $X_1 \times X_2$ выполнено $Q = \left(R \cap \overline{L}\right) \cup \left(\overline{R} \cap L\right)$.

Для нечетких отношений вводятся понятия *проекций* следующим образом. *Первая проекция нечеткого бинарного отношения* $R$ определяется функцией принадлежности

$$\mu_R^{(1)}(x) = \max_y \mu_R(x,y)$$

*Вторая проекция нечеткого бинарного отношения* $R$ определяется функцией принадлежности

$$\mu_R^{(2)}(y) = \max_x \mu_R(x,y)$$

*Глобальной проекцией* $h(R)$ нечеткого бинарного отношения $R$ называется вторая проекция первой проекции (или наоборот):

$$h(R) = \max_x \max_y \mu_R(x,y) = \max_y \max_x \mu_R(x,y)\,.$$

*Носителем* $S(R)$ нечеткого отношения $R$ называется обычное множество упорядоченных пар $(x,y)$, для которых функция принадлежности положительна, то есть

$$S(R)=\{(x,y):\ \mu_R(x,y)>0\}\ .$$

Не трудно видеть, что, если мы заменим в определениях введенных операций функции принадлежности $\mu(x,y)$ на характеристические функции обычных подмножеств, то получим определения соответствующих операций, принятые в теории обычных отношений. В этом смысле теория нечетких отношений является обобщением теории (обычных) отношений.

Так как нечеткие отношения есть нечеткие множества на специфическом универсальном множестве, основные операции над ними обладают теми же свойствами. Доказательства свойств аналогичны доказательствам, приведенным в главе 2.

Как и в случае нечетких множеств, нечеткое отношение можно декомпозировать на специальным образом устроенную систему обычных отношений и, наоборот, из системы обычных отношений, удовлетворяющим некоторым требованиям, можно синтезировать нечеткое отношение. Такая декомпозиция и синтез нечетких отношений производится с помощью подмножеств $\alpha$ – уровня нечеткого отношения.

Пусть $\alpha\in[0,1]$, $R\in X\times X$. *Подмножеством* $\alpha-$*уровня* нечеткого отношения $R$ называется обычное подмножество $R_\alpha=\{(x,y):\mu_R(x,y)\ge\alpha\}$.

Функция принадлежности подмножества $\alpha$ – уровня задается выражением

$$\mu_{R_\alpha}(x,y)=\begin{cases}1,\ \mu_R(x,y)\ge\alpha\\ 0,\ \mu_R(x,y)<\alpha\end{cases}$$

Очевидным, является свойство: если $\alpha_1\ge\alpha_2$, то $R_{\alpha_1}\subset R_{\alpha_2}$.

**Утверждение 3.1** *(о декомпозиции)*. Любое отношение $R$ можно представить в форме

$$R=\max_\alpha \alpha\times R_\alpha\ .$$

*Доказательство.* Справедливы следующие соотношения:

$$\mu_R(x,\ y)=\max_{\alpha\le\mu_R(x,\ y)}\alpha=\max_\alpha \alpha\times\mu_{R_\alpha}(x,\ y)=\mu_{\max\limits_\alpha \alpha\times R_\alpha(x,\ y)}\ .$$

Утверждение доказано.

В случае конечных или счетных универсальных множеств очевидна интерпретация нечеткого отношения в виде взвешенного графа $G$, в котором каждая пара вершин $(x, y)$ из $X_1 \times X_2$ соединяется ребром с весом $\mu_G(x, y)$.

***Определение.*** Нечеткое множество $G$, такое, что

$$\forall (x, y) \in X_1 \times X_2 : \mu_G(x, y) \in M,$$

где $M = [0, 1]$ – множество принадлежностей элементов $X_1 \times X_2$, называется *нечетким графом*.

Таким образом, нечеткий граф можно рассматривать как графическую интерпретацию нечеткого отношения. Нечеткий граф можно представить в виде матрицы, как и нечеткое отношение.

**Пример 3.2**

Пусть $X_1 = \{x_1, x_2\}$ и $X_2 = \{y_1, y_2, y_3\}$.

Тогда множество упорядоченных пар $(x_i, y_j)$, $i = 1, 2$, $j = 1, 2, 3$ определяет прямое произведение $X_1 \times X_2$, т.е.

$$X_1 \times X_2 = \{(x_1, y_1), (x_1, y_2), (x_1, y_3), (x_2, y_1), (x_2, y_2), (x_2, y_3)\}.$$

На прямом произведении задан нечеткий граф $G$ (нечеткое отношение) следующей функцией принадлежности:

$$\mu_G(x, y) = \{0.5, 1, 0, 0.8, 0.2, 0.3\}.$$

Этот нечеткое $G$ можно представить в виде матрицы (рис. 3.1) и в виде графа (рис.3.2).

| $G$ | $y_1$ | $y_2$ | $y_3$ |
|---|---|---|---|
| $x_1$ | 0.5 | 1 | 0 |
| $x_2$ | 0.8 | 0.2 | 0.3 |

Рис. 3.1

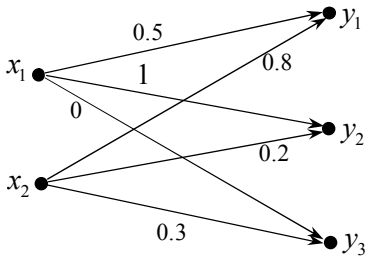



Рис. 3.2

## 3.2. Композиция нечетких отношений

Операция композиции нечетких отношений $R_1$ в $X \times Y$ и $R_2$ в $Y \times Z$ позволяет определить нечеткое отношение в $X \times Z$.

### $(\max - \min)$ – *композиция и ее свойства*

Пусть $R_1$ есть нечеткое отношение в $X \times Y$, $R_2$ – нечеткое отношение в $Y \times Z$. Тогда $(\max - \min)$ – композиция $R_1 \circ R_2$ определяется выражением

$$\mu_{R_1 \circ R_2}(x, z) = \max_y \left[ \min \left\{ \mu_{R_1}(x, y), \mu_{R_2}(y, z) \right\} \right], \qquad (3.1)$$

где $x \in X$, $y \in Y$, $z \in Z$.

Вычисление композиции нечетких отношений аналогично вычислению произведения матриц («строка на столбец»), только вместо произведения и суммы выполняются операции взятия минимума и максимума соответственно.

**Пример 3.3**

Пусть

$X = \{x_1, x_2, x_3\}$, $Y = \{y_1, y_2, y_3, y_4, y_5\}$, $Z = \{z_1, z_2, z_3, z_4\}$.

Нечеткое отношение $R_1$ на $X \times Y$ задано следующей матрицей:

| $R_1$ | $y_1$ | $y_2$ | $y_3$ | $y_4$ | $y_5$ |
|---|---|---|---|---|---|
| $x_1$ | 0.1 | 0.2 | 0.5 | 0.7 | 0.3 |
| $x_2$ | 0.3 | 0 | 1 | 0.3 | 0.7 |
| $x_3$ | 0.1 | 0.8 | 0 | 0 | 1 |

Нечеткое отношение $R_2$ на $Y \times Z$ задано следующей матрицей:

| $R_2$ | $z_1$ | $z_2$ | $z_3$ | $z_4$ |
|---|---|---|---|---|
| $y_1$ | 0.9 | 0 | 0.4 | 0.7 |
| $y_2$ | 0.3 | 0.4 | 0.9 | 1 |
| $y_3$ | 1 | 0.4 | 0.2 | 0 |
| $y_4$ | 0.3 | 0.4 | 0.7 | 0.2 |
| $y_5$ | 0.5 | 0.5 | 0.9 | 0.1 |

Например, для определения значения функции принадлежности $\mu_{R_1 \circ R_2}(x_1, z_1)$ имеем:

$$\min\left\{\mu_{R_1}(x_1, y_1), \mu_{R_2}(y_1, z_1)\right\} = \min\{0.1, 0.9\} = 0.1;$$

$$\min\left\{\mu_{R_1}(x_1, y_2), \mu_{R_2}(y_2, z_1)\right\} = \min\{0.2, 0.3\} = 0.2;$$

$$\min\left\{\mu_{R_1}(x_1, y_3), \mu_{R_2}(y_3, z_1)\right\} = \min\{0.5, 1\} = 0.5;$$

$$\min\left\{\mu_{R_1}(x_1, y_4), \mu_{R_2}(y_4, z_1)\right\} = \min\{0.7, 0.3\} = 0.3;$$

$$\min\left\{\mu_{R_1}(x_1, y_5), \mu_{R_2}(y_5, z_1)\right\} = \min\{0.3, 0.5\} = 0.3$$

$$\max_{y_i}\left[\min\left\{\mu_{R_1}(x_1, y_i), \mu_{R_2}(y_i, z_1)\right\}\right] = \max[0.1, 0.2, 0.5, 0.3, 0.3] = 0.5.$$

Аналогичные действия выполняются для остальных элементов матрицы значений функции принадлежности.

Тогда нечеткое отношение $R_1 \circ R_2$ определено на $X \times Z$ и выражается следующей матрицей:

| $R_1 \circ R_2$ | $z_1$ | $z_2$ | $z_3$ | $z_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.4 | 0.7 | 0.2 |
| $x_2$ | 1 | 0.5 | 0.7 | 0.8 |
| $x_3$ | 0.5 | 0.5 | 0.9 | 0.8 |

Среди свойств $(\max - \min)$ – композиции можно выделить следующие:

– ассоциативность: $(R_1 \circ R_2) \circ R_3 = R_1 \circ (R_2 \circ R_3)$. Если отношение $R$ определено на $X \times X$, т.е $R \subset X \times X$, то можно записать $R \circ R = R^2$ и в общем случае $\underbrace{R \circ R \circ \ldots \circ R}_{k \text{ раз}} = R^k$.

– дистрибутивность относительно объединения:

$$R_1 \circ (R_2 \cup R_3) = (R_1 \circ R_2) \cup (R_1 \circ R_3)$$

– не дистрибутивность относительно пересечения:

$$R_1 \circ (R_2 \cap R_3) \neq (R_1 \circ R_2) \cap (R_1 \circ R_3)$$

– монотонность:

$$A \subset B \Rightarrow R \circ A \subset R \circ B$$

Доказательства данных свойств достаточно очевидны.

### $(\max - *)$ – *композиция и ее свойства*

Понятие $(\max - \min)$ – композиции можно обобщить следующим образом: заменить в (3.1) операцию $\min$ на любую другую, для которой выполняются свойство *ассоциативности* и *монотонного неубывания* по каждому аргументу.

Пусть операция '$*$' является ассоциативной и монотонно не убывает по каждому аргументу. Тогда $(\max - *)$ – композиция определяется следующей функцией принадлежности

$$\mu_{R_1 * R_2}(x, z) = \max_y \left[ \mu_{R_1}(x, y) * \mu_{R_2}(y, z) \right],$$

где $x \in X$, $y \in Y$, $z \in Z$.

Важным частным случаем $(\max - *)$ – композиции является $(\max - \cdot)$ – композиция, или $(\max - \text{prod})$ – композиция. В этом случае операция '$*$' – это произведение, и, таким образом, (3.1) принимает следующий вид:

$$\mu_{R_1 \cdot R_2}(x, z) = \max_y \left[ \mu_{R_1}(x, y) \cdot \mu_{R_2}(y, z) \right],$$

где $x \in X$, $y \in Y$, $z \in Z$.

Другим примером $(\max - *)$ – композиции в случае, когда множество принадлежностей $M = [0, 1]$, является композиция, получаемая из (3. ) заменой операции $\min$ на среднее арифметическое, или $(\max - \text{average})$ – композиция:

$$\mu_{R_1 \cdot R_2}(x, z) = \max_y \left[ \frac{1}{2} \left( \mu_{R_1}(x, y) + \mu_{R_2}(y, z) \right) \right],$$

где $x \in X$, $y \in Y$, $z \in Z$.

Другие конкретные $(\max - *)$ – композиции и их свойства рассматриваются в работах [8], [15].

Выбор того или иного варианта $(\max - *)$ – композиции в приложениях определяется свойствами задачи.

### $(\min - \max)$ – *композиция*

Минимаксная композиция используется аналогично максиминной композиции в некоторых типах задач принятия решений.

*Минимаксная композиция* $R_1 \circ R_2$ нечетких отношений $R_1$ и $R_2$ на множестве $X$ определяется функцией принадлежности вида

$$\mu_{R_1 \circ R_2}(x, z) = \min_{y \in X} \left[ \max \left\{ \mu_{R_1}(x, y), \mu_{R_2}(y, z) \right\} \right].$$

## 3.3. Транзитивное замыкание нечеткого бинарного отношения

Рассмотрим случай $X_1 = X_2 = X$, $M = [0, 1]$. Для нечетких отношений довольно естественно вводятся свойства рефлексивности, симметричности и транзитивности.

Нечеткое бинарное отношение $R$ называется ***рефлексивным***, если $\mu_R(x,x) = 1$, $\forall x \in X$.

Нечеткое бинарное отношение $R$ называется ***симметричным***, если $\mu_R(x,y) = \alpha \Rightarrow \mu_R(y,x) = \alpha$, $\forall x, y \in X$.

Нечеткое бинарное отношение $R$ называется ***транзитивным***, если $\mu_R(x, z) \geq \max\limits_y \left[ \min\left( \mu_R(x, y), \mu_R(y, z) \right) \right]$, $\forall x, y, z \in X$.

**Пример 3.4**

Отношение «$x$ близко к $y$» рефлексивно, симметрично, но не транзитивно. Отношение «$x$ много больше $y$» – не рефлексивно, не симметрично, но транзитивно.

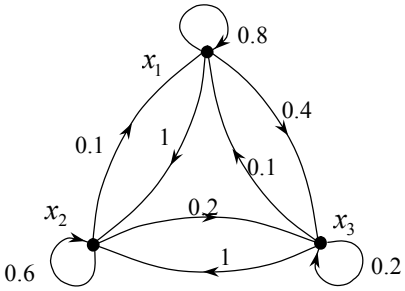
Для проверки свойства транзитивности некоторого бинарного отношения $R$ необходимо выполнить $n^2$ раз $n$ операций, т.е. $n^3$ операций, где $n$ – размерность универсального множества.

**Пример 3.5**

Отношение $R$ на универсальном множестве $X = \{x_1, x_2, x_3\}$ не является рефлексивным, не является симметричным. Это очевидно, т.к. следует непосредственно из определений рефлексивности и симметричности. Необходимо проверить транзитивность отношения $R$.

| $R$ | $x_1$ | $x_2$ | $x_3$ |
|---|---|---|---|
| $x_1$ | 0.8 | 1 | 0.4 |
| $x_2$ | 0.1 | 0.6 | 0.2 |
| $x_3$ | 0.1 | 1 | 0.4 |

*а)*




*б)*

Рис. 3.3

Для проверки используем условие транзитивности

$$\mu_R(x, z) \geq \max_y \left[ \min\left( \mu_R(x, y), \mu_R(y, z) \right) \right], \ \forall x, y, z \in X .$$

Распишем проверку по всем дугам графа, как отношения:

– дуга $(x_1, x_1)$

$$\min\left( \mu_R(x_1, x_1), \mu_R(x_1, x_1) \right) = \min(0.8, 0.8) = 0.8 ,$$

$$\min\left( \mu_R(x_1, x_2), \mu_R(x_2, x_1) \right) = \min(1, 0.1) = 0.1 ,$$

$$\min\left( \mu_R(x_1, x_3), \mu_R(x_3, x_1) \right) = \min(0.4, 0.1) = 0.1 ,$$

$$\max(0.8, 0.1, 0.1) = 0.8 , \qquad \mu_R(x_1, x_1) = 0.8 \geq 0.8 ;$$

– дуга $(x_1, x_2)$

$$\min\left( \mu_R(x_1, x_1), \mu_R(x_1, x_2) \right) = \min(0.8, 1) = 0.8 ,$$

$$\min\left( \mu_R(x_1, x_2), \mu_R(x_2, x_2) \right) = \min(1, 0.6) = 0.6 ,$$

$$\min\left( \mu_R(x_1, x_3), \mu_R(x_3, x_2) \right) = \min(0.4, 1) = 0.4 ,$$

$$\max(0.8, 0.6, 0.4) = 0.8 , \qquad \mu_R(x_1, x_2) = 1 \geq 0.8 ;$$

– дуга $(x_1, x_3)$

$$\min\left( \mu_R(x_1, x_1), \mu_R(x_1, x_3) \right) = \min(0.8, 0.4) = 0.4 ,$$

$$\min\left( \mu_R(x_1, x_2), \mu_R(x_2, x_3) \right) = \min(1, 0.2) = 0.2 ,$$

$$\min\left( \mu_R(x_1, x_3), \mu_R(x_3, x_3) \right) = \min(0.4, 0.2) = 0.2 ,$$

$$\max(0.4, 0.2, 0.2) = 0.4 , \qquad \mu_R(x_1, x_2) = 0.4 \geq 0.4 ;$$

– дуга $(x_2, x_1)$

$$\min\left( \mu_R(x_2, x_1), \mu_R(x_1, x_1) \right) = \min(0.1, 0.8) = 0.1 ,$$

$$\min\left( \mu_R(x_2, x_2), \mu_R(x_2, x_1) \right) = \min(0.6, 0.1) = 0.1 ,$$

$$\min\left( \mu_R(x_2, x_3), \mu_R(x_3, x_1) \right) = \min(0.2, 0.1) = 0.1 ,$$

$$\max(0.1, 0.1, 0.1) = 0.1 , \qquad \mu_R(x_2, x_1) = 0.1 \geq 0.1 ;$$

– дуга $(x_2, x_2)$

$$\min\left( \mu_R(x_2, x_1), \mu_R(x_1, x_2) \right) = \min(0.1, 1) = 0.1 ,$$

$$\min\left( \mu_R(x_2, x_2), \mu_R(x_2, x_2) \right) = \min(0.6, 0.6) = 0.6 ,$$

$$\min\left( \mu_R(x_2, x_3), \mu_R(x_3, x_2) \right) = \min(0.2, 1) = 0.2 ,$$

$$\max(0.1, 0.6, 0.2) = 0.6 , \qquad \mu_R(x_2, x_1) = 0.6 \geq 0.6 ;$$

– дуга $(x_2, x_3)$

$$\min\left( \mu_R(x_2, x_1), \mu_R(x_1, x_3) \right) = \min(0.1, 0.4) = 0.1 ,$$

$\min\left(\mu_R(x_2, x_2),\ \mu_R(x_2, x_3)\right) = \min(0.6,\ 0.2) = 0.2$,

$\min\left(\mu_R(x_2, x_3),\ \mu_R(x_3, x_3)\right) = \min(0.3,\ 0.2) = 0.2$,

$\max(0.1,\ 0.2,\ 0.2) = 0.2$, $\mu_R(x_2, x_3) = 0.2 \geq 0.2$;

– дуга $(x_3, x_1)$

$\min\left(\mu_R(x_3, x_1),\ \mu_R(x_1, x_1)\right) = \min(0.1,\ 0.8) = 0.1$,

$\min\left(\mu_R(x_3, x_2),\ \mu_R(x_2, x_1)\right) = \min(1,\ 0.1) = 0.1$,

$\min\left(\mu_R(x_3, x_3),\ \mu_R(x_3, x_1)\right) = \min(0.2,\ 0.1) = 0.1$,

$\max(0.1,\ 0.1,\ 0.1) = 0.1$, $\mu_R(x_3, x_1) = 0.1 \geq 0.1$;

– дуга $(x_3, x_2)$

$\min\left(\mu_R(x_3, x_1),\ \mu_R(x_1, x_2)\right) = \min(0.1,\ 1) = 0.1$,

$\min\left(\mu_R(x_3, x_2),\ \mu_R(x_2, x_2)\right) = \min(1,\ 0.6) = 0.6$,

$\min\left(\mu_R(x_3, x_3),\ \mu_R(x_3, x_2)\right) = \min(0.2,\ 1) = 0.2$,

$\max(0.1,\ 0.6,\ 0.2) = 0.6$, $\mu_R(x_3, x_1) = 1 \geq 0.6$;

– дуга $(x_3, x_3)$

$\min\left(\mu_R(x_3, x_1),\ \mu_R(x_1, x_3)\right) = \min(0.1,\ 0.4) = 0.1$,

$\min\left(\mu_R(x_3, x_2),\ \mu_R(x_2, x_3)\right) = \min(1,\ 0.2) = 0.2$,

$\min\left(\mu_R(x_3, x_3),\ \mu_R(x_3, x_3)\right) = \min(0.2,\ 0.2) = 0.2$,

$\max(0.1,\ 0.2,\ 0.2) = 0.2$, $\mu_R(x_2, x_3) = 0.2 \geq 0.2$;

Таким образом, отношение $R$ транзитивно.

Для не транзитивных бинарных отношений вводится операция *транзитивного замыкания.*

Пусть $R$ – нечеткое отношение в $X \times X$. Определим $R^2 = R \circ R$ функцией принадлежности

$$\mu_{R^2}(x,\ z) = \max_y \left[\min\left(\mu_R(x,\ y),\ \mu_R(y,\ z)\right)\right],$$

где $x, y, z \in X$.

Сравнивая последнее выражение с определением транзитивности для нечетких бинарных отношений, не трудно увидеть, что свойство транзитивности можно записать в следующем виде:

$$R^2 \subset R.$$

Аналогично можно определить по индукции $R^k$:

$$R^k = R \circ R^{k-1},\ \ k = 2, 3, \ldots.$$

Пусть $R^2 \subset R$ и $R^{k+1} \subset R^k$, $k = 2,3,...$ Тогда, очевидно, $R^k \subset R$.

***Определение.*** *Транзитивным замыканием* нечеткого бинарного отношения называется отношение

$$\hat{R} = R \cup R^2 \cup R^3 \cup \ldots . \tag{3.2}$$

Справедливо следующее утверждение.

**Утверждение 3.1.** Транзитивное замыкание любого бинарного отношения есть транзитивное бинарное отношение

*Доказательство*

Согласно (3.2) можно записать:

$$\hat{R}^2 = \hat{R} \circ \hat{R} = R^2 \cup R^3 \cup R^4 \cup \ldots .$$

Сравнивая последнее выражение с (3.2), можно записать: $\hat{R}^2 \subset \hat{R}$, что и доказывает транзитивность $\hat{R}$.

Утверждение доказано.

Как проверить, построили мы или нет транзитивное замыкание конкретного нечеткого бинарного отношения? Ответ на этот вопрос дает следующее утверждение.

**Утверждение 3.2.** Пусть $R$ – некоторое нечеткое бинарное отношение. Если существует $k$ такое, что $R^{k+1} = R^k$, то $\hat{R} = R \cup R^2 \cup \ldots \cup R^k$.

***Доказательство.*** Доказательство непосредственно следует из соотношений:

$$\hat{R} = R \cup R^2 \cup \ldots \cup R^k \cup R^{k+1} \cup R^{k+2} \cup \ldots =$$
$$= R \cup R^2 \cup \ldots \cup R^k \cup R^k \cup R^k \cup \ldots = R \cup R^2 \cup \ldots \cup R^k$$

Утверждение доказано.

Всегда ли композиция $R_1 \circ R_2$ или $R_2 \circ R_1$ двух транзитивных отношений $R_1$ и $R_2$ есть транзитивное отношение? Можно привести пример, что это свойство не всегда выполняется.

**Пример 3.4.**

Пусть $R_1$ и $R_2$ задаются следующими таблицами:

| $R_1$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.9 | 0 | 0 |
| $x_2$ | 0 | 0.7 | 0 | 0 |
| $x_3$ | 0 | 1 | 0.1 | 0.1 |
| $x_4$ | 0 | 1 | 0.4 | 1 |

| $R_2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.7 | 0 | 0 | 0 |
| $x_2$ | 0.8 | 1 | 0.6 | 0.6 |
| $x_3$ | 0 | 0 | 0.5 | 0.5 |
| $x_4$ | 0 | 0 | 0.2 | 0.4 |

Отношение $R_1$ транзитивно, т.к. $R_1^2 \subset R_1$:

| $R_1$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.9 | 0 | 0 |
| $x_2$ | 0 | 0.7 | 0 | 0 |
| $x_3$ | 0 | 1 | 0.1 | 0.1 |
| $x_4$ | 0 | 1 | 0.4 | 1 |

$\circ$

| $R_1$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.9 | 0 | 0 |
| $x_2$ | 0 | 0.7 | 0 | 0 |
| $x_3$ | 0 | 1 | 0.1 | 0.1 |
| $x_4$ | 0 | 1 | 0.4 | 1 |

$=$

$=$

| $R_1^2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.7 | 0 | 0 |
| $x_2$ | 0 | 0.7 | 0 | 0 |
| $x_3$ | 0 | 0.7 | 0.1 | 0.1 |
| $x_4$ | 0 | 1 | 0.4 | 1 |

Отношение $R_2$ транзитивно, т.к. $R_2^2 \subset R_2$:

| $R_2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.7 | 0 | 0 | 0 |
| $x_2$ | 0.8 | 1 | 0.6 | 0.6 |
| $x_3$ | 0 | 0 | 0.5 | 0.5 |
| $x_4$ | 0 | 0 | 0.2 | 0.4 |

$\circ$

| $R_2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.7 | 0 | 0 | 0 |
| $x_2$ | 0.8 | 1 | 0.6 | 0.6 |
| $x_3$ | 0 | 0 | 0.5 | 0.5 |
| $x_4$ | 0 | 0 | 0.2 | 0.4 |

$=$

$=$

| $R_2^2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.7 | 0 | 0 | 0 |
| $x_2$ | 0.8 | 1 | 0.6 | 0.6 |
| $x_3$ | 0 | 0 | 0.5 | 0.5 |
| $x_4$ | 0 | 0 | 0.2 | 0.4 |

Теперь подсчитаем $R_2 \circ R_1$

| $R_1$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.9 | 0 | 0 |
| $x_2$ | 0 | 0.7 | 0 | 0 |
| $x_3$ | 0 | 1 | 0.1 | 0.1 |
| $x_4$ | 0 | 1 | 0.4 | 1 |

$\circ$

| $R_2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.7 | 0 | 0 | 0 |
| $x_2$ | 0.8 | 1 | 0.6 | 0.6 |
| $x_3$ | 0 | 0 | 0.5 | 0.5 |
| $x_4$ | 0 | 0 | 0.2 | 0.4 |

$=$

$=$

| $R_2 \circ R_1$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.8 | 0.9 | 0.6 | 0.6 |
| $x_2$ | 0.7 | 0.7 | 0.6 | 0.6 |
| $x_3$ | 0.8 | 1 | 0.6 | 0.6 |
| $x_4$ | 0.8 | 1 | 0.6 | 0.6 |

$\left(R_2 \circ R_1\right)^2$:

| $R_2 \circ R_1$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.8 | 0.9 | 0.6 | 0.6 |
| $x_2$ | 0.7 | 0.7 | 0.6 | 0.6 |
| $x_3$ | 0.8 | 1 | 0.6 | 0.6 |
| $x_4$ | 0.8 | 1 | 0.6 | 0.6 |

$\circ$

| $R_2 \circ R_1$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.8 | 0.9 | 0.6 | 0.6 |
| $x_2$ | 0.7 | 0.7 | 0.6 | 0.6 |
| $x_3$ | 0.8 | 1 | 0.6 | 0.6 |
| $x_4$ | 0.8 | 1 | 0.6 | 0.6 |

$=$

$=$

| $(R_2 \circ R_1)^2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.8 | 0.8 | 0.6 | 0.6 |
| $x_2$ | 0.7 | 0.7 | 0.6 | 0.6 |
| $x_3$ | 0.8 | 0.8 | 0.6 | 0.6 |
| $x_4$ | 0.8 | 0.8 | 0.6 | 0.6 |

Сравнивая полученные результаты, видим, что $\left(R_2 \circ R_1\right)^2 \subset R_2 \circ R_1$, и, следовательно, отношение $R_2 \circ R_1$ является транзитивным.

Теперь подсчитаем $R_1 \circ R_2$

| $R_2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.7 | 0 | 0 | 0 |
| $x_2$ | 0.8 | 1 | 0.6 | 0.6 |
| $x_3$ | 0 | 0 | 0.5 | 0.5 |
| $x_4$ | 0 | 0 | 0.2 | 0.4 |

$\circ$

| $R_1$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.9 | 0 | 0 |
| $x_2$ | 0 | 0.7 | 0 | 0 |
| $x_3$ | 0 | 1 | 0.1 | 0.1 |
| $x_4$ | 0 | 1 | 0.4 | 1 |

$=$

$=$

| $R_1 \circ R_2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.7 | 0 | 0 |
| $x_2$ | 0.5 | 0.8 | 0.4 | 0.6 |
| $x_3$ | 0 | 0.5 | 0.4 | 0.5 |
| $x_4$ | 0 | 0.4 | 0.4 | 0.4 |

и $\left(R_1 \circ R_2\right)^2$:

| $R_1 \circ R_2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.7 | 0 | 0 |
| $x_2$ | 0.5 | 0.8 | 0.4 | 0.6 |
| $x_3$ | 0 | 0.5 | 0.4 | 0.5 |
| $x_4$ | 0 | 0.4 | 0.4 | 0.4 |

$\circ$

| $R_1 \circ R_2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.7 | 0 | 0 |
| $x_2$ | 0.5 | 0.8 | 0.4 | 0.6 |
| $x_3$ | 0 | 0.5 | 0.4 | 0.5 |
| $x_4$ | 0 | 0.4 | 0.4 | 0.4 |

$=$

$=$

| $(R_1 \circ R_2)^2$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 0.5 | 0.7 | 0.4 | 0.6 |
| $x_2$ | 0.5 | 0.8 | 0.4 | 0.6 |
| $x_3$ | 0.5 | 0.5 | 0.4 | 0.5 |
| $x_4$ | 0.4 | 0.4 | 0.4 | 0.4 |

Сравнивая полученные результаты, видим, что $\left(R_1 \circ R_2\right)^2 \not\subset R_1 \circ R_2$, и, следовательно, отношение $R_1 \circ R_2$ является транзитивным.

Таким образом, композиция двух транзитивных отношений не всегда дает транзитивное отношение.

## 3.4. Некоторые специальные типы нечетких отношений

Транзитивное и рефлексивное нечеткое бинарное отношение называется *нечетким отношением предпорядка*.

Для нечетких отношений предпорядка справедливо следующее утверждение

**Утверждение 3.3.** Если $R$ – предпорядок, то $R^k = R$, $\forall k$ $(k = 1,2,3,...)$.

*Доказательство.* Согласно определению $(\min - \max)$ – композиции

$$\mu_{R^2}(x,\, z) = \max_y \left[ \min\left(\mu_R(x,\, y),\, \mu_R(y,\, z)\right)\right] \tag{3.3}$$

При $y = x$ и $y = z$ правая часть (3.3) содержит два равных члена $\min\left(\mu_R(x,x),\, \mu_R(x,z)\right)$ и $\min\left(\mu_R(x,z),\, \mu_R(z,z)\right)$, которые равны $\mu_R(x,z)$, поскольку в силу рефлексивности $R$

$$\mu_R(x,x) = \mu_R(z,z) = 1$$

Таким образом,

$$\mu_{R^2}(x,z) \le \mu_R(x,z)\,. \tag{3.4}$$

С другой стороны, по определению рефлексивности

$$\mu_R(x,\, z) \ge \max_y \left[ \min\left(\mu_R(x,\, y),\, \mu_R(y,\, z)\right)\right] \tag{3.5}$$

Сравнивая (3.3), (3.4) и (3.5) и обозначая через $\tau(x,y)$ выражение $\max\limits_y \left[ \min\left(\mu_R(x,\, y),\, \mu_R(y,\, z)\right)\right]$, можем написать:

$$\tau(x,y) = \mu_{R^2}(x,z) \le \mu_R(x,z) \ge \tau(x,y)$$

Данное соотношение возможно только при замене $\le$ и $\ge$ на $=$. Таким образом, получаем, что $R^2 = R$. Согласно определению, $R^k = R \circ R^{k-1}$. Поэтому из равенства $R^2 = R$ получаем доказательство утверждения.

Как следствие из данного утверждения, получаем, что, если $R$ – предпорядок, то $\hat{R} = R$.

Нечеткое бинарное отношение называется ***антисимметричным***, если

$$\forall (x, y) \in X \times X : x \neq y \quad \mu_R(x, y) \neq \mu_R(y, x) \text{ или } \mu_R(x, y) = \mu_R(y, x) = 0.$$

Нечеткое бинарное отношение называется ***совершенно антисимметричным***, если

$$\forall (x, y) \in X \times X : x \neq y \qquad \mu_R(x, y) > 0 \Rightarrow \mu_R(y, x) = 0.$$

Из определений непосредственно следует, что любое совершенное антисимметричное отношение будет и антисимметричным отношением.

Антисимметричное нечеткое отношение предпорядка называется нечетким ***отношением порядка.***

В различных работах по нечетким отношениям вводятся понятия нечеткого отношения полного порядка, частичного порядка, совершенного порядка и т.п. Оказывается, что эти отношения индуцируют соответствующие порядки в универсуме $X$. В рамках этих работ изучаются классические для отношений порядка понятия: наибольший и наименьший элементы, верхний и нижний пределы, максимальная цепь и т.п. Это довольно большой и очень интересный раздел теории нечетких отношений. Однако, даже для ознакомления с ним необходимо достаточно подробное ознакомление с обычными отношениями порядка. Объем курса, к сожалению, не позволяет провести такой экскурс, поэтому остановимся на других свойствах нечетких бинарных отношений.

Рассмотрим более подробно отношение подобия и связанные с ним отношения различия, сходства и их свойства. Эти отношения интересны для тем, что они имеют интересные приложения в задачах обработки информации, демонстрирующие новые возможности такой обработки, предоставляемые введением и учетом нечеткости.

***Нечетким отношением подобия*** называется транзитивное рефлексивное симметричное нечеткое бинарное отношение.

Очевидно, что отношение подобия является предпорядком.

**Пример 3.5.** Для любого $a$: $0 \le a \le 1$ нечеткое бинарное отношение является отношением подобия

| $R$ | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| $x_1$ | 1 | $a$ | $a$ | $a$ |
| $x_2$ | $a$ | 1 | $a$ | $a$ |
| $x_3$ | $a$ | $a$ | 1 | $a$ |
| $x_4$ | $a$ | $a$ | $a$ | 1 |

Итак, пусть $R$ – отношение подобия, определенное на $X \times X$ . Теперь рассмотрим его дополнение в $X \times X$, т.е. отношение $\overline{R}$, определяемое функцией принадлежности

$$\mu_{\overline{R}}(x,y) = 1 - \mu_R(x,y) \quad \forall (x,y) \in X \times X .$$

Достаточно очевидно, что отношение $\overline{R}$ является симметричным и антирефлексивным (т.е. $\mu_{\overline{R}}(x,y) = 0 \quad \forall (x,y) \in X \times X$). Из $(\max - \min)$-транзитивности $R$ следует, что

$$1 - \mu_{\overline{R}}(x,y) \ge \max_y \min\left[\left(1 - \mu_{\overline{R}}(x,y)\right), \left(1 - \mu_{\overline{R}}(y,z)\right)\right]. \qquad (3.6)$$

Из теоремы Де-Моргана $\overline{R} \cap \overline{L} = \overline{R \cup L}$ непосредственно следует, что

$$\min\left[\left(1 - \mu_{\overline{R}}(x,y)\right), \left(1 - \mu_{\overline{R}}(y,z)\right)\right] = 1 - \max\left[\mu_{\overline{R}}(x,y),\ \mu_{\overline{R}}(y,z)\right].$$

Таким образом, (3.6) можно переписать следующим образом:

$$1 - \mu_{\overline{R}}(y,z) \ge \max_y \left[1 - \max\left(\mu_{\overline{R}}(x,y),\ \mu_{\overline{R}}(y,z)\right)\right].$$

или, что то же самое,

$$\mu_{\overline{R}}(y,z) \le \min_y \max\left(\mu_{\overline{R}}(x,y),\ \mu_{\overline{R}}(y,z)\right).$$

Последнее выражение определяет для отношения $\overline{R}$ $(\min - \max)$ – *транзитивность*. Нечеткое бинарное отношение, обладающее свойствами антирефлексивности, симметричности и $(\min - \max)$ – транзитивности называется *отношением различия*. Справедливо следующая теорема.

**Теорема 3.1.** Отношение различия $R$ определяет метрику в $X$ следующим образом:

$$d(x,y)=\mu_R(x,y) \quad x\in X\ ,\ \ y\in X\ . \tag{3.7}$$

*Доказательство*. Проверим аксиомы расстояния для (3.7).

1. $d(x,y)\geq 0 \quad \forall x,y\in X$ – очевидно.

2. $d(x,y)\geq 0 \quad \forall x,y\in X$ – следствие антирефлексивности отношения различия $R$.

3. $d(x,y)=d(y,x)$ – следствие симметричности отношения различия $R$.

4. $d(x,y)*d(y,z)\geq d(x,z)$ – следствие (3.7) при определении операции $*$ как $(\min-\max)$ – транзитивности.

Утверждение доказано.

Будем называть расстояние (3. ) $(\min-\max)$ – *расстоянием* в $X$.

Если $R$ – отношение подобия, то, очевидно, $(\min-\max)$ – расстоянием будет $d(x,y)=1-\mu_R(x,y)$.

Итак, имея (построив) отношение подобия, мы можем ввести функцию расстояния на элементах универсального множества. Однако, требование транзитивности часто является слишком «жестким»: реальные результаты экспертных опросов часто не транзитивны. В этой ситуации мы должны либо «заставлять» экспертов давать транзитивные ответы, либо предложить процедуру вычисления расстояния для отношений без транзитивности. Последнее возможно, и базовую роль в такой процедуре играет понятие транзитивного замыкания.

Рефлексивное и симметричное отношение называется *отношением сходства*.

Достаточно очевидно, что, если $R$ – отношение сходства, то $R\circ R$, где «$\circ$» $(\max-\min)$ – композиция, также является отношением сходства, т.е. $(\max-\min)$ – композиция сохраняет рефлексивность и симметричность.

В таком случае, если $R$ – отношение сходства, то $\hat{R}$ – отношение подобия. В таком случае $(\min-\max)$ – расстояние, порожденное $R$, можно определить следующим образом:

$$\mu_R(x,y)=1-\mu_{\hat{R}}(x,y)\ .$$

Итак, теперь мы можем, имея минимальные ограничения на ответы экспертов, вводить достаточно корректно и естественно расстояние ме-

жду различными порой довольно экзотическими понятиями. Например, имея оценки сходства людей различных национальностей, «вычислять» расстояния между национальностями.

Для большого класса задач анализа информации, в частности, проблем распознавания образов и кластер – анализа, большое значение имеет следующая теорема.

**Теорема 3.2.**

Пусть $R$ – отношение подобия в $X \times X$. Тогда

$$R = \max_{\alpha} \alpha \times R_\alpha, \quad 0 < \alpha \le 1,$$

где $R_\alpha$ – отношения эквивалентности в смысле обычной теории множеств.

*Доказательство*. Докажем, что $R_\alpha$ есть отношение эквивалентности, т.е. является рефлексивным, симметричным и транзитивным.

1. Так как $R$ – отношение подобия, то $\mu_R(x,x) = 1$, и, следовательно, $(x,x) \in R_\alpha$, $\forall \alpha \in [0,1]$. Таким образом, $R_\alpha$ рефлексивно $\forall \alpha \in [0,1]$.

2. Пусть $(x,y) \in R_\alpha$, $\alpha \in [0,1]$. Докажем, что $(y,x) \in R_\alpha$. Если $(x,y) \in R_\alpha$, то $\mu_R(x,y) \ge \alpha$, и, в силу симметричности $R$, $\mu_R(y,x) \ge \alpha$. Из последнего неравенства непосредственно следует, что $(y,x) \in R_\alpha$. Таким образом, $R_\alpha$ симметрично $\forall \alpha \in [0,1]$.

3. Пусть $(x,y) \in R_\alpha$, $(y,z) \in R_\alpha$, $\alpha \in [0,1]$. Докажем, что $(x,z) \in R_\alpha$. Если $(x,y) \in R_\alpha$, то $\mu_R(x,y) \ge \alpha$, аналогично $\mu_R(y,z) \ge \alpha$. В силу транзитивности $R$, $\mu_R(x,z) \ge \alpha$. Из последнего неравенства непосредственно следует, что $(x,z) \in R_\alpha$. Таким образом, $R_\alpha$ транзитивно $\forall \alpha \in [0,1]$.

Поскольку $R_\alpha$ рефлексивно, симметрично и транзитивно, то $R_\alpha$ – отношение эквивалентности.

Теорема доказана.

Данная теорема имеет интересные приложения в распознавании образов и кластеризации нечетко определенной информации. Проиллюстрируем это на следующем примере.

**Пример 3.3.**

Пусть $X = \{A, B, C, D, E\}$, $R$ – отношение сходства, заданное следующей функцией принадлежности (3.8)

| $R$ | $A$ | $B$ | $C$ | $D$ | $E$ |
|---|---|---|---|---|---|
| $A$ | 1 | 0.8 | 0.7 | 1 | 0.9 |
| $B$ | 0.8 | 1 | 0.7 | 0.8 | 0.8 |
| $C$ | 0.7 | 0.7 | 1 | 0.7 | 0.7 |
| $D$ | 1 | 0.8 | 0.7 | 1 | 0.9 |
| $E$ | 0.9 | 0.8 | 0.7 | 0.9 | 1 |

(3.8)

В соответствии с теоремой 2 мы можем следующим образом декомпозировать (3.8) на отношения эквивалентности:

$$\begin{array}{|ccccc|}\hline 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.8 & 1 & 0.7 & 0.8 & 0.8 \\ 0.7 & 0.7 & 1 & 0.7 & 0.7 \\ 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.9 & 0.8 & 0.7 & 0.9 & 1 \\ \hline \end{array} = \max_{\alpha} \left\{ 0.7 \times \begin{array}{|ccccc|}\hline 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.8 & 1 & 0.7 & 0.8 & 0.8 \\ 0.7 & 0.7 & 1 & 0.7 & 0.7 \\ 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.9 & 0.8 & 0.7 & 0.9 & 1 \\ \hline \end{array} ;$$

$$0.7 \times \begin{array}{|ccccc|}\hline 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.8 & 1 & 0.7 & 0.8 & 0.8 \\ 0.7 & 0.7 & 1 & 0.7 & 0.7 \\ 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.9 & 0.8 & 0.7 & 0.9 & 1 \\ \hline \end{array} ; \quad 0.7 \times \begin{array}{|ccccc|}\hline 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.8 & 1 & 0.7 & 0.8 & 0.8 \\ 0.7 & 0.7 & 1 & 0.7 & 0.7 \\ 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.9 & 0.8 & 0.7 & 0.9 & 1 \\ \hline \end{array} ; \quad 1.0 \times \begin{array}{|ccccc|}\hline 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.8 & 1 & 0.7 & 0.8 & 0.8 \\ 0.7 & 0.7 & 1 & 0.7 & 0.7 \\ 1 & 0.8 & 0.7 & 1 & 0.9 \\ 0.9 & 0.8 & 0.7 & 0.9 & 1 \\ \hline \end{array} \right\}$$

Таким образом, мы получаем следующую систему вложенных классов, соответствующих данному отношению подобия:

| $\alpha$ – *уровень* | *Количество классов* | *Содержимое классов* |
|---|---|---|
| 0.7 | 1 | $\{A,B,C,D,E\}$ |
| 0.8 | 2 | $\{A,B,D,E\},\{C\}$ |
| 0.9 | 3 | $\{A,D,E\},\{B\},\{C\}$ |
| 1.0 | 4 | $\{A,D\},\{B\},\{C\},\{E\}$ |

Таким образом, мы можем использовать данный аппарат для решения различных задач «вскрытия» внутренней структуры довольно сложных объектов. Например, мы можем построить отношение сходства между политиками (причем, на довольно объективных основаниях - например, по результатам голосования по различным вопросам). После этого мы можем построить систему классов для возможных $\alpha$ – уровней. Эти классы будут описывать возможные коалиции и блоки, которые могут возникнуть при том или ином развитии политического процесса.

Справедлива и теорема, обратная к теореме 3.2.

**Теорема 3.3.**

Пусть $R_{\alpha_i}$ $(1 \le i \le n)$ обычные отношения эквивалентности, $0 < \alpha_1 < \ldots \ \ldots < \alpha_n \le 1$; $R_{\alpha_i} \subset R_{\alpha_i}$. Тогда отношение

$$R = \max_{\alpha} \alpha \times R_\alpha, \ \ 0 < \alpha \le 1,$$

есть отношение подобия.

***Доказательство.*** Докажем, что при выполнении условий теоремы отношение $R$ будет действительно отношением подобия, т.е. обладать свойствами рефлексивности, симметричности и транзитивности.

1. Так как $R_1 \neq \varnothing$, то $(x,x) \in R_1$, и, следовательно,

$$\mu_R(x,y) = 1, \ \forall x \in X.$$

2. Симметричность $R$ вытекает из симметричности каждого $R_\alpha$.

3. Пусть $\mu_R(x,y) = a$, $\mu_R(x,y) = b$. Обозначим $c = \min(a,b)$. Тогда

$$(x,y) \in R_c, \ (y,z) \in R_c.$$

В силу транзитивности $R_c$ получаем, что $(x,z) \in R_c$. Следовательно,

$$\mu_{R^2}(x,\ z) \geq \max_{y}\left[\min\left(\mu_R(x,\ y),\ \mu_R(y,\ z)\right)\right],\ \forall x, y, z \in X\ .$$

Теорема доказана.

## Глава 4

# ОСНОВЫ НЕЧЕТКОЙ ЛОГИКИ

## 4.1. Понятие нечетких булевых переменных

Логика в обычном смысле слова есть представление механизмов мышления, то, что никогда не может быть нечетким, но всегда строгим и формальным. Однако математики, исследовавшие эти механизмы мышления, заметили, что в действительности существует не одна логика (например, булева), а столько логик, сколько мы пожелаем, потому что все определяется выбором соответствующей системы аксиом. Конечно, как только аксиомы выбраны, все утверждения, построенные на их основе, должны быть строго, без противоречий увязаны друг с другом согласно правилам, установленным в этой системе аксиом.

Булева логика – это логика, связанная с булевой теорией множеств; аналогично нечеткая логика связана с теорией нечетких множеств. Как установлено, единой теории нечетких множеств не существует, каждый может построить их столько, сколько пожелает.

Человеческое мышление – это совмещение интуиции и строгости, которое, с одной стороны, рассматривает все в целом или по аналогии (поэтому необходимо нечетко), а с другой стороны – логически и последовательно (необходимо формально) и, значит, в совокупности представляет собой нечеткий механизм. Законы мышления, которые мы захотели бы включить в программы компьютеров, должны быть обязательно *формальными*; законы мышления, проявляемые в диалоге человека с человеком – *нечеткие*. Поэтому можно ли утверждать, что теория нечетких множеств в ее обобщенной форме хорошо приспособлена к человеческому диалогу? Да, если математическое обеспечение, разработанное с учетом нечеткой логики, станет операционным и сможет быть технически реализовано, тогда человеко-машинное общение станет более удобным, быстрым и лучше приспособленным к решению соответствующих проблем.

Пусть $\mu_A(x)$ есть функция принадлежности элемента $x$ нечеткому множеству $A$ универсального множества $X$. Будем предполагать, что множество степеней принадлежности произвольного элемента универсального множества к нечеткому множеству есть

$$M = [0, 1]. \tag{4.1}$$

Будем использовать следующее обозначение:

$$a = \mu_A(x), \quad b = \mu_B(x) \quad \text{и т. д.}. \tag{4.2}$$

Известно, что в бинарной булевой алгебре переменные, обозначаемые здесь через $a$, $b$, …, могут принимать только значения $0$ или $1$. Однако, основные соответствия справедливы как для нечетких булевых переменных с функций принадлежности $M = [0, 1]$, так и для бинарных булевых переменных с функций принадлежности $M = \{0, 1\}$, т. е. понятие *нечеткая булева переменная* обобщает понятие - *булева переменная*.

Термин «*нечеткая логик*а» используется обычно в двух различных значениях. В узком смысле, *нечеткая логика* – это логическое исчисление, являющееся расширением *многозначной логики*. В ее широком смысле, который сегодня является преобладающим в использовании, *нечеткая логика* равнозначна теории нечетких множеств. С этой точки зрения, *нечеткая логика* в узком смысле является разделом нечеткой логики в широком смысле.

## 4.2. Операции над нечеткими булевыми переменными

Пусть $x$ – элемент универсального множества $X$ и $A$, $B$, …– нечеткие множества этого универсального множества. Пусть

$$a = \mu_A(x), \quad b = \mu_B(x), \ldots; \; a, b, \ldots \in M = [0,1]. \tag{4.3}$$

В нечеткой логике принимаются следующие операции:

– $a \wedge b = \min(a, b)$ – операция *конъюнкции*; (4.4)

– $a \vee b = \max(a, b)$ – операция *дизъюнкции*; (4.5)

– $\overline{a} = 1 - a$ – операция *отрицания*. (4.6)

Определения (4.4) – (4.6) являются ***базовыми***, из них следуют все остальные операции нечеткой логики. Например,

– $a \oplus b = (\overline{a} \wedge b) \vee (a \wedge \overline{b})$ – *дизъюнктивная сумма*;

– $a \mid b = \overline{a \wedge b} = \overline{a} \vee \overline{b}$ –*штрих Шеффера*;

– $a \downarrow b = \overline{a \vee b} = \overline{a} \wedge \overline{b}$ –*стрелка Пирса*;

– $a \sim b = (a \equiv b) = (\overline{a} \wedge \overline{b}) \vee (a \wedge b)$ – *эквивалентность*;

– $a \to b = \overline{a} \vee b$ *–импликация* и т.п.

Основным *критерием адекватности* вводимых операций нечеткой логики является их проверка на булевых переменных обычной логики.

Для операций нечеткой логики – конъюнкции, дизъюнкции и отрицания имеют место следующие *основные свойства*:

– коммутативность $a \wedge b = b \wedge a$, $a \vee b = b \vee a$;

– ассоциативность $(a \wedge b) \wedge c = a \wedge (b \wedge c)$,

$$(a \vee b) \vee c = a \vee (b \vee c);$$

– идемпотентность $a \wedge a = a$, $a \vee a = a$;

– дистрибутивность $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$,

$$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$$

– двойственность (теоремы де Моргана) $\overline{a \vee b} = \overline{a} \wedge \overline{b}$,

$$\overline{a \wedge b} = \overline{a} \vee \overline{b};$$

– двойное отрицание $\overline{(\overline{a})} = a$,

– а также выполняются $a \wedge 0 = 0$, $a \vee 0 = a$, $a \wedge 1 = a$, $a \vee 1 = 1$.

Однако *не выполняются*

– закон противоречия $a \wedge \overline{a} \neq 0$ и (4.7)

– закон исключения третьего $a \vee \overline{a} \neq 1$ при $a \in (0,1)$. (4.8)

Доказательства большинства свойств достаточно тривиальны, за исключением свойств дистрибутивности и двойственности. Подходы к доказательствам свойств дистрибутивности и двойственности аналогичны, они построены на рассмотрении всевозможных соотношений величин $a$, $b$ и $c$.

Докажем одно из свойств. Например, свойство дистрибутивности $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$. Предположим, что значения $a$, $b$ и $c$ могут находиться в отношениях, определяемых следующими тремя различными полными порядками (если, по крайней мере, две из этих величин равны, то доказательство тривиально):

1) $0 \le a \le b \le c \le 1$;
2) $0 \le b \le c \le a \le 1$;
3) $0 \le c \le a \le b \le 1$.

Имеем

1) $a \wedge (b \vee c) = \min\left[a, \max(b, c)\right] = \min(a, c) = a$;

$(a \wedge b) \vee (a \wedge c) = \max\left[\min(a,b), \min(a, c)\right] = \max(a,a) = a$.

2) $a \wedge (b \vee c) = \min\left[a, \max(b, c)\right] = \min(a, c) = c$;

$$(a \wedge b) \vee (a \wedge c) = \max\left[\min(a,b), \min(a, c)\right] = \max(b,c) = c\,.$$

2) $a \wedge (b \vee c) = \min\left[a, \max(b, c)\right] = \min(a, b) = a$;

$$(a \wedge b) \vee (a \wedge c) = \max\left[\min(a,b), \min(a, c)\right] = \max(a,c) = a\,.$$

Докажем теорему де Моргана $\overline{a \vee b} = \overline{a} \wedge \overline{b}$. Пусть $0 \le a < b \le 1$:

$$\max\left[(1-a), (1-b)\right] = 1 - a\,,$$

$$\min\left[a, b\right] = a\,,$$

$$\max\left[(1-a), (1-b)\right] + \min\left[a, b\right] = 1 - a + a = 1.$$

Тогда $\max\left[(1-a), (1-b)\right] = 1 - \min\left[a, b\right]$ или $\overline{a \vee b} = \overline{a} \wedge \overline{b}$.

Из-за того, что имеют место исключения, а именно, $a \wedge \overline{a} \neq 0$ и $a \vee \overline{a} \neq 1$, то структура, определяемая на множестве переменных $a$, $b$,… операциями $\wedge$, $\vee$ и $\overline{\ }$, не может рассматриваться как алгебра в смысле современной математики.

## 4.3. Функции нечетких булевых переменных

Функции, построенные с помощью нечетких булевых переменных, называются *функциями нечетких переменных*. Например, функция $f(a,b,...)$ называется функцией нечетких переменных, если все аргументы $a,b,...$ являются нечеткими и выполнено условие $0 \le f(a,b,...) \le 1$.

В отличие от булевых функций для систематического анализа функций от нечетких аргументов нельзя воспользоваться методом составления таблиц истинности.

Функции нечетких переменных не поддаются упрощению так легко, как булевы функции, поскольку не обладают свойствами (4.7) и (4.8). По этой же причине эти функции нельзя представить в дизъюнктивной нормальной форме (ДНФ) или в конъюнктивной нормальной форме (КНФ).

Иногда определенное число упрощений можно успешно провести, используя только *основные свойства* и следующее *свойство поглощения*:

$$a \vee (a \wedge b) = a$$

и его двойственную форму

$$a \wedge (a \vee b) = a\,.$$

**Пример 4.1**

Функция нечетких булевых переменных легко упрощается

$$f(a,b,c) = (a \wedge b \wedge \overline{c}) \vee \left(\overline{a} \wedge (\overline{b} \vee c)\right) \vee (b \wedge \overline{c}) =$$

$$= \underbrace{(a \wedge b \wedge \overline{c})}_{(1)} \vee \underbrace{(\overline{a} \wedge \overline{b})}_{(2)} \vee \underbrace{(\overline{a} \wedge c)}_{(3)} \vee \underbrace{\overline{a}}_{(4)} \vee \underbrace{(b \wedge \overline{c})}_{(5)} = (b \wedge \overline{c}) \vee \overline{a} \qquad (4.9)$$

согласно свойству поглощения для (1) и (5), а также для (2), (3) и (4).

Здесь уместно отметить важную роль скобок. Известно, что число различных булевых функций при $n$ различных переменных равно $2^{(2^n)}$. В случае $n$ нечетких переменных число нечетких функций, составленных произвольным образом из этих $n$ переменных и операций $\wedge$, $\vee$ и $\overline{\ }$, также конечно.

*Замечание*. Операцию $\vee$ можно выразить через операцию $\wedge$ и операцию отрицания $\overline{\ }$ и наоборот. Действительно, например, законы де Моргана можно представить и другим способом:

$$a \wedge b = \min(a,b) = 1 - \max(\overline{a},\overline{b}) = \overline{\overline{a} \vee \overline{b}}\,, \qquad (4.10)$$

$$a \vee b = \max(a,b) = 1 - \min(\overline{a},\overline{b}) = \overline{\overline{a} \wedge \overline{b}}\,. \qquad (4.11)$$

Таким образом, достаточно использовать операторы $\wedge$ и $\overline{\ }$ или операторы $\vee$ и $\overline{\ }$ для того, чтобы представить любую функцию нечетких переменных, содержащую символы $\wedge$, $\vee$ и $\overline{\ }$, хотя выражение становится более громоздким.

Следует напомнить, что в обычной булевой алгебре для того, чтобы представить произвольную булеву функцию, достаточно одного оператора.

Для изучения булевых бинарных функций используется так называемая таблица истинности, в которой бинарным переменным придаются все возможные значения и выписываются соответствующие значения функции [8]. Для функций нечетких переменных вместо таблицы истинности строится таблица другого типа, которая играет аналогичную роль и называется *таблицей значений*. При этом рассматриваются все возможные соотношения между нечеткими переменными.

Чтобы изучить функцию *одной нечеткой переменной* $a$ необходимо рассмотреть значения нечеткой функции в следующих двух случаях:

$$a \le \overline{a}\,, \quad \overline{a} \le a\,.$$

Для изучения функции двух переменных $a$ и $b$ рассматривается ее значение в следующих восьми случаях:

$$a \le b \le \overline{b} \le \overline{a}\,, \quad a \le \overline{b} \le b \le \overline{a}\,, \quad \overline{a} \le b \le \overline{b} \le a\,, \quad \overline{a} \le \overline{b} \le b \le a\,,$$

$b \le a \le \overline{a} \le \overline{b}$ , $b \le \overline{a} \le a \le \overline{b}$ , $\overline{b} \le a \le \overline{a} \le b$ , $\overline{b} \le \overline{a} \le a \le b$ .

Чтобы изучить функцию трех переменных $a$ , $b$ и $c$, необходимо рассмотреть 48 случаев соотношения переменных.

Для изучения функции $n$ переменных рассматривается

$P_n \cdot 2^n$ случаев, где $P_n = n \cdot (n-1) \cdot ... \cdot 3 \cdot 2 \cdot 1$ .

Во всех соотношениях нечетких переменных устанавливается эффект антисимметрии, возникающий из-за того, что если $x \le y$ , то $\overline{y} \le \overline{x}$ .

**Пример 4.2**

Перечислить значения функции

$$f(a,b) = \left(a \wedge \overline{a}\right) \vee \left(\overline{a} \wedge b \wedge \overline{b}\right) .$$

Результат представлен в следующей таблице.

| | $a \wedge \overline{a}$ | $\overline{a} \wedge b \wedge \overline{b}$ | $f(a,b) = \left(a \wedge \overline{a}\right) \vee \left(\overline{a} \wedge b \wedge \overline{b}\right)$ |
|---|---|---|---|
| $a \le b \le \overline{b} \le \overline{a}$ | $a$ | $b$ | $b$ |
| $a \le \overline{b} \le b \le \overline{a}$ | $a$ | $\overline{b}$ | $\overline{b}$ |
| $\overline{a} \le b \le \overline{b} \le a$ | $\overline{a}$ | $\overline{a}$ | $\overline{a}$ |
| $\overline{a} \le \overline{b} \le b \le a$ | $\overline{a}$ | $\overline{a}$ | $\overline{a}$ |
| $b \le a \le \overline{a} \le \overline{b}$ | $a$ | $b$ | $a$ |
| $b \le \overline{a} \le a \le \overline{b}$ | $\overline{a}$ | $b$ | $\overline{a}$ |
| $\overline{b} \le a \le \overline{a} \le b$ | $a \wedge$ | $\overline{b}$ | $a$ |
| $\overline{b} \le \overline{a} \le a \le b$ | $\overline{a}$ | $\overline{b}$ | $\overline{a}$ |

***Определение.*** Две функции $f_1$ и $f_2$ нечетких переменных считаются *равносильными* (или *тождественными*), если они имеют одну и ту же таблицу значений, включающую все возможные случаи.

***Определение.*** Нечеткие функции, в которых нечеткие переменные $a,b,... \in [0,1]$ подвергаются другим, отличным от $\wedge$ , $\vee$ и $\overline{\ }$, операциям называются *смешанными функциями нечетких переменных*.

В число таких операций входят, например, умножение $a \cdot b$ и суммирование $a \hat{+} b = a + b - a \cdot b$ , для которых, как легко проверить, выполняется свойство $a,b \in [0,1] \to a \cdot b \in [0,1]$ и $a,b \in [0,1] \to a \hat{+} b \in [0,1]$ соответственно. Например, выражение

$f(a,b,c) = (a \hat{+} b) \vee (\overline{b} \hat{+} c) \wedge a \wedge c$ – смешанная нечеткая функция.

Принято называть *аналитической функцией нечетких переменных* функцию, которую можно выразить, используя только операции $\wedge$, $\vee$ и $\bar{}$. Так как в основном приходится иметь дело с аналитическими функциями нечетких переменных, то для упрощения изложения (если это не вызывает ошибок) аналитические функции нечетких переменных называют просто функциями нечетких переменных.

С помощью двойственных *законов дистрибутивности* любую функцию $f(a,b,...)$ можно представить в *полиномиальной* форме относительно операции $\wedge$ или $\vee$.

Рассмотрим пример. Пусть

$$f(a,b,c) = (\bar{a} \wedge \bar{b}) \vee (a \wedge b \wedge \bar{c}) .$$

Эта функция записана в полиномиальной форме относительно $\vee$. Используя свойства ассоциативности и дистрибутивности функцию можно преобразовать в полиномиальную форму относительно $\wedge$:

$$f(a,b,c) = (\bar{a} \vee a) \wedge (\bar{a} \vee b) \wedge (\bar{a} \vee c) \wedge (\bar{b} \vee a) \wedge (\bar{b} \vee b) \wedge (\bar{b} \wedge c) .$$

***Определение.*** Одночлен полиномиальной формы $f(a,b,...)$ относительно $\vee$ (или относительно $\wedge$) называется *максимальным*, если он не поглощается никаким другим одночленом этой полиномиальной формы.

Например, в следующей функции третий член не является максимальным, т.к. он поглощается вторым

$$f(a,b,c) = (a \vee \bar{b}) \wedge c \wedge (\bar{a} \vee b \vee c) ,$$

поэтому эту функцию можно выразить через максимальные одночлены относительно $\wedge$:

$$f(a,b,c) = (a \vee \bar{b}) \wedge c$$

или, используя свойства ассоциативности и дистрибутивности, построить полиномиальную форму относительно $\vee$, состоящую из максимальных одночленов.

***Определение.*** Всякая полиномиальная форма относительно $\vee$, состоящая только из максимальных одночленов по $\wedge$, называется *приведенной полиномиальной формой относительно* $\vee$.

Замена в определении операции $\wedge$ на операцию $\vee$ и наоборот приводит к определению *приведенной полиномиальной формы относительно* $\wedge$.

Аналитической функции $f(a,b,...)$ могут соответствовать несколько приведенных полиномиальных форм. Например, следующие две приведенные полиномиальные формы:

$$f(a,b) = (a \wedge \bar{a}) \vee (a \wedge b) \vee (a \wedge b) ,$$

$$f(a,b) = (a \wedge b) \vee (a \wedge \overline{b})$$

соответствуют одной и той же аналитической функции, что можно проверить, построив таблицы значений с полным перечислением.

Для любой аналитической функции существует, по крайней мере, одна приведенная полиномиальная форма относительно $\vee$ и, по крайней мере, одна приведенная полиномиальная форма относительно $\wedge$. Можно переходить от одной из них к другой разложением по $\wedge$ (соответственно по $\vee$) с последующим сокращением не максимальных одночленов.

Достаточное условие *тождественности* двух функций нечетких переменных состоит в том, чтобы их можно было привести к одной и той же приведенной полиномиальной форме относительно $\vee$ (соответственно относительно $\wedge$). Необходимое и достаточное условие состоит в том, чтобы у этих функций была одна и та же таблица значений.

## 4.4. Анализ функций нечетких булевых переменных

Пусть каждая нечеткая переменная функции $f(a_1, a_2, ..., a_n)$ принадлежит некоторому интервалу $a_i \in I_i \subset M$, $i = 1, 2, ..., n$, где $M$ – дискретная функция принадлежности. Необходимо найти область определения $I$ функция $f(a_1, a_2, ..., a_n)$, т.е. определить какому интервалу принадлежит эта функция.

Для анализа *функций дискретных переменных* удобно использовать таблицы значений, играющие роль таблиц истинности при анализе функций булевых переменных.

Рассмотрим, как применяются эти таблицы.

**Пример 4.3.**

Пусть

$$f(a,b,c) = (a \wedge \overline{b} \wedge c) \vee \overline{c},$$

где $a, b, c \in I = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$, $a \in \{0, 0.2, 0.4\}$, $b \in \{0.2, 0.4, 0.8\}$, $c \in \{0, 0.2, 1\}$.

Найти область определения функции $f(a,b,c)$.

Сначала найдем область определения $d = a \wedge \overline{b}$ с помощью таблицы (рис. 4.1), где $\overline{b} \in \{0.2, 0.6, 0.8\}$.

| $a \wedge \overline{b}$ | 1 | 0.8 | 0.6 | 0.4 | 0.2 | 0 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0 |
| 0.4 | 0.4 | 0.4 | 0.4 | 0.4 | 0.2 | 0 |
| 0.6 | 0.6 | 0.6 | 0.6 | 0.4 | 0.2 | 0 |
| 0.8 | 0.8 | 0.8 | 0.6 | 0.4 | 0.2 | 0 |
| 1 | 1 | 0.8 | 0.6 | 0.4 | 0.2 | 0 |

($a$ — строки, $\overline{b}$ — столбцы)

$$d = a \wedge \overline{b} = \{0,\ 0.2,\ 0.4\}$$

Рис. 4.1.

| $d \wedge c$ | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.2 | 0 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 |
| 0.4 | 0 | 0.2 | 0.4 | 0.4 | 0.4 | 0.4 |
| 0.6 | 0 | 0.2 | 0.4 | 0.6 | 0.6 | 0.6 |
| 0.8 | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 0.8 |
| 1 | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 |

($d$ — строки, $c$ — столбцы)

$$e = (a \wedge \overline{b}) \wedge c =$$
$$= d \wedge c = \{0,\ 0.2,\ 0.4\}$$

Рис. 4.2.

| $e \vee \overline{c}$ | 1 | 0.8 | 0.6 | 0.4 | 0.2 | 0 |
|---|---|---|---|---|---|---|
| 0 | 1 | 0.8 | 0.6 | 0.4 | 0.2 | 0 |
| 0.2 | 1 | 0.8 | 0.6 | 0.4 | 0.2 | 0.2 |
| 0.4 | 1 | 0.8 | 0.6 | 0.4 | 0.4 | 0.4 |
| 0.6 | 1 | 0.8 | 0.6 | 0.6 | 0.6 | 0.6 |
| 0.8 | 1 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |

($e$ — строки, $\overline{c}$ — столбцы)

$$f(a,b,c) = (a \wedge \overline{b} \wedge c) \vee \overline{c} =$$
$$= e \vee \overline{c} = \{0,\ 0.2,\ 0.4,\ 0.8,\ 1\}$$

Рис. 4.3.

Наконец, вычисляем область определения функции

$$f(a,b,c) = (a \wedge \overline{b} \wedge c) \vee \overline{c}$$

с помощью таблицы на рис. 4.3:

$$f(a,b,c) = (a \wedge \overline{b} \wedge c) \vee \overline{c} = \{0,\ 0.2,\ 0.4\} \cup \{0.8,\ 1\} = \{0,\ 0.2,\ 0.4,\ 0.8,\ 1\}.$$

Рассмотрим подход к анализу нечеткой функции непрерывных нечетких переменных. Такая задача часто называется *задачей композиции интервалов*.

Пусть $a \in D_a = [a_1, a_2[$ и $b \in D_b = [b_1, b_2[$. Тогда какому интервалу $D_{a \wedge b}$ принадлежит функция $f(a,b) = a \wedge b$ ? Легко определить, что

$$f(a,b) = a \wedge b \in D_{a \wedge b} = [a_1 \wedge b_1,\ a_2 \wedge b_2[ .$$

Аналогично можно определить интервал, которому принадлежит функция $f(a,b) = a \vee b$ :

$$f(a,b) = a \vee b \in D_{a \vee b} = [a_1 \vee b_1,\ a_2 \vee b_2[ .$$

**Пример 4.4.**

Пусть $a \in D_a = [0.4, 0.9[$ и $b \in D_b = [0.3, 0.7[$.

Очевидно, что

$$a \wedge b \in D_{a \wedge b} = [0.4 \wedge 0.3,\ 0.9 \wedge 0.7[ = [0.3,\ 0.7[ .$$

$$a \vee b \in D_{a \vee b} = [0.4 \vee 0.3,\ 0.9 \vee 0.7[ = [0.4,\ 0.9[ .$$

Используя свойства операций (например, ассоциативности, дистрибутивности и т.п.) над нечеткими переменными, можно выполнить анализ более сложных функций с учетом соотношений конкретными значениями интервалов.

Если нечеткие переменные принимают свои значения в дополнениях к своим интервалам, т.е.

$$a \in \bar{D}_a = [0, a_1[ \cup [a_2, 1] \quad \text{и} \quad b \in \bar{D}_b = [0, b_1[ \cup [b_2, 1],$$

то получим следующие результаты:

для $f(a,b) = a \wedge b$ , $a \in \bar{D}_a$ и $b \in D_b$ имеем

$$f(a,b) = a \wedge b \in D_{a \wedge b} = [0, a_1 \wedge b_2[ \cup [a_2 \wedge b_1, b_2[$$

В ниже следующей таблице приведены основные случаи для операций $\wedge$ и $\vee$, когда $D_a = [a_1, a_2[$, $D_b = [b_1, b_2[$. С помощью приведенных в таблице выражений можно получать области определения более сложных функций $f(a,b,...)$.

| $f(a,b)$ | Область определения $a$ | Область определения $b$ | Область определения $f(a,b)$ |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| $a \wedge b$ | $D_a$ | $D_b$ | $[a_1 \wedge b_1,\ a_2 \wedge b_2[$ |
| | $\bar{D}_a$ | $D_b$ | $[0,\ a_1 \wedge b_2[ \cup [a_2 \wedge b_1,\ b_2[$ |
| | $D_a$ | $\bar{D}_b$ | $[0,\ b_1 \wedge a_2[ \cup [b_2 \wedge a_1,\ a_2[$ |
| | $\bar{D}_a$ | $\bar{D}_b$ | $[0,\ a_1 \wedge b_1[ \cup [a_2 \wedge b_2,\ 1]$ |

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| $a \vee b$ | $D_a$ | $D_b$ | $[a_1 \vee b_1,\ a_2 \vee b_2[$ |
| | $\bar{D}_a$ | $D_b$ | $[b_1,\ a_1 \vee b_2[ \cup [a_2 \vee b_1,\ 1]$ |
| | $D_a$ | $\bar{D}_b$ | $[a_1,\ b_1 \vee a_2[ \cup [b_2 \vee a_1,\ 1]$ |
| | $\bar{D}_a$ | $\bar{D}_b$ | $[0,\ a_1 \wedge b_1[ \cup [a_2 \wedge b_2,\ 1]$ |

Не следует путать определения $\bar{D}_a$, $D_{\bar{a}}$ и $\bar{D}_{\bar{a}}$, т.к.

$$\bar{D}_a = [0, a_1[ \cup [a_2, 1], \quad D_{\bar{a}} = [1 - a_2, 1 - a_1] \quad \text{и}$$

$$\bar{D}_{\bar{a}} = [0, 1 - a_2] \cup ]1 - a_1, 1].$$

Анализ *функций непрерывных переменных* может выполняться на интервале $M = [0, 1]$, разбитом на $m$ попарно граничащих интервалов, замкнутых слева и открытых справа, за исключением последнего интервала:

$$I_1 = [\alpha_0 = 0,\ \alpha_1[, \quad I_2 = [\alpha_1,\ \alpha_2[, \ \ldots, \quad I_m = [\alpha_{m-1},\ \alpha_m = 1],$$

где

$$M = I_1 \cup I_2 \cup \ldots \cup I_m = \left([\alpha_0 = 0,\ \alpha_1[\right) \cup \left([\alpha_1,\ \alpha_2[\right) \cup \ldots \cup \left([\alpha_{m-1},\ \alpha_m = 1]\right).$$

Необходимо найти условия, при которых функция $n$ нечетких переменных

$$f(a_1, a_2, \ldots, a_n), \quad a_i \in [0, 1], \quad i = 1, 2, \ldots n$$

будет принадлежать интервалу некоторому $I_k$. Анализ строится на гипотезах о соотношении между переменными.

**Пример 4.5.**

Пусть

$$f(a, b, c) = (a \wedge \bar{b}) \vee (\bar{a} \wedge c) \vee \bar{c},$$

предположим, что интервал $[0, 1]$ разделен на три интервала:

$$I_1 = [0,\ 0.2[, \quad I_2 = [0.2,\ 0.3[, \quad I_3 = [0.3,\ 1].$$

Вычислить при каких значениях переменных функция определена в том или ином интервале.

*Решение*.

Рассмотрим интервал $I_1 = [0,\ 0.2[$.

*Гипотеза 1*: $a \wedge \bar{b} > \bar{a} \wedge c$, $a \wedge \bar{b} > \bar{c}$.

Тогда имеем $0 \le a \wedge \bar{b} < 0.2$. Таким образом, $0 \le \min(a, 1 - b) < 0.2$,

$a \geq 0$ и $b \leq 1$, *и* $a < 0.2$ или/и $b > 0.8$.

*Гипотеза 2*: $\overline{a} \wedge c > a \wedge \overline{b}$, $\overline{a} \wedge c > \overline{c}$.

Тогда имеем $0 \leq \overline{a} \wedge c < 0.2$. Таким образом, $0 \leq \min(1-a, c) < 0.2$,

$a \leq 1$ и $c \geq 0$, *и* $a > 0.8$ или/и $c < 0.2$.

*Гипотеза 3*: $\overline{c} > a \wedge \overline{b}$, $\overline{c} > \overline{a} \wedge c$.

Тогда имеем $0 \leq \overline{c} < 0.2$. Таким образом, $0 \leq \overline{c} < 0.2$,

$0 \leq 1-c < 0.2$ или $0.8 < c \leq 1$.

Рассмотрим интервал $I_2 = [0.2, 0.3[$.

*Гипотеза 1*: $a \wedge \overline{b} > \overline{a} \wedge c$, $a \wedge \overline{b} > \overline{c}$.

$0.2 \leq a \wedge \overline{b} < 0.3$, $0.2 \leq \min(a, 1-b) < 0.3$,

$a \geq 0.2$ и $b \leq 0.8$, *и* $a < 0.3$ или/и $b > 0.7$.

*Гипотеза 2*: $\overline{a} \wedge c > a \wedge \overline{b}$, $\overline{a} \wedge c > \overline{c}$.

$0.2 \leq \overline{a} \wedge c < 0.3$, $0.2 \leq \min(1-a, c) < 0.3$,

$a \leq 0.8$ и $c \geq 0.2$, *и* $a > 0.7$ и $c < 0.3$.

*Гипотеза 3*: $\overline{c} > a \wedge \overline{b}$, $\overline{c} > \overline{a} \wedge c$.

$0.2 \leq \overline{c} < 0.3$ *и* $c \leq 0.8$ и $c > 0.7$.

Рассмотрим интервал $I_2 = [0.3, 1]$.

*Гипотеза 1*: $a \wedge \overline{b} > \overline{a} \wedge c$, $a \wedge \overline{b} > \overline{c}$.

$0.3 \leq a \wedge \overline{b} \leq 1$, $0.3 \leq \min(a, 1-b) \leq 1$,

$a \geq 0.3$ и $b \leq 0.7$, *и* $a \leq 1$ или/и $b \geq 0$.

*Гипотеза 2*: $\overline{a} \wedge c > a \wedge \overline{b}$, $\overline{a} \wedge c > \overline{c}$.

$0.3 \leq \overline{a} \wedge c \leq 1$, $0.3 \leq \min(1-a, c) <\leq 1$,

$a \leq 0.7$ и $c \geq 0.3$, *и* $a \geq 0$ или (и) $c \leq 1$.

*Гипотеза 3*: $\overline{c} > a \wedge \overline{b}$, $\overline{c} > \overline{a} \wedge c$.

$0.3 \leq \overline{c} \leq 1$ *и* $c \leq 0.7$ и $c \geq 0$.

Результаты анализа можно перегруппировать следующим образом:

а) $0 \leq f(a,b,c) < 0.2$.

Свойство $P_1^{(1)}$:

[( $a \geq 0$ ) и ( $b \leq 1$ )] или/и [( $a \leq 1$ ) и ( $c \geq 0$ )] или/и ( $c \leq 1$ ).

Свойство $P_2^{(1)}$:

[( $a < 0.2$ ) или/и ( $b > 0.8$ )] и [( $a > 0.8$ ) и ( $c < 0.2$ )] и ( $c > 0.8$ ).

Если соотношения $P_1^{(1)}$ и $P_2^{(1)}$ выполняются, то функция $f(a,b,c)$ определена на интервале $I_1 = [0, 0.2[$.

б) $0.2 \le f(a,b,c) < 0.3$.

Свойство $P_1^{(2)}$:

[( $a \ge 0.2$ ) и ( $b \le 0.8$ )] или/и [( $a \le 0.8$ ) и ( $c \ge 0.2$ )] или/и ( $c \le 0.8$ ).

Свойство $P_2^{(2)}$:

[( $a < 0.3$ ) или/и ( $b > 0.7$ )] и [( $a > 0.7$ ) и ( $c < 0.3$ )] и ( $c > 0.7$ ).

Если соотношения $P_1^{(2)}$ и $P_2^{(2)}$ выполняются, то функция $f(a,b,c)$ определена на интервале $I_2 = [0.2,\ 0.3[$.

б) $0.3 \le f(a,b,c) \le 1$.

Свойство $P_1^{(3)}$:

[( $a \ge 0.3$ ) и ( $b \le 0.7$ )] или/и [( $a \le 0.7$ ) и ( $c \ge 0.3$ )] или/и ( $c \le 0.7$ ).

Свойство $P_2^{(3)}$:

[( $a \le 1$ ) или/и ( $b \ge 0$ )] и [( $a \ge 0$ ) и ( $c \le 1$ )] и ( $c \ge 0$ ).

Если соотношения $P_1^{(3)}$ и $P_2^{(3)}$ выполняются, то функция $f(a,b,c)$ определена на интервале $I_3 = [0.3,\ 1]$.

При конкретных числовых значениях переменных можно получить конкретное значение функции. Предположим, что

$$a = 0.55\ ,\quad b = 0.67\ ,\quad c = 0,42\ .$$

Тогда имеем

$$f(a,b,c) = f(0.55,\ 0.67,\ 0.42) = (a \wedge \overline{b}) \vee (\overline{a} \wedge c) \vee \overline{c} =$$
$$= (0.55 \wedge 0.33) \vee (0.45 \wedge 0.42) \vee 0.58 =$$
$$= 0.33 \vee 0.42 \vee 0.58 = 0.58\ .$$

При этом выполняются свойства $P_1^{(3)}$ и $P_2^{(3)}$.

# Глава 5

# ОСНОВЫ ТЕОРИИ НЕЙРОННЫХ СЕТЕЙ

## 5.1. Основные положения теории нейронных сетей

Под нейронными сетями (НС) подразумеваются вычислительные структуры, которые моделируют простые биологические процессы, обычно ассоциируемые с процессами человеческого мозга. Адаптируемые и обучаемые, они представляют собой распараллеленные системы, способные к обучению путем анализа положительных и отрицательных воздействий. Элементарным преобразователем в данных сетях является *искусственный нейрон* или просто *нейрон*, названный так по аналогии с биологическим прототипом.

К настоящему времени предложено и изучено большое количество моделей нейроподобных элементов и нейронных сетей, ряд из которых рассмотрен в настоящей главе.

Термин «нейронные сети» сформировался в 40-х годах XX века в среде исследователей, изучавших принципы организации и функционирования биологических нейронных сетей. Основные результаты, полученные в этой области, связаны с именами американских исследователей У.Маккалоха, Д.Хебба, Ф.Розенблатта, М. Минского, Дж. Хопфилда и др.

Представим некоторые проблемы, решаемые с помощью нейронных сетей.

*Классификация образов*. Задача состоит в указании принадлежности входного образа (например, речевого сигнала или рукописного символа), представленного вектором признаков, одному или нескольким предварительно определенным классам. К известным приложениям относятся распознавание букв, распознавание речи, классификация сигнала электрокардиограммы, классификация клеток крови.

*Кластеризация (категоризация)*. При решении задачи кластеризации, которая известна также как классификация образов «без учителя», отсутствует обучающая выборка с метками классов. Алгоритм кластеризации основан на подобии образов и размещает близкие образы в один кластер. Известны случаи применения кластеризации для извлечения знаний, сжатия данных и исследования свойств данных.

*Аппроксимация функций.* Предположим, что имеется обучающая выборка $\big((x_1, y_1), (x_2, y_2),..., (x_n, y_n)\big)$ (пары данных вход-выход), которая генерируется неизвестной функцией $F(x)$, искаженной шумом. Задача аппроксимации состоит в нахождении оценки неизвестной функции $F(x)$. Аппроксимация функций необходима при решении многочисленных инженерных и научных задач моделирования.

*Предсказание (прогноз).* Пусть заданы $n$ дискретных отсчетов $\{y(t_1),\ y(t_2),...,\ y(t_k)\}$ в последовательные моменты времени $t_1, t_2,...,t_k$. Задача состоит в предсказании значения $y(t_{k+1})$ в некоторый будущий момент времени $t_{k+1}$. Предсказание (прогноз) имеет значительное влияние на принятие решений в бизнесе, науке и технике. Предсказание цен на фондовой бирже и прогноз погоды являются типичными приложениями техники предсказания (прогноза).

*Оптимизация.* Многочисленные проблемы в математике, статистике, технике, науке, медицине и экономике могут рассматриваться как проблемы оптимизации. Задачей алгоритма оптимизации является нахождение такого решения, которое удовлетворяет системе ограничений и максимизирует или минимизирует целевую функцию. Известная задача коммивояжера является классическим примером задачи оптимизации.

*Память, адресуемая по содержанию.* В модели вычислений фон Неймана обращение к памяти доступно только посредством адреса, который не зависит от содержания памяти. Более того, если допущена ошибка в вычислении адреса, то может быть найдена совершенно иная информация. Ассоциативная память, или память, адресуемая по содержанию, доступна по указанию заданного содержания. Содержимое памяти может быть вызвано даже по частичному входу или искаженному содержанию. Ассоциативная память чрезвычайно желательна при создании мультимедийных информационных баз данных.

*Управление.* Рассмотрим динамическую систему, заданную совокупностью $\{u(t), y(t)\}$, где $u(t)$ является входным управляющим воздействием, а $y(t)$ – выходом системы в момент времени $t$. В системах управления с эталонной моделью целью управления является расчет такого входного воздействия $u(t)$, при котором система следует по желаемой траектории, диктуемой эталонной моделью. Примером является оптимальное управление двигателем.

## 5.2. Структура и свойства искусственного нейрона

Искусственный нейрон построен по образу и подобию биологического нейрона, т.е. искусственный нейрон представляет собой электронную модель биологического нейрона.

*Биологический нейрон*. Нервная система и мозг человека состоят из нейронов, соединенных между собой нервными волокнами. Нервные волокна способны передавать электрические импульсы между нейронами. Все процессы передачи раздражений от нашей кожи, ушей и глаз к мозгу, процессы мышления и управления действиями – все это реализовано в живом организме как передача электрических импульсов между нейронами.

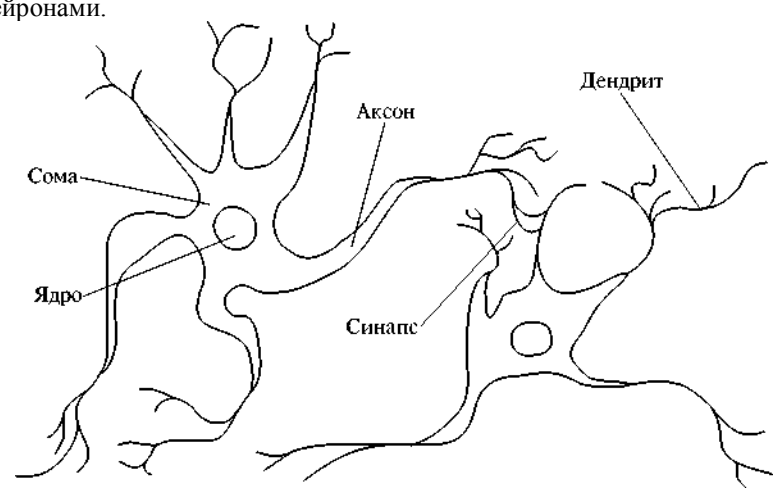



Рис. 5.1. Взаимосвязь биологических нейронов

*Нейрон* (нервная клетка) является особой биологической клеткой, которая обрабатывает информацию (рис. 5.1). Он состоит из тела и отро-

стков нервных волокон двух типов – *дендритов*, по которым принимаются импульсы, и единственного *аксона*, по которому нейрон может передавать импульс. Тело нейрона включает *ядро*, которое содержит информацию о наследственных свойствах, и *плазму*, обладающую молекулярными средствами для производства необходимых нейрону материалов. Нейрон получает сигналы (импульсы) от аксонов других нейронов через дендриты (приемники) и передает сигналы, сгенерированные телом клетки, вдоль своего аксона (передатчик), который в конце разветвляется на волокна. На окончаниях этих волокон находятся специальные образования – *синапсы*, которые влияют на силу импульса.

*Синапс* является элементарной структурой и функциональным узлом между двумя нейронами (волокно аксона одного нейрона и дендрит другого). Когда импульс достигает синаптического окончания, высвобождаются определенные химические вещества, называемые *нейротрансмиттерами*. Нейротрансмиттеры диффундируют через *синаптическую* щель, возбуждая или затормаживая, в зависимости от типа синапса, способность нейрона-приемника генерировать электрические импульсы. Результативность синапса может настраиваться проходящими через него сигналами, так что синапсы могут обучаться в зависимости от активности процессов, в которых они участвуют. Эта зависимость от предыстории действует как память, которая, возможно, ответственна за память человека. Важно отметить, что веса синапсов могут изменяться со временем, что изменяет и поведение соответствующего нейрона.

*Искусственный нейрон* (или просто *нейрон*) – это составная часть нейронной сети. На рис. 5.2 показана его структура.

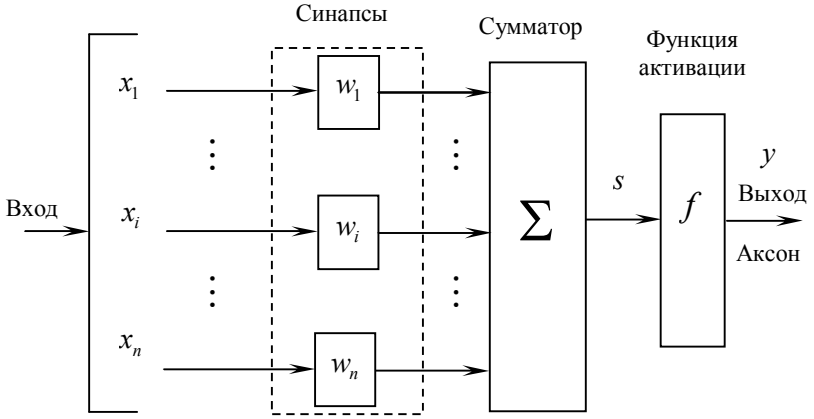



Рис.5.2. Модель (структура) искусственного нейрона.

В состав нейрона входят *умножители (синапсы)*, *сумматор* и *нелинейный преобразователь*. Синапсы осуществляют связь между нейронами и умножают входной сигнал на число, характеризующие силу связи. – вес синапса. Сумматор выполняет сложение сигналов, поступающих по синаптическим связям от других нейронов, и внешних входных сигналов. Нелинейный преобразователь реализует нелинейную функцию одного аргумента – выхода сумматора. Эта функция называется «функция активизации» или «передаточная функция» нейрона. Нейрон в целом реализует скалярную функцию векторного аргумента. Математическая модель нейрона описывается соотношением

$$s = \sum_{i=1}^{n} w_i x_i + b\,,$$

где $w_i$ – вес синапса, $i = 1, 2, ...n$; $b$ – значение смещения; $s$ – результат суммирования; $x_i$ – компонента входного вектора (входного сигнала), $i = 1, 2, ...n$; $y$ – выходной сигнал нейрона; $n$ – число входов нейрона; $f$ – нелинейное преобразование (функция активации или передаточная функция).

**Таблица 5.1.** Перечень функций активации нейронов.

| Название | Формула | Область значений |
|---|---|---|
| 1 | 2 | 3 |
| Пороговая | $f(s) = \begin{cases} 0, & s < \theta \\ 1, & s \geq \theta \end{cases}$ | $[0, 1]$ |
| Знаковая (сигнатурная) | $f(s) = \begin{cases} 0, & s < \theta \\ 1, & s \geq \theta \end{cases}$ | $[-1, 1]$ |
| Сигмоидальная (логистическая) | $f(s) = \dfrac{1}{1 + c^{-s}}$ | $(0, 1)$ |
| Полулинейная | $f(s) = \begin{cases} s, & s > 0 \\ 1, & s \leq 0 \end{cases}$ | $(0, \infty)$ |
| Линейная | $f(s) = s$ | $(-\infty, \infty)$ |
| Радиальная базисная (гауссова) | $f(s) = c^{-s^2}$ | $(0, 1)$ |

**Таблица 5.1.** (продолжение)

| 1 | 2 | 3 |
|---|---|---|
| Полулинейная с насыщением | $f(s)=\begin{cases}0, & s\le 0\\ s, & -1<s<1,\\ 1, & s\ge 1\end{cases}$ | $(0, 1)$ |
| Линейная с насыщением | $f(s)=\begin{cases}-1, & s\le -1\\ s, & -1<s<1,\\ 1, & s\ge 1\end{cases}$ | $(-1, 1)$ |
| Гиперболический тангенс (сигмоидальная) | $f(s)=\dfrac{e^{s}-e^{-s}}{e^{s}+e^{-s}}$ | $(-1, 1)$ |
| Треугольная | $f(s)=\begin{cases}0, & s<\theta\\ 1, & s\ge\theta\end{cases}$ | $(0, 1)$ |

В общем случае входной сигнал, весовые коэффициенты и значения смещения могут принимать действительные значения. Выход $(y)$ определяется видом функции активации и может быть как действительным, так и целым. Во многих практических задачах входы, веса и смещения могут принимать лишь некоторые фиксированные значения.

Синаптические связи с положительными весами называют *возбуждающими*, с отрицательными весами – *тормозящими*.

Таким образом, нейрон полностью описывается своими весами $w_i$ и передаточной функцией $f(x)$. Получив набор чисел (вектор) $x_i$ в качестве входов, нейрон выдает некоторое число $y$ на выходе.

Описанный вычислительный элемент можно считать упрощенной математической моделью биологических нейронов – клеток, из которых состоит нервная система человека и животных.

Чтобы подчеркнуть различие нейронов биологических и математических, вторые иногда называют *нейроноподобными* элементами или *формальными нейронами*.

На входной сигнал $(s)$ нелинейный преобразователь отвечает выходным сигналом $f(s)$, который представляет собой выход нейрона $y$. Примеры активационных функций представлены в таблице 5.1 функций активации нейронов и на рис. 5.3.

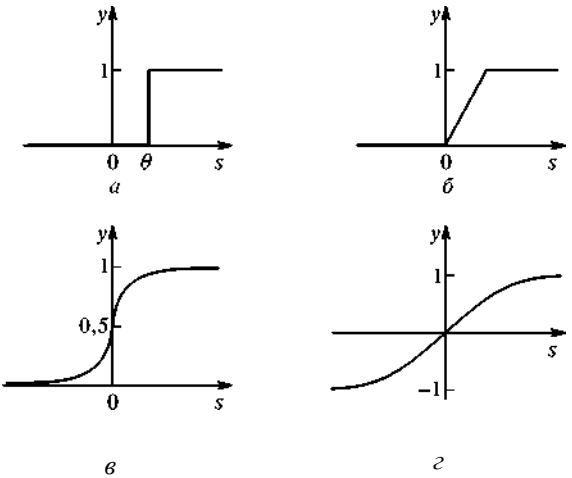



Рис. 5.3. Примеры активационных функций:
*а* — функция единичного скачка; *б*— линейный порог (гистерезис);
*в* — сигмоид (гиперболическийтангенс); *г* — сигмоид (логистическая)

Одной из наиболее распространенных является нелинейная функция с насыщением, так называемая логистическая функция или *сигмоид* (т.е. функция $S$ – образного вида)

$$f(s)=\frac{1}{1+e^{-as}}.$$

При уменьшении $a$ сигмоид становится более пологим, в пределе при $a=0$ вырождаясь в горизонтальную линию на уровне $0.5$ . При увеличении $a$ сигмоид приближается по внешнему виду к функции единичного скачка с порогом $\theta$ в точке $s=0$. Из выражения для сигмоида очевидно, что выходное значение нейрона лежит в диапазоне $[0,1]$. Одно из ценных свойств сигмоидной функции – простое выражение для ее производной, применение которого будет рассмотрено в дальнейшем:

$$f'(s)=a\,f(s)\big(1-f(s)\big).$$

Следует отметить, что сигмоидная функция дифференцируема на всей оси абсцисс, что используется в некоторых алгоритмах обучения. Кроме того она обладает свойством усиливать слабые сигналы лучше, чем большие, и предотвращает насыщение от больших сигналов, так как они соответствуют областям аргументов, где сигмоид имеет пологий наклон.

Возвращаясь к общим чертам, присущим всем нейронным сетям, отметим принцип параллельной обработки сигналов, который достигается путем объединения большого числа нейронов в так называемые слои и соединения определенным образом нейронов различных слоев, а также, в некоторых конфигурациях, и нейронов одного слоя между собой, причем обработка взаимодействия всех нейронов ведется послойно.

## 5.3. Классификация нейронных сетей и их свойства

Как отмечалось, искусственная нейронная сеть (нейросеть) – это набор нейронов, соединенных между собой. Как правило, передаточные (активационные) функции всех нейронов в сети фиксированы, а веса являются параметрами сети и могут изменяться. Некоторые входы нейронов помечены как внешние входы сети, а некоторые выходы – как внешние выходы сети. Подавая любые числа на входы сети, мы получаем какой-то набор чисел на выходах сети. Таким образом, работа нейросети состоит в преобразовании входного вектора $X$ в выходной вектор $Y$, причем это преобразование задается весами сети.

Практически любую задачу можно свести к задаче, решаемой нейросетью. В табл. 5.2 показано, каким образом следует сформулировать в терминах нейросети задачу распознавания рукописных букв.

Поясним, зачем требуется выбирать выход с максимальным уровнем сигнала. Дело в том, что уровень выходного сигнала, как правило, может принимать любые значения из какого-то отрезка.

Однако в данной задаче нас интересует не аналоговый ответ, а всего лишь номер категории (номер буквы в алфавите). Поэтому используется следующий подход – каждой категории сопоставляется свой выход, а ответом сети считается та категория, на чьем выходе уровень сигнала максимален. В определенном смысле уровень сигнала на выходе «А» – это достоверность того, что на вход была подана рукописная буква «А». Задачи, в которых нужно отнести входные данные к одной из известных категорий, называются задачами *классификации*. Изложенный подход – стандартный способ классификации с помощью нейронных сетей.

**Таблица 5.2**

Задача распознавания рукописных букв в терминах нейросети.

| Задача распознавания рукописных букв | |
|---|---|
| Дано: растровое черно-белое изображение буквы размером $30 \times 30$ пикселов | Надо: определить, какая это буква (в алфавите 33 буквы) |

**Таблица 5.2.** (продолжение)

| Формулировка задачи для нейросети | |
|---|---|
| Дано: входной вектор из 900 двоичных символов $(900 = 30 \times 30)$ | Надо: построить нейросеть с 900 входами и 33 выходами, которые помечены буквами. Если на входе сети – изображение буквы «А», то максимальное значение выходного сигнала достигается на выходе «А». Аналогично сеть работает для всех 33 букв |

Теперь, когда стало ясно, что именно мы хотим построить, мы можем переходить к вопросу «как строить такую сеть». Этот вопрос решается в два этапа.

1. Выбор типа (архитектуры) сети.
2. Подбор весов (обучение) сети.

На первом этапе следует выбрать следующее:

– какие нейроны мы хотим использовать (число входов, передаточные функции);

– каким образом следует соединить их между собой;

– что взять в качестве входов и выходов сети.

Эта задача на первый взгляд кажется необозримой, но, к счастью, нам необязательно придумывать нейросеть «с нуля» – существует несколько десятков различных нейросетевых архитектур, причем эффективность многих из них доказана математически. Наиболее популярные и изученные архитектуры – это *многослойный персептрон*, *нейросеть с общей регрессией*, *сети Кохон*ена и другие, которые будут рассмотрены ниже.

На втором этапе нам следует «обучить» выбранную сеть, т.е. подобрать такие значения ее весов, чтобы сеть работала нужным образом. Необученная сеть подобна ребенку – ее можно научить чему угодно. В используемых на практике нейросетях количество весов может составлять несколько десятков тысяч, поэтому обучение – действительно сложный процесс. Для многих архитектур разработаны специальные алгоритмы обучения, которые позволяют настроить веса сети определенным образом.

В зависимости от функций, выполняемых нейронами в сети, можно выделить три их типа:

– входные нейроны – это нейроны, на которые подается входной

вектор, кодирующий входное воздействие или образ внешней среды; в них обычно не осуществляется вычислительных процедур, информация передается с входа на выход нейрона путем изменения его активации;

– выходные нейроны – это нейроны, выходные значения которых представляют выход сети;

– промежуточные нейроны – это нейроны, составляющие основу искусственных нейронных сетей.

В большинстве нейронных моделей тип нейрона связан с его расположением в сети. Если нейрон имеет только выходные связи, то это входной нейрон, если наоборот – выходной нейрон. Однако может встретиться случай, когда выход топологически внутреннего нейрона рассматривается как часть выхода сети. В процессе функционирования (эволюции состояния) сети осуществляется преобразование входного вектора в выходной, т.е. некоторая переработка информации. Конкретный вид выполняемого сетью преобразования информации обусловливается не только характеристиками нейроподобных элементов (функциями активации и т.п.), но и особенностями ее архитектуры. А именно, той или иной топологией межнейронных связей, выбором определенных подмножеств нейроподобных элементов для ввода и вывода информации, способами обучения сети, наличием или отсутствием конкуренции между нейронами, направлением и способами управления и синхронизации передачи информации между нейронами.

***Топология нейронных сетей***. С точки зрения топологии, среди нейронных сетей, сформированных на основе нейроподобных элементов, можно выделить три основных типа:

– *полносвязные* сети (рис. 5.4 а);

– *многослойные* или слоистые сети (рис. 5.4 б);

– *слабосвязные* сети (т.е. с локальными связями) (рис. 5.4 в).

Полносвязная сеть представляет собой искусственную нейронную сеть, каждый нейрон которой передает свой выходной сигнал остальным нейронам, в том числе и самому себе. Все входные сигналы подаются всем нейронам. Выходными сигналами сети могут быть все или некоторые выходные сигналы нейронов после нескольких тактов функционирования сети.

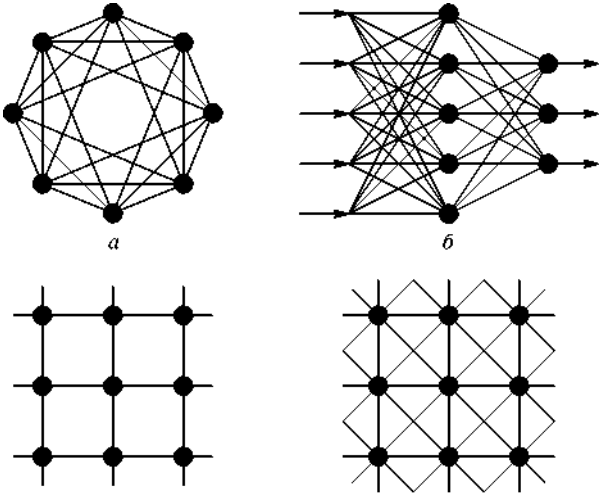



Рис. 5.4. Архитектуры нейронных сетей:
*а* – полносвязная сеть; *б* – многослойная сеть с последовательными связями; *в* – слабосвязные сети.

В многослойных сетях нейроны объединяются в слои. Слой содержит совокупность нейронов с едиными входными сигналами. Число нейронов в каждом слое может быть любым и никак заранее не связано с количеством нейронов в других слоях. В общем случае сеть состоит из $Q$ слоев, пронумерованных слева направо. Внешние входные сигналы подаются на входы нейронов первого слоя (входной слой часто нумеруют как нулевой), а выходами сети являются выходные сигналы последнего слоя. Вход нейронной сети можно рассматривать как выход «нулевого слоя» вырожденных нейронов, которые служат лишь в качестве распределительных точек, суммирования и преобразования сигналов здесь не производится. Кроме входного и выходного слоев в многослойной нейронной сети есть один или несколько промежуточных (скрытых) слоев. Связи от выходов нейронов некоторого слоя $q$ к входам нейронов следующего слоя $(q+1)$ называются последовательными.

В свою очередь, среди слоистых сетей выделяют следующие типы.

1. *Монотонные.* Это специальный частный случай слоистых сетей с дополнительными условиями на связи и элементы. Каждый слой, кроме последнего (выходного), разбит на два блока: возбуждающий $(B)$ и тормозящий $(T)$. Связи между блоками тоже разделяются на тормозящие и возбуждающие. Если от блока $A$ к блоку $C$ ведут только возбу-

ждающие связи, то это означает, что любой выходной сигнал блока является монотонной неубывающей функцией любого выходного сигнала блока $A$ . Если же эти связи только тормозящие, то любой выходной сигнал блока $C$ является невозрастающей функцией любого выходного сигнала блока $A$. Для элементов монотонных сетей необходима монотонная зависимость выходного сигнала элемента от параметров входных сигналов.

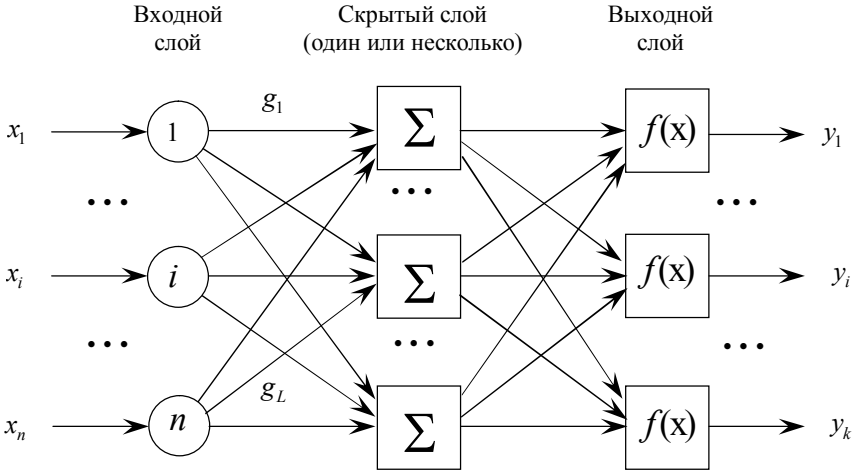



Рис. 5.5. Многослойная (двухслойная) сеть прямого распространения

2. *Сети без обратных связей.* В таких сетях нейроны входного слоя получают входные сигналы, преобразуют их и передают нейронам 1-го скрытого слоя, далее срабатывает 1-й скрытый слой и т.д. до $Q$ - го, который выдает выходные сигналы для интерпретатора и пользователя. Если не оговорено противное, то каждый выходной сигнал $i$ - го слоя подастся на вход всех нейронов $(q+l)$ - го слоя; однако возможен вариант соединения $q$ - го слоя с произвольным $(q+p)$ - м слоем.

Следует отметить, что классическим вариантом слоистых сетей являются сети прямого распространения (рис. 5.5).

3. *Сети с обратными связями.* Это сети, у которых информация с последующих слоев передается на предыдущие.

В качестве примера сетей с обратными связями на рис. 5.6 представлены так называемые частично-рекуррентные сети Элмана и Жордана.

Известные сети можно разделить по принципу структуры нейронов в них на гомогенные или однородные и гетерогенные. Гомогенные сети состоят из нейронов одного типа с единой функцией активации. В гетерогенную сеть входят нейроны с различными функциями активации.

Развивая дальше вопрос о возможной классификации нейронных сетей, важно отметить существование *бинарных* и *аналоговых* сетей. Первые из них оперируют с двоичными сигналами, и выход каждого нейрона может принимать только два значения: логический ноль («заторможенное» состояние) и логическая единица («возбужденное» состояние).

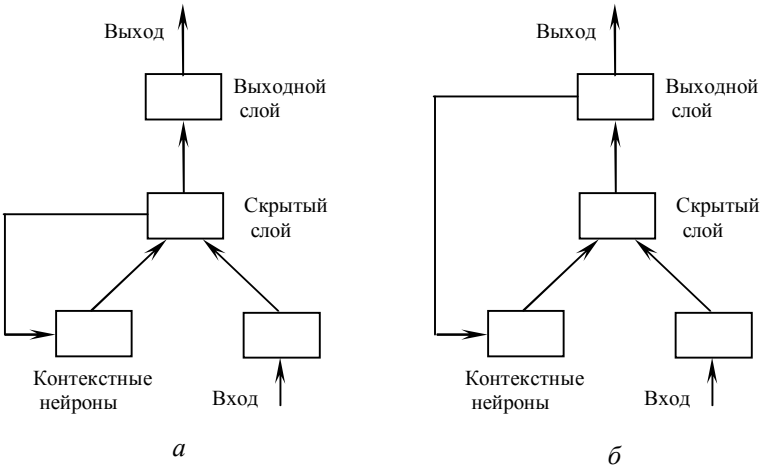



Рис. 5.6. Частично-рекуррентные сети: *а* – Элмана; *б* – Жордана.

Еще одна классификация делит нейронные сети на *асинхронные* и *синхронные*. В первом случае в каждый момент времени свое состояние меняет лишь один нейрон. Во втором – состояние меняется сразу у целой группы нейронов, как правило, у всего слоя. Алгоритмически ход времени в нейронной сети задается итерационным выполнением однотипных действий над нейронами. Далее будут рассматриваться только синхронные нейронные сети.

Сети можно классифицировать также по *числу слоев*.

Теоретически число слоев и число нейронов в каждом слое может быть произвольным, однако фактически оно ограничено ресурсами компьютера или специализированной микросхемы, на которых обычно реализуется искусственная нейронная сеть. Чем сложнее искусственная нейронная сеть, тем масштабнее задачи, подвластные ей.

Выбор структуры искусственной нейронной сети осуществляется в соответствии с особенностями и сложностью задачи. Для решения некоторых отдельных типов задач уже существуют оптимальные, на сегодняшний день, конфигурации. Если же задача не может быть сведена ни к одному из известных типов, разработчику приходится решать сложную проблему синтеза новой конфигурации. При этом он руководствуется несколькими основополагающими принципами:

– возможности сети возрастают с увеличением числа слоев сети и числа нейронов в них;

– введение обратных связей наряду с увеличением возможностей сети поднимает вопрос о так называемой динамической устойчивости сети;

– сложность алгоритмов функционирования сети (в том числе, например, введение нескольких типов синапсов – возбуждающих, тормозящих и др.) также способствует усилению мощи искусственной нейронной сети.

Вопрос о необходимых и достаточных свойствах сети для решения того или иного рода задач представляет собой целое направление *нейрокомпьютерной науки*. Так как проблема синтеза искусственной нейронной сети сильно зависит от решаемой задачи, дать общие подробные рекомендации затруднительно. В большинстве случаев оптимальный вариант получается на основе интуитивного подбора, хотя в литературе приведены доказательства того, что для любого алгоритма существует нейронная сеть, которая может его реализовать. Остановимся на этом подробнее.

Многие задачи: распознавания образов (зрительных, речевых и т.д.), выполнения функциональных преобразований при обработке сигналов, управления, прогнозирования, идентификации сложных систем и т.д., сводятся к следующей математической постановке. Необходимо построить отображение $X \to Y$ такое, чтобы на каждый возможный входной сигнал $X$ формировался правильный выходной сигнал $Y$. Отображение задается конечным набором пар («вход», «известный выход»). Число таких пар (обучающих примеров) существенно меньше общего числа возможных сочетаний значений входных и выходных сигналов. Совокупность всех обучающих примеров носит название обучающей выборки.

В задачах *распознавания* образов $X$ – некоторое представление образа (изображение, вектор чисел и т. д.), $Y$ – номер класса, к которому принадлежит входной образ.

В задачах *управления* $X$ – набор контролируемых параметров управляемого объекта, $Y$ – код, определяющий управляющее воздействие, соответствующее текущим значениям контролируемых параметров.

В задачах *прогнозирования* в качестве входных сигналов используются временны*е* ряды, представляющие значения контролируемых переменных на некотором интервале времени. Выходной сигнал – множество переменных, которое является подмножеством переменных входного сигнала.

При идентификации $X$ и $Y$ представляют входные и выходные сигналы системы соответственно.

Вообще говоря, большая часть прикладных задач может быть сведена к реализации некоторого сложного многомерного функционального преобразования.

В результате построения такого отображения (т.е. $X \to Y$) необходимо добиться того, чтобы:

– обеспечивалось формирование правильных выходных сигналов в соответствии со всеми примерами обучающей выборки;

– обеспечивалось формирование правильных выходных сигналов в соответствии со всеми возможными входными сигналами, которые не вошли в обучающую выборку.

Второе требование в значительной степени усложняет задачу формирования обучающей выборки. В общем виде эта задача в настоящее время еще не решена, однако во всех известных случаях было найдено частное решение. Дальнейшие рассуждения предполагают, что обучающая выборка уже сформирована.

Отметим, что теоретической основой для построения нейронных сетей является следующее

**Утверждение**. *Для любого множества пар входных - выходных векторов произвольной размерности* $\{(X_k, Y_k), k = 1,...,n\}$ *существует двухслойная однородная нейронная сеть с последовательными связями, с сигмоидальными передаточными функциями и с конечным числом нейронов, которая для каждого входного вектора* $X_k$ *формирует соответствующий ему выходной вектор* $Y_k$.

Таким образом, для представления многомерных функций многих переменных может быть использована однородная нейронная сеть, имеющая всего один скрытый слой, с сигмоидальными передаточными функциями нейронов.

Для оценки числа нейронов в скрытых слоях однородных нейронных сетей можно воспользоваться формулой для оценки необходимого

числа синаптических весов $L_w$ (в многослойной сети с сигмоидальными передаточными функциями):

$$\frac{mN}{1+\log_2 N} \le L_w \le m\left(\frac{N}{m}+1\right)(n+m+1)+m\ ,$$

где $n$ – размерность входного сигнала, $m$ – размерность выходного сигнала, $N$ – число элементов обучающей выборки.

Оценив необходимое число весов, можно рассчитать число нейронов в скрытых слоях. Например, число нейронов в двухслойной сети составит:

$$L = \frac{L_w}{n+m}$$

Известны и другие подобные формулы, например, вида

$$2(L+n+m) \le N \le 10(L+n+m)\ ,$$

$$\frac{N}{10} - n - m \le L \le \frac{N}{2} - n - m\ .$$

Точно так же можно рассчитать число нейронов в сетях с большим числом слоев, которые иногда целесообразно использовать: такие многослойные нейронные сети могут иметь меньшие размерности матриц синаптических весов нейронов одного слоя, чем двухслойные сети, реализующие то же самое отображение. К сожалению, строгая методика построения данных сетей пока отсутствует.

Отметим, что отечественному читателю приведенные результаты обычно известны в виде так называемой теоремы о полноте.

**Теорема** (о полноте) *Любая непрерывная функция на замкнутом ограниченном множестве может быть равномерно приближена функциями, вычисляемыми нейронными сетями, если функция активации нейрона дважды непрерывно дифференцируема.*

Таким образом, нейронные сети являются универсальными аппроксимирующими системами.

Очевидно, что процесс функционирования искусственных нейронных сетей, т.е. сущность действий, которые она способна выполнять, зависит от величин синаптических связей, поэтому, задавшись определенной структурой искусственной нейронной сети, отвечающей какой-либо задаче, разработчик сети должен найти оптимальные значения всех переменных весовых коэффициентов (некоторые синаптические связи могут быть постоянными).

Этот этап называется обучением искусственной нейронной сети, и от того, насколько качественно он будет выполнен, зависит способность сети решать поставленные перед ней проблемы во время функционирования.

### 5.4. Обучение нейронных сетей

Обучить нейронную сеть – значит, сообщить ей, чего мы от нее добиваемся. Этот процесс очень похож на обучение ребенка алфавиту. Показав ребенку изображение буквы «$A$», мы спрашиваем его: «Какая это буква?» Если ответ неверен, мы сообщаем ребенку тот ответ, который мы хотели бы от него получить: «Это буква $A$». Ребенок запоминает этот пример вместе с верным ответом, т.е. в его памяти происходят некоторые изменения в нужном направлении. Мы будем повторять процесс предъявления букв снова и снова до тех пор, когда все буквы будут твердо запомнены. Такой процесс называют «обучение с учителем» (рис. 6.7).

При обучении сети мы действуем совершенно аналогично. У нас имеется некоторая база данных, содержащая примеры (набор рукописных изображений букв). Предъявляя изображение буквы «$A$» на вход сети, мы получаем от нее некоторый ответ, не обязательно верный. Нам известен и верный (желаемый) ответ – в данном случае нам хотелось бы, чтобы на выходе с меткой «$A$» уровень сигнала был максимален. Обычно в качестве желаемого выхода в задаче классификации берут набор $(1, 0, 0,...)$, где $1$ стоит на выходе с меткой «$A$», а $0$ – на всех остальных выходах. Вычисляя разность между желаемым ответом и реальным ответом сети, мы получаем (для букв русского алфавита) 33 числа – вектор ошибки. Алгоритм обучения – это набор формул, который позволяет по вектору ошибки вычислить требуемые поправки для весов сети. Одну и ту же букву (а также различные изображения одной и той же буквы) мы можем предъявлять сети много раз. В этом смысле обучение скорее напоминает повторение упражнений в спорте – тренировку.

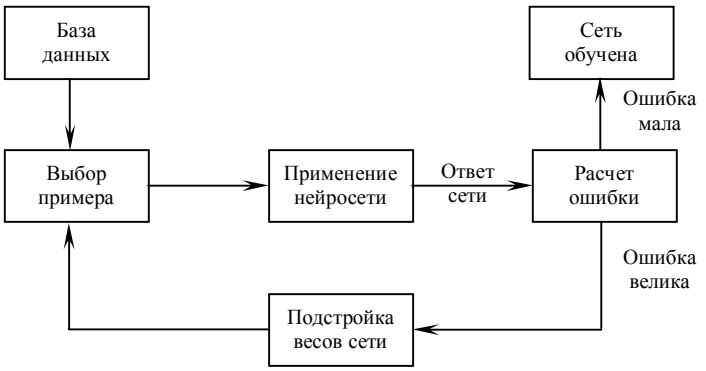



Рис. 5.7. Иллюстрация процесса обучения нейронной сети

Оказывается, что после многократного предъявления примеров веса сети стабилизируются, причем сеть дает правильные ответы на все (или почти все) примеры из базы данных. В таком случае говорят, что «сеть выучила все примеры», «сеть обучена», или «сеть натренирована». В программных реализациях можно видеть, что в процессе обучения функция ошибки (например, сумма квадратов ошибок по всем выходам) постепенно уменьшается. Когда функция ошибки достигает нуля или приемлемого малого уровня, тренировку останавливают, а полученную сеть считают натренированной и готовой к применению на новых данных.

Важно отметить, что вся информация, которую сеть имеет о задаче, содержится в наборе примеров. Поэтому качество обучения сети напрямую зависит от количества примеров в обучающей выборке, а также от того, насколько полно эти примеры описывают данную задачу. Так, например, бессмысленно использовать сеть для предсказания финансового кризиса, если в обучающей выборке кризисов не представлено. Считается, что для полноценной тренировки требуется хотя бы несколько десятков (а лучше сотен) примеров.

Математически процесс обучения можно описать следующим образом.

В процессе функционирования нейронная сеть формирует выходной сигнал $Y$ в соответствии с входным сигналом $X$, реализуя некоторую функцию $Y = G(X)$. Если архитектура сети задана, то вид функции $G$ определяется значениями синаптических весов и смещений сети.

Пусть решением некоторой задачи является функция $Y = F(X)$, заданная парами входных-выходных данных $(X^1, Y^1)$, $(X^2, Y^2)$,..., $(X^N, Y^N)$ для которых $Y^k = F(X^k)$, $k = 1, 2, ..., N$.

Обучение состоит в поиске (синтезе) функции $G$, близкой к $F$ в смысле некоторой функции ошибки $E$ (см. рис. 6.7).

Если выбраны множество обучающих примеров – пар $(X^k, Y^k)$ (где $k = 1, 2, ..., N$) и способ вычисления функции ошибки $E$, то обучение нейронной сети превращается в задачу многомерной оптимизации, имеющую очень большую размерность, при этом, поскольку функция $E$ может иметь произвольный вид, обучение в общем случае – *многоэкстремальная невыпуклая задача оптимизации.*

Для решения этой задачи могут быть использованы следующие (итерационные) алгоритмы:

– алгоритмы локальной оптимизации с вычислением частных производных первого порядка;

– алгоритмы локальной оптимизации с вычислением частных производных первого и второго порядка;

– стохастические алгоритмы оптимизации;

– алгоритмы глобальной оптимизации.

К первой группе относятся: градиентный алгоритм (метод скорейшего спуска); методы с одномерной и двумерной оптимизацией целевой функции в направлении антиградиента; метод сопряженных градиентов; методы, учитывающие направление антиградиента на нескольких шагах алгоритма.

Ко второй группе относятся: метод Ньютона, методы оптимизации с разреженными матрицами Гессе, квазиньютоновские методы, метод Гаусса-Ньютона, метод Левенберга-Марквардта и др.

Стохастическими методами являются: поиск в случайном направлении, метод Монте-Карло (численный метод статистических испытаний).

Задачи глобальной оптимизации решаются с помощью перебора значений переменных, от которых зависит целевая функция (функция ошибки $E$ ).

***Алгоритм обратного распространения.*** Рассмотрим идею одного из самых распространенных алгоритмов обучения – алгоритм обратного распространения ошибки (*back propagation*). Это итеративный градиентный алгоритм обучения, который используется с целью минимизации среднеквадратичного отклонения текущего выхода и желаемого выхода многослойных нейронных сетей.

Алгоритм обратного распространения используется для обучения многослойных нейронных сетей с последовательными связями вида рис. 6.5. Как отмечено выше, нейроны в таких сетях делятся на группы с общим входным сигналом – слои, при этом на каждый нейрон первого слоя подаются все элементы внешнего входного сигнала, а все выходы нейронов $m$-го слоя подаются на каждый нейрон слоя $(q+1)$. Нейроны выполняют взвешенное (с синаптическими весами) суммирование элементов входных сигналов; к данной сумме прибавляется смещение нейрона. Над полученным результатом выполняется активационной функцией затем нелинейное преобразование. Значение функции активации есть выход нейрона.

В многослойных сетях оптимальные выходные значения нейронов всех слоев, кроме последнего, как правило, неизвестны, и трех- или более слойный персептрон уже невозможно обучить, руководствуясь только величинами ошибок на выходах нейронной сети. Наиболее приемлемым вариантом обучения в таких условиях оказался градиентный метод поиска минимума функции ошибки с рассмотрением сигналов ошибки

от выходов нейронной сети к ее входам, в направлении, обратном прямому распространению сигналов в обычном режиме работы. Этот алгоритм обучения нейронной сети получил название *процедуры обратного распространения*.

В данном алгоритме функция ошибки представляет собой сумму квадратов рассогласования (ошибки) желаемого выхода сети и реального. При вычислении элементов вектора-градиента использован своеобразный вид производных функций активации сигмоидального типа. Алгоритм действует циклически (итеративно), и его циклы принято называть эпохами. На каждой эпохе на вход сети поочередно подаются все обучающие наблюдения, выходные значения сети сравниваются с целевыми значениями и вычисляется ошибка. Значение ошибки, а также градиента поверхности ошибок используется для корректировки весов, после чего все действия повторяются. Начальная конфигурация сети выбирается случайным образом, и процесс обучения прекращается, либо когда пройдено определенное количество эпох, либо когда ошибка достигнет некоторого определенного уровня малости, либо когда ошибка перестанет уменьшаться (пользователь может сам выбрать нужное условие остановки).

Приведем словесное описание *алгоритма*.

1. Весам сети присваиваются небольшие начальные значения.

2. Выбирается очередная обучающая пара $(X, Y)$ из обучающего множества; вектор $X$ подается на вход сети.

3. Вычисляется выход сети.

4. Вычисляется разность между требуемым (целевым, $Y$) и реальным (вычисленным) выходом сети.

***5.*** Веса сети корректируются так, чтобы минимизировать ошибку.

***6.*** Шаги со 2-го по 5-й повторяются для каждой пары обучающего множества до тех пор, пока ошибка на всем множестве не достигнет приемлемой величины.

Шаги 2 и 3 подобны тем, которые выполняются в уже обученной сети.

Вычисления в сети выполняются послойно. На шаге 3 каждый из выходов сети вычитается из соответствующей компоненты целевого вектора с целью получения ошибки. Эта ошибка используется на шаге 5 для коррекции весов сети.

Шаги 2 и 3 можно рассматривать как «проход вперед», так как сигнал распространяется по сети от входа к выходу. Шаги 4 и 5 составляют «обратный проход», поскольку здесь вычисляемый сигнал ошибки распространяется обратно по сети и используется для подстройки весов.

Математическое представление алгоритма рассмотрим вначале на примере простейшей нейронной сети, содержащей только один нейрон (условно показанный кружком, рис. 5.8).

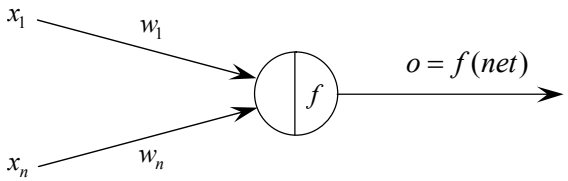



Рис. 5.8. Нейронная сеть из одного нейрона

Будем полагать, что выход (*output*) сети $o = f(net)$ определяется функцией активации сигмоидного (см. табл. 5.1) типа

$$o = o(\mathrm{w}^T \mathrm{x}) = \frac{1}{1 + e^{-\mathrm{w}^T \mathrm{x}}},$$

где $\mathrm{x}^T = (x_1, ..., x_n)$ – вектор входных сигналов, $\mathrm{w}^T = (w_1, ..., w_n)$ – вектор весов сети, «$T$» – символ транспонирования.

Предположим, далее, что для обучения сети используется выборка

$$\mathrm{x}^1 = (x_1^1, x_2^1, ..., x_n^1)^T, \quad y^1,$$
$$\mathrm{x}^2 = (x_1^2, x_2^2, ..., x_n^2)^T, \quad y^2,$$
$$. . . . . .$$
$$\mathrm{x}^N = (x_1^N, x_2^N, ..., x_n^N)^T, \quad y^N,$$

где $y^k$ – значения желаемого (целевого) выхода.

В качестве функции ошибки для $k$ - го образца ($k$ - го элемента обучающей выборки) примем величину, пропорциональную квадрату разности желаемого выхода и выхода сети:

$$E_k = \frac{1}{2}(y^k - o^k)^2 = \frac{1}{2}\left(y^k - o^k(\mathrm{w}^T \mathrm{x}^k)\right)^2 = \frac{1}{2}\left(y^k - \frac{1}{1 + e^{-\mathrm{w}^T \mathrm{x}^k}}\right)^2.$$

Соответственно, суммарная функция ошибки по всем элементам выборки:

$$E = \sum_{k=1}^{N} E_k .$$

Очевидно, как $E_k$, так и $E$ являются функциями вектора весов сети $\mathrm{w}$. Задача обучения сети сводится в данном случае к подбору такого

вектора $\mathrm{w}$, при котором достигается минимум $E$. Данную задачу (оптимизации) будем решать градиентным методом, используя соотношение

$$\mathrm{w} = \mathrm{w} - \eta\, E'_k(\mathrm{w})$$

где «=» – оператор присвоения, $E'_k(\mathrm{w})$ – обозначение вектора-градиента, $\eta$ – некоторая константа.

Представляя данный вектор в развернутом виде и учитывая приведенное выше выражение для производной сигмоидной функции, получим:

$$E'_k(\mathrm{w}) = \frac{d}{d\,\mathrm{w}}\left(\frac{1}{2}\left(y^k - \frac{1}{1+e^{-\mathrm{w}^T\mathrm{x}^k}}\right)^2\right) = -(y^k - o^k)\,o^k(1-o^k)\,\mathrm{x}^k .$$

Это дает возможность записать алгоритм коррекции (подстройки) вектора весовых коэффициентов сети в форме

$$\mathrm{w} = \mathrm{w} + \eta(y^k - o^k)\,o^k(1-o^k)\,\mathrm{x}^k = \mathrm{w} + \eta\,\delta_k\,\mathrm{x}^k$$

где

$$\delta_k = (y^k - o^k)\,o^k(1-o^k)$$

Полученные математические выражения полностью определяют *алгоритм обучения* рассматриваемой нейронной сети, который может быть представлен теперь в следующем виде.

1. Задаются некоторые $\eta$ $(0<\eta<1)$, $E_{\max}$ и некоторые малые случайные веса $w_i$ сети.

2. Задаются $k=1$ и $E=0$.

3. Вводится очередная обучающая пара $(\mathrm{x}^k, \mathrm{y}^k)$. Производятся обозначения

$$\mathrm{x} = \mathrm{x}^k, \quad \mathrm{y} = \mathrm{y}^k$$

и вычисляется величина выхода сети:

$$\mathrm{o} = \mathrm{o}(\mathrm{w}^T\mathrm{x}) = \frac{1}{1+e^{-\mathrm{w}^T\mathrm{x}}} .$$

4. Обновляются (корректируются) веса:

$$\mathrm{w} = \mathrm{w} + \eta(\mathrm{y}-\mathrm{o})\,\mathrm{o}(1-\mathrm{o})\,\mathrm{x}$$

5. Корректируется (наращивается) значение функции ошибки:

$$E = E + \frac{1}{2}(\mathrm{y}-\mathrm{o})^2 .$$

6. Если $k<N$, тогда $k=k+1$ и переход к шагу 3, в противоположном случае – переход на шаг 7.

7. Завершение цикла обучения. Если $E < E_{\max}$ ,то окончание всей процедуры обучения. Если $E \geq E_{\max}$ , тогда начинается новый цикл обучения переходом к шагу 2.

Рассмотрим теперь более общий случай, полагая, что двухслойная нейронная сеть содержит несколько $(L)$ скрытых нейронов и один выходной (рис. 5.9).

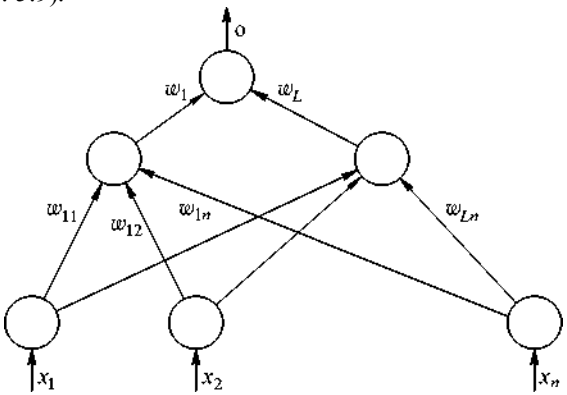



Рис. 5.9. Двухслойная нейронная сеть.

В данном случае функция ошибки зависит от векторов весов скрытого слоя и вектора весов, связанных с выходным нейроном. Выход сети описывается выражением

$$\mathrm{o}^k = \frac{1}{1+e^{-\mathrm{W}^T \mathrm{x}}},$$

где $W$ – вектор весов выходного нейрона, $\mathrm{o}^k$ – вектор выходов нейронов скрытого слоя с элементами

$$o_i^k = \frac{1}{1+e^{-w_i^T \mathrm{x}^k}}$$

где $w_i$ обозначает вектор весов, связанных с $i$ - м скрытым нейроном, $i = 1,...,L$ .

Правило корректировки весов в рассматриваемой нейронной сети также основано на минимизации квадратичной функции ошибки градиентным методом на основе выражений:

$$W = W - \eta \frac{\partial E_k(W,w)}{\partial W},$$

$$w_i = w_i - \eta \frac{\partial E_k(W,w)}{\partial w_i},$$

где $\eta = \text{const}$ коэффициент скорости обучения, $0<\eta<1$; $i = 1,2,...,L$.

Используя правило дифференцирования сложной функции и выражение для производной сигмоидной функции активации, получим:

$$\frac{\partial E_k(W,w)}{\partial W} = \frac{1}{2}\frac{\partial}{\partial W}\left(y^k - \frac{1}{1+e^{-W^T o^k}}\right)^2 = -(\mathrm{y}^{\mathrm{k}} - \mathrm{O}^{\mathrm{k}})\,\mathrm{o}^{\mathrm{k}}(1-\mathrm{O}^{\mathrm{k}})\,\mathrm{o}^{\mathrm{k}}$$

откуда следует

$$W = W + \eta(\mathrm{y}^k - \mathrm{O}^k)\,\mathrm{O}^k(1-\mathrm{O}^k)\,\mathrm{o}^k = W + \eta\delta_k \mathrm{o}^{\mathrm{k}}$$

или в скалярной форме

$$W_i = W_i + \eta\delta_k \mathrm{o}_i^k, \quad i = 1,2,...,L,$$

где

$$\delta_k = (\mathrm{y}^k - \mathrm{O}^k)\,\mathrm{O}^k(1-\mathrm{O}^k).$$

Поступая аналогично, найдем

$$\frac{\partial E_k(W,w)}{\partial w_i} = -(\mathrm{y}^k - \mathrm{O}^k)\,\mathrm{O}^k(1-\mathrm{O}^k)W_i \mathrm{o}_i^k \mathrm{x}^k,$$

откуда получаем

$$w_i = w_i + \eta\delta_k W_i \mathrm{o}_i^k(1-\mathrm{o}_i^k)\mathrm{x}^k,$$

или (в скалярной форме)

$$w_{ij} = w_{ij} + \eta\delta_k W_i \mathrm{o}_i^k(1-\mathrm{o}_i^k)x_j^k, \quad i = 1,2,...,L, \quad j = 1,2,..,n.$$

*Алгоритм обучения* может быть теперь представлен в виде следующих шагов.

1. Задаются некоторые $\eta$ $(0<\eta<1)$, $E_{\max}$ и некоторые малые случайные веса $w_i$ сети.

2. Задаются $k = 1$ и $E = 0$.

3. Вводится очередная обучающая пара $(\mathrm{x}^k, \mathrm{y}^k)$. Производятся обозначения

$$\mathrm{x} = \mathrm{x}^k, \quad \mathrm{y} = \mathrm{y}^k$$

и вычисляется величина выхода сети:

$$\mathrm{O} = \frac{1}{1+e^{-\mathrm{W}^T \mathrm{o}}},$$

где $W$ – вектор весов выходного нейрона, $\mathrm{o}^k$ – вектор выходов нейронов скрытого слоя с элементами

$$o_i = \frac{1}{1+e^{-\mathrm{W}_i^T \mathrm{x}}},$$

$\mathrm{w}_i$ – обозначает вектор весов, связанных с $i$ - м скрытым нейроном, $i = 1, 2, ..., L$.

4. Производится корректировка весов выходного нейрона:

$$W = W + \eta \delta_k \mathrm{o}$$

где $\delta = (\mathrm{y} - \mathrm{O})\mathrm{O}(1 - \mathrm{O})$.

5. Корректируются веса нейронов скрытого слоя:

$$w_i = w_i + \eta \, \delta W_i \, \mathrm{o_i}(1 - \mathrm{o_i})\mathrm{x}, \quad i = 1, 2, ..., L.$$

6. Корректируется (наращивается) значение функции ошибки:

$$E = E + \frac{1}{2}(y - o)^2$$

Если к < N, тогда $k = k + 1$ и переход к шагу 3, в противоположном случае переход на шаг 8.

7. Завершение цикла обучения. Если $E < E_{\max}$ то окончание всей процедуры обучения. Если $E \geq E_{\max}$, тогда начинается новый цикл обучения переходом к шагу 2.

Рассмотренная процедура может быть легко обобщена на случай сети с произвольным количеством слоев и нейронов в каждом слое. Обратим внимание, что в данной процедуре сначала происходит коррекция весов для выходного нейрона, а затем – для нейронов скрытого слоя, т.е. от конца сети к ее началу. Отсюда и название – обратное распространение ошибки. Ввиду использования для обозначений греческой буквы $\delta$, эту процедуру обучения называют еще иногда *обобщенным дельта-правилом.*

Дадим изложенному геометрическую интерпретацию.

В алгоритме *обратного распространения* вычисляется вектор градиента поверхности ошибок. Этот вектор указывает направление кратчайшего спуска по поверхности из данной точки, поэтому, если мы «немного» продвинемся по нему, ошибка уменьшится. Последовательность таких шагов (замедляющаяся по мере приближения к дну) в конце концов приведет к минимуму того или иного типа. Определенную трудность здесь представляет вопрос о том, какую нужно брать длину шагов (что определяется величиной коэффициента скорости обучения $\eta$).

При большой длине шага сходимость будет более быстрой, но имеется опасность перепрыгнуть через решение или (если поверхность ошибок имеет особо вычурную форму) уйти в неправильном направлении. Классическим примером такого явления при обучении нейронной сети

является ситуация, когда алгоритм очень медленно продвигается по узкому оврагу с крутыми склонами, прыгая с одной его стороны на другую. Напротив, при маленьком шаге, вероятно, будет схвачено верное направление, однако при этом потребуется достаточно много итераций. На практике величина шага берется пропорциональной крутизне склона (так что алгоритм замедляет ход вблизи минимума) с некоторой константой $\eta$, которая, как отмечалось, называется коэффициентом скорости обучения. Правильный выбор скорости обучения зависит от конкретной задачи и обычно осуществляется опытным путем; эта константа может также зависеть от времени, уменьшаясь по мере продвижения алгоритма.

Обычно этот алгоритм видоизменяется таким образом, чтобы включать слагаемое импульса (или инерции). Этот член способствует продвижению в фиксированном направлении, поэтому если было сделано несколько шагов в одном и том же направлении, то алгоритм «увеличивает скорость», что (иногда) позволяет избежать локального минимума, а также быстрее проходить плоские участки.

Таким образом, алгоритм действует итеративно, и его шаги принято называть эпохами. На каждой эпохе на вход сети поочередно подаются все обучающие наблюдения, выходные значения сети сравниваются с целевыми значениями и вычисляется ошибка. Значение ошибки, а также градиента поверхности ошибок используется для корректировки весов, после чего все действия повторяются. Начальная конфигурация сети выбирается случайным образом, и процесс обучения прекращается, либо когда пройдено определенное количество эпох, либо когда ошибка достигнет некоторого определенного уровня малости, либо когда ошибка перестанет уменьшаться (пользователь может сам выбрать нужное условие остановки).

Классический метод обратного распространения относится к алгоритмам с линейной сходимостью и известными недостатками. К недостаткам метода относятся невысокая скорость сходимости, возможность сходиться не к глобальному, а к локальным решениям минимумам отмеченной функции, возможность паралича сети, при котором большинство нейронов функционирует при очень больших значениях аргумента функций активации. На пологом участке функции процесс обучения практически замирает поскольку ошибка пропорциональна производной, которая на данных участках мала.

Для устранения этих недостатков были предложены многочисленные модификации алгоритма обратного распространения, которые связаны с использованием различных функций ошибки, различных процедур определения направления и величины шага и т. п.

*Переобучение и обобщение*. Одна из наиболее серьезных трудностей изложенного подхода заключается в том, что таким образом мы минимизируем не ту ошибку, которую на самом деле нужно минимизировать, – ошибку, которую можно ожидать от сети, когда ей будут подаваться совершенно новые наблюдения.

Иначе говоря, мы хотели бы, чтобы нейронная сеть обладала способностью *обобщать* результат на новые наблюдения. В действительности сеть обучается минимизировать ошибку на обучающем множестве, и в отсутствие идеального и бесконечно большого обучающего множества это совсем не то же самое, что минимизировать «настоящую» ошибку на поверхности ошибок в заранее неизвестной модели явления.

Сильнее всего это различие проявляется в проблеме переобучения, или слишком близкой подгонки. Это явление проще будет продемонстрировать не для нейронной сети, а на примере аппроксимации посредством полиномов, – при этом суть явления абсолютно та же.

Полином (или многочлен) – это выражение, содержащее только константы и целые степени независимой переменной. *Примеры*:

$$y = 2x + 3, \quad y = 3x^2 + 4x + 1.$$

Графики полиномов могут иметь различную форму, причем, чем выше степень многочлена (и, тем самым, чем больше членов в него входит), тем более сложной может быть эта форма. Если у нас есть некоторые данные, мы можем поставить цель подогнать к ним полиномиальную кривую (модель) и получить таким образом объяснение для имеющейся зависимости. Наши данные могут быть зашумлены, поэтому нельзя считать, что самая лучшая модель задается кривой, которая в точности проходит через все имеющиеся точки. Полином низкого порядка может быть недостаточно гибким средством для аппроксимации данных, в то время как полином высокого порядка может оказаться чересчур гибким, и будет точно следовать данным, принимая при этом замысловатую форму, не имеющую никакого отношения к форме настоящей зависимости.

Нейронная сеть сталкивается с точно такой же трудностью. Сети с большим числом весов моделируют более сложные функции и, следовательно, склонны к переобучению. Сеть же с небольшим числом весов может оказаться недостаточно гибкой, чтобы смоделировать имеющуюся зависимость.

Как же выбрать «правильную» степень сложности для сети? Почти всегда более сложная сеть дает меньшую ошибку, но это может свидетельствовать не о хорошем качестве модели, а о переобучении.

Ответ состоит в том, чтобы использовать механизм контрольной проверки, при котором часть обучающих наблюдений резервируется и в

обучении по алгоритму обратного распространения не используется. Вместо этого, по мере работы алгоритма, она используется для независимого контроля результата. В самом начале работы ошибки сети на обучающем и контрольном множествах будут одинаковыми (если они существенно отличаются, то, вероятно, разбиение всех наблюдений на два множества было неоднородно). По мере того как сеть обучается, ошибка обучения, естественно, убывает, и, пока обучение уменьшает действительную функцию ошибок, ошибка на контрольном множестве также будет убывать. Если же контрольная ошибка перестала убывать или даже стала расти, это указывает на то, что сеть начала слишком близко аппроксимировать данные и обучение следует остановить. Это явление чересчур точной аппроксимации в процессе обучения и называется переобучением. Если такое случилось, то обычно советуют уменьшить число скрытых элементов и/или слоев, ибо сеть является слишком мощной для данной задачи. Если же сеть, наоборот, была взята недостаточно богатой для того, чтобы моделировать имеющуюся зависимость, то переобучения, скорее всего, не произойдет, и обе ошибки – обучения и проверки – не достигнут достаточного уровня малости.

Описанные проблемы с локальными минимумами и выбором размера сети приводят к тому, что при практической работе с нейронными сетями, как правило, приходится экспериментировать с большим числом различных сетей, порой обучая каждую из них по несколько раз (чтобы не быть введенным в заблуждение локальными минимумами) и сравнивая полученные результаты. Главным показателем качества результата является здесь контрольная ошибка. При этом в соответствии с общенаучным принципом, согласно которому при прочих равных следует предпочесть более простую модель, из двух сетей с приблизительно равными ошибками контроля имеет смысл выбрать ту, которая меньше.

Классический метод обратного распространения относится к алгоритмам с линейной сходимостью. Для увеличения скорости сходимости необходимо использовать матрицы вторых производных функции ошибки.

Были предложены многочисленные модификации алгоритма обратного распространения, которые связаны с использованием различных функций ошибки, различных процедур определения направления и величины шага.

1. Функции ошибки:

– интегральные функций ошибки по всей совокупности обучающих примеров;

– функции ошибки целых и дробных степеней.

2. Процедуры определения величины шага на каждой итерации:

– дихотомия;

– инерционные соотношения.

3. Процедуры определения направления шага:

– с использованием матрицы производных второго порядка (метод Ньютона и др.);

– с использованием направлений на нескольких шагах (партан метод и др.).

***Обучение без учителя.*** Рассмотренный алгоритм обучения нейронной сети с помощью процедуры обратного распространения подразумевает наличие некоего внешнего звена, предоставляющего сети кроме входных также и целевые выходные образы.

Алгоритмы, пользующиеся подобной концепцией, называются алгоритмами обучения с учителем. Для их успешного функционирования необходимо наличие экспертов, создающих на предварительном этапе для каждого входного образа эталонный выходной. Так как создание искусственного интеллекта движется по пути копирования природных прообразов, ученые не прекращают спор на тему, можно ли считать алгоритмы обучения с учителем натуральными или же они полностью искусственны. Например, обучение человеческого мозга, на первый взгляд, происходит без учителя: на зрительные, слуховые, тактильные и прочие рецепторы поступает информация извне, и внутри нервной системы происходит некая самоорганизация. Однако нельзя отрицать и того, что в жизни человека не мало учителей – и в буквальном, и в переносном смысле, – которые координируют внешние воздействия. Вместе с тем, чем бы ни закончился спор приверженцев этих двух концепций обучения – с учителем и без учителя, они обе имеют право на существование.

Главная черта, делающая обучение без учителя привлекательным, – это его «самостоятельность». Процесс обучения, как и в случае обучения с учителем, заключается в подстраивании весов сети. Некоторые алгоритмы, правда, изменяют и структуру сети, т.е. количество нейронов и их взаимосвязи, но такие преобразования правильнее назвать более широким термином – *самоорганизацией*, и здесь они рассматриваться не будут. Очевидно, что подстройка весов может проводиться только на основании информации, доступной в нейроне, т.е. его состояния и уже имеющихся весовых коэффициентов. Исходя из этого соображения и, что более важно, по аналогии с известными принципами самоорганизации нервных клеток, построены алгоритмы обучения Хебба.

*Сигнальный метод обучения Хебба* заключается в изменении весов по следующему правилу:

$$w_{ij} = w_{ij} + \eta o_j^{(q-1)} o_i^{(q)}$$

где $o_j^{(q-1)}$ – выходное значение $j$ - го нейрона слоя $q-1$, $o_i^{(q)}$ – выходное значение $i$ - го нейрона слоя $q$; $w_{ij}$ – весовой коэффициент синапса, соединяющего эти нейроны, $\eta$ – коэффициент скорости обучения. Здесь и далее, для общности, под $q$ подразумевается произвольный слой сети. При обучении по данному методу усиливаются связи между возбужденными нейронами.

Полный алгоритм обучения с применением вышеприведенной формулы будет выглядеть так:

1. На стадии инициализации всем весовым коэффициентам присваиваются небольшие случайные значения.

2. На входы сети подается входной образ, и сигналы возбуждения распространяются по всем слоям согласно принципам классических сетей прямого распространения (*feedforward*), т.е. для каждого нейрона рассчитывается взвешенная сумма его входов, к которой затем применяется активационная (передаточная) функция нейрона, в результате чего получается его выходное значение.

3. На основании полученных выходных значений нейронов по приведенной формуле производится изменение весовых коэффициентов.

4. Цикл с шага 2, пока выходные значения сети не застабилизируются с заданной точностью. Применение этого нового способа определения завершения обучения, отличного от использовавшегося для сети обратного распространения, обусловлено тем, что подстраиваемые значения синапсов фактически не ограничены.

На втором шаге цикла попеременно предъявляются все образы из входного набора.

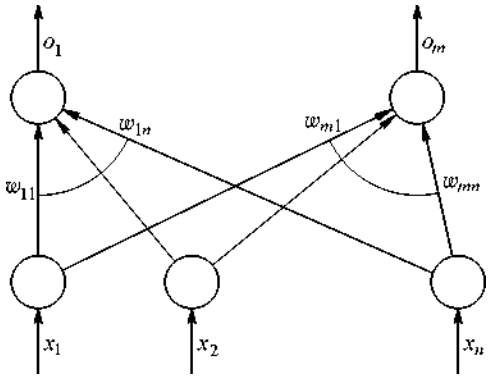



Рис. 5.10. Структура сети Кохонена.

Следует отметить, что вид откликов на каждый класс входных образов неизвестен заранее и будет представлять собой произвольное сочетание состояний нейронов выходного слоя, обусловленное случайным распределением весов на стадии инициализации. Вместе с тем, сеть способна обобщать схожие образы, относя их к одному классу. Тестирование обученной сети позволяет определить топологию классов в выходном слое. Для приведения откликов обученной сети к удобному представлению можно дополнить сеть одним слоем, который, например, по алгоритму обучения однослойного персептрона необходимо заставить отображать выходные реакции сети в требуемые образы.

Другой алгоритм обучения без учителя – алгоритм Кохонена [12] – предусматривает самообучение по правилу «победитель забирает все». Структура сети, реализующей данное правило, представлена на рис. 6.10.

Критерий подстройки весов сети (все векторы весов должны быть нормализованы, т.е. иметь единичную длину: $\| \mathrm{w}_i \| = 1$, $i = 1, 2, \ldots m$ ) выглядит следующим образом:

$$\| \mathrm{x} - \mathrm{w}_r \| = \min_{i=1,2,\ldots,m} \| \mathrm{x} - \mathrm{w}_i \|,$$

где индекс $r$ обозначает нейрон-победитель, соответствующий вектору весов $\mathrm{w}_r$, который ближе всех расположен к (текущему) входному вектору $\mathrm{x}$.

Поскольку (с учетом того, что $\mathrm{w}_i^T \mathrm{w}_i = \| \mathrm{w}_i \|^2 = 1$)

$$\| \mathrm{x} - \mathrm{w}_i \|^2 = (\mathrm{x} - \mathrm{w}_i)^T (\mathrm{x} - \mathrm{w}_i) = \mathrm{x}^T \mathrm{x} - 2\mathrm{w}_i^T \mathrm{x} + 1,$$

процедура нахождения $\mathrm{w}_r$ эквивалентна решению оптимизационной задачи

$$\mathrm{w}_r^T \mathrm{x} = \max_{i=1,2,\ldots,m} \mathrm{w}_i^T \mathrm{x},$$

чему можно дать следующую геометрическую интерпретацию (рис. 5.11).

Так как скалярное произведение $\mathrm{w}_i^T \mathrm{x}$ с учетом $\| \mathrm{w}_i \| = 1$ представляет собой просто проекцию вектора $\mathrm{x}$ на направление вектора $\mathrm{w}_i$, то нейрон-победитель определяется по тому вектору весов, чье направление ближе всего к направлению $\mathrm{x}$ (на рис. 5.11 таким является вектор $\mathrm{w}_2$).

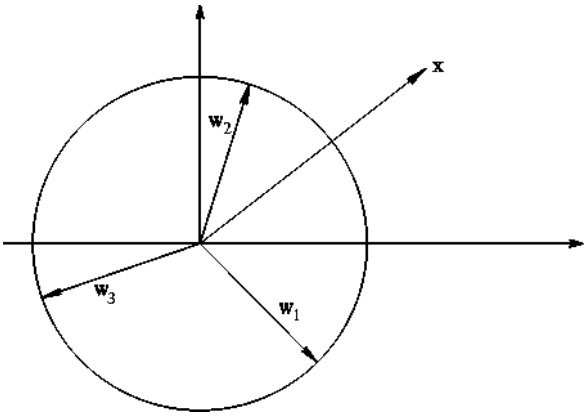



Рис. 5.11. Иллюстрация к алгоритму самообучения Кохонена.

После выявления нейрона-победителя его выход устанавливается равным единице (у остальных нейронов устанавливаются нулевые выходы), а веса корректируются так, чтобы уменьшить квадрат величины рассогласования $\| \mathrm{x} - \mathrm{w}_r \|^2$. При использовании градиентного подхода это приводит к следующей математической формулировке:

$$\mathrm{w}_r = \mathrm{w}_r - \eta \frac{d \| \mathrm{x} - \mathrm{w}_r \|^2}{d\mathrm{w}_r}$$

где, как и раньше, $\eta$ – константа, определяющая скорость обучения.

Найдем производную в правой части последнего выражения:

$$\frac{d \| \mathrm{x} - \mathrm{w}_r \|^2}{d\mathrm{w}_r} = \frac{d\left((\mathrm{x} - \mathrm{w}_r)^T (\mathrm{x} - \mathrm{w}_r)\right)}{d\mathrm{w}_r} =$$

$$= \frac{d\left(\mathrm{x}^T \mathrm{x} - 2\mathrm{w}_r^T \mathrm{x} + \mathrm{w}_r^T \mathrm{w}_r\right)}{d\mathrm{w}_r} = -2(\mathrm{x} - \mathrm{w}_r).$$

При этом

$$\mathrm{w}_r = \mathrm{w}_r + \eta(\mathrm{x} - \mathrm{w}_r).$$

Заметим, что коррекции весов у других нейронов не производится.

*Алгоритм обучения (без учителя) Кохонена* может быть теперь описан следующим образом.

1. Задаются случайные нормализованные по длине векторы $\mathrm{w}_i$.

2. Начало цикла обучения; ввод очередного вектора входов $\mathrm{x}$.

3. Определение нейрона-победителя, корректировка вектора его ве-

сов и задание единичного выхода:

$$w_r = w_r + \eta(x - w_r), \quad o_r = 1.$$

4. Нормализация найденного вектора:

$$w_r = \frac{w_r}{\| w_r \|}$$

5. Задание значений для остальных нейронов:

$$w_i = w_i, \quad o_i = 0, \quad i \neq r.$$

6. Проверка выполнения правила останова (в качестве такого правила можно принять, например, стабилизацию векторов весов на каких-то значениях); если оно не выполнено – продолжение цикла обучения (переходом к шагу 2), в противоположном случае – переход к шагу 7.

7. Конец процедуры обучения.

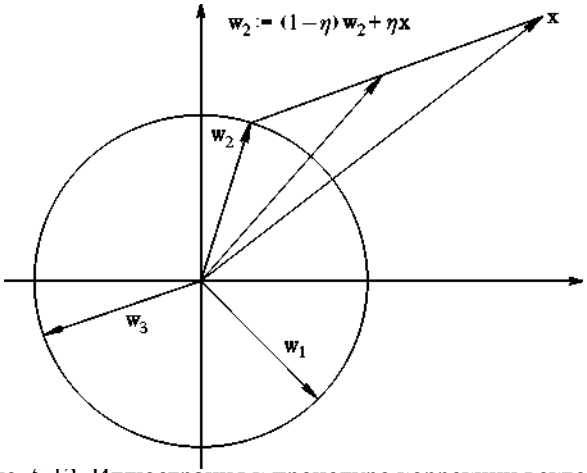



Рис. 5.12. Иллюстрация к процедуре коррекции вектора весов нейрона-победителя.

Заметим, что выражение для коррекции вектора весовых коэффициентов нейрона-победителя может быть представлено в форме

$$w_r = w_r + \eta(x - w_r) = (1 - \eta) w_r + \eta x, \quad 0 \le \eta \le 1.$$

т.е. новое (скорректированное) значение данного вектора является взвешенной суммой старого значения (до коррекции) и предъявленного вектора входов, чему соответствует геометрическая иллюстрация рис. 5.12.

Нетрудно показать, что итоговым результатом подобных коррекций (в условиях рассматриваемого примера для двумерного случая) являются векторы весов, показывающие на центры кластеров (центры группирования) входных образов (рис. 5.13).

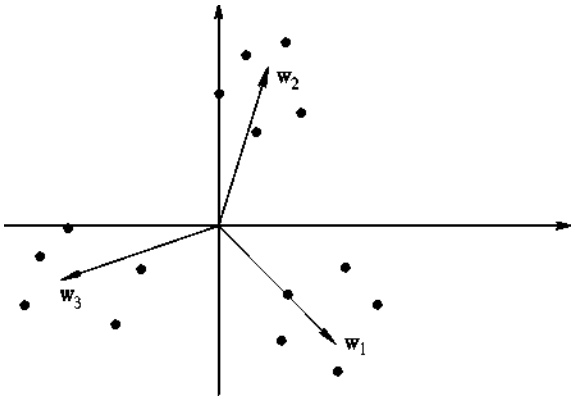



Рис. 5.13. Векторы весов НС после окончания процесса обучения.

Иначе говоря, алгоритм обучения Кохонена обеспечивает решение задачи автоматической классификации, т.е. отнесения предъявленного вектора входов к одному из классов (на рис. 5.13 таких классов 3). Правда, такая классификация возможна только в случае, когда кластеры являются линейно разделимыми (гиперплоскостями) относительно начала координат в пространстве входов нейронной сети.

Отметим, что число нейронов нейронной сети для успешного решения указанной задачи должно быть не меньше, чем число кластеров; поскольку точное число кластеров может быть заранее неизвестно, количество нейронов задают с определенным запасом. «Лишние» нейроны, у которых в процессе обучения сети веса изменяются хаотически, по завершении данного процесса могут быть удалены.

### 5.5. Применение нейронных сетей

После того как сеть обучена, мы можем применять ее для решения различных задач (рис. 5.14).

Важнейшая особенность человеческого мозга состоит в том, что, однажды обучившись определенному процессу, он может вернодействовать и в тех ситуациях, в которых он не бывал в процессе обучения. Например, мы можем читать почти любой почерк, даже если видим его первый раз в жизни. Так же и нейросеть, грамотным образом обученная, может с большой вероятностью правильно реагировать на новые, не предъявленные ей ранее данные. Например, мы можем нарисовать букву

«А» другим почерком, а затем предложить нашей сети классифицировать новое изображение.

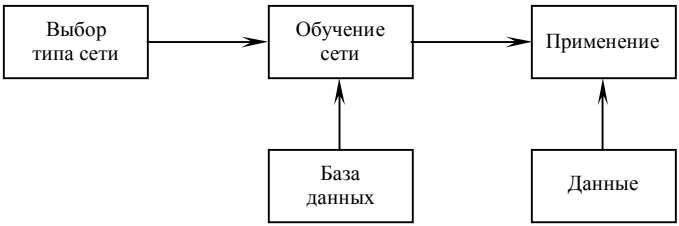



Рис. 5.14. Этапы нейросетевого проекта.

Веса обученной сети хранят достаточно много информации о сходстве и различиях букв, поэтому можно рассчитывать на правильный ответ и для нового варианта изображения.

*Области применения нейросетей: классификация*. Отметим, что задачи классификации (типа распознавания букв) очень плохо алгоритмизуются. Если в случае распознавания букв верный ответ очевиден для нас заранее, то в более сложных практических задачах обученная нейросеть выступает как эксперт, обладающий большим опытом и способный дать ответ на трудный вопрос.

Примером такой задачи служит медицинская диагностика, где сеть может учитывать большое количество числовых параметров (энцефалограмма, давление, вес и т.д.). Конечно, «мнение» сети в этом случае нельзя считать окончательным. Классификация предприятий по степени их перспективности – это уже привычный способ использования нейросетей в практике западных компаний (деление компаний на перспективные и убыточные). При этом сеть также использует множество экономических показателей, сложным образом связанных между собой.

Нейросетевой подход особенно эффективен в задачах экспертной оценки по той причине, что он сочетает в себе способность компьютера к обработке чисел и способность мозга к обобщению и распознаванию. Говорят, что у хорошего врача способность к распознаванию в своей области столь велика, что он может провести приблизительную диагностику уже по внешнему виду пациента. Можно согласиться также, что опытный трейдер чувствует направление движения рынка по виду графика. Однако в первом случае все факторы наглядны, т.е. характеристики пациента мгновенно воспринимаются мозгом как «бледное лицо», «блеск в глазах» и т.д. Во втором же случае учитывается только один фактор, показанный на графике, – курс за определенный период времени. Нейросеть позволяет обрабатывать огромное количество факторов

(до нескольких тысяч), независимо от их наглядности, – это универсальный «хороший врач», который может поставить свой диагноз в любой области.

*Кластеризация и поиск зависимостей*. Помимо задач классификации, нейросети широко используются для поиска зависимостей в данных и кластеризации.

Например, нейросеть на основе методики МГУА (метод группового учета аргументов) позволяет на основе обучающей выборки построить зависимость одного параметра от других в виде полинома. Такая сеть может не только мгновенно выучить таблицу умножения, но и найти сложные скрытые зависимости в данных (например, финансовых), которые не обнаруживаются стандартными статистическими методами.

Кластеризация – это разбиение набора примеров на несколько компактных областей (кластеров), причем число кластеров заранее неизвестно. Кластеризация позволяет представить неоднородные данные в более наглядном виде и использовать далее для исследования каждого кластера различные методы. Например, таким образом можно быстро выявить фальсифицированные страховые случаи или недобросовестные предприятия.

*Прогнозирование*. Задачи прогнозирования особенно важны для практики, в частности, для финансовых приложений, поэтому поясним способы применения нейросетей в этой области более подробно.

Рассмотрим практическую задачу, ответ в которой неочевиден, – задачу прогнозирования курса акций на 1 день вперед.

Пусть у нас имеется база данных, содержащая значения курса за последние 300 дней. Простейший вариант в данном случае – попытаться построить прогноз завтрашней цены на основе курсов за последние несколько дней. Понятно, что прогнозирующая сеть должна иметь всего один выход и столько входов, сколько предыдущих значений мы хотим использовать для прогноза – например, 4 последних значения. Составить обучающий пример очень просто: входными значениями будут курсы за 4 последовательных дня, а желаемым выходом – известный нам курс в следующий день за этими четырьмя. Таким образом, каждая строка таблицы с обучающей последовательностью (выборкой) представляет собой обучающий пример, где первые 4 числа – входные значения сети, а пятое число – желаемое значение выхода.

Заметим, что объем обучающей выборки зависит от выбранного количества входов. Если сделать 299 входов, то такая сеть потенциально могла бы строить лучший прогноз, чем сеть с 4 входами, однако в этом случае мы имеем дело с огромным массивом данных, что делает обучение и использование сети практически невозможным. При выборе числа входов следует учитывать это, выбирая разумный компромисс между

глубиной предсказания (число входов) и качеством обучения (объем тренировочного набора).

Широкое распространение технологии на основе нейронных сетей получили, например, в медицине, идеальным полем применения которых, являются задачи классификации, позволяющие определить принадлежность объекта к одному из нескольких возможных классов, т.е. ускорить дифференциальную диагностику заболеваний. При этом для принятия решений могут использоваться самые разнообразные клинические, функциональные данные и лабораторные тесты. Список областей медицины, в которых начали применяться нейронные сети, чрезвычайно обширен и продолжает расти.

Это связано с такими основными достоинствами нейронных сетей, как:

– способ обработки данных больше похож на обработку сигналов, вместо программы – набор искусственных нейронов, вместо программирования – обучение нейронов (настройка синапсов, т.е. весовых коэффициентов нейронов);

– устойчивость к шумам, т.е. искажения данных не влияют существенно на результат (в том числе выход из строя отдельных нейронов).

– представление обрабатываемых объектов весами нейронов неявно. В результате чего сеть может работать с объектами, которые ей ранее не встречались, и обобщать результаты обучения.

– простота имитации (эмуляции) нейронной сети на современном компьютере с помощью стандартных пакетов программ (например, STATISTICA Neural Networks [9]).

Следует отметить, что обученная нейронная сеть записывается на диск компьютера как обыкновенный файл и может храниться там столько, сколько необходимо. В любой момент времени можно продолжить обучение нейронной сети решению данной задачи со старой или новой обучающей выборкой.

Вообще говоря, в зависимости от типа решаемой задачи, целесообразно применять нейронную сеть наиболее подходящей для такой задачи структуры. Рассмотрим некоторые, наиболее употребительные виды НС.

***Персептроны.*** В качестве научного предмета искусственные нейронные сети впервые заявили о себе в 40-е годы. Стремясь воспроизвести функции человеческого мозга, исследователи создали простые аппаратные (а позже программные) модели биологического нейрона и системы его соединений. Когда нейрофизиологи достигли более глубокого понимания нервной системы человека, эти ранние попытки стали восприниматься как весьма грубые аппроксимации. Тем не менее, на этом пути были достигнуты впечатляющие результаты, стимулировав-

шие дальнейшие исследования, приведшие к созданию более изощренных сетей.

Первое систематическое изучение искусственных нейронных сетей было предпринято Маккалохом и Питтсом в 1943 г. Позднее они исследовали сетевые парадигмы для распознавания изображений, подвергаемых сдвигам и поворотам, используя при этом простую нейронную модель, показанную на рис. 5.15. Элемент $\Sigma$ умножает каждый вход $x_i$ на вес $w_i$ и суммирует взвешенные входы. Если эта сумма больше заданного порогового значения, выход равен единице, в противном случае – нулю. Эти системы (и множество им подобных) получили название *персептронов*. Они состоят из одного слоя искусственных нейронов, соединенных с помощью весовых коэффициентов с множеством входов (рис. 5.16), хотя в принципе описываются и более сложные системы.

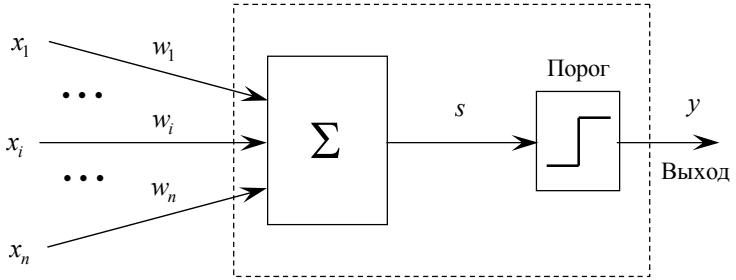



Рис. 5.15. Персептронный нейрон.

В 60-е годы персептроны вызвали большой интерес и оптимизм. Розенблатт доказал замечательные теоремы об обучении персептронов, приводимые ниже. Уидроу дал ряд убедительных демонстраций систем персептронного типа, и исследователи во всем мире стремились изучить возможности этих систем. Первоначальная эйфория сменилась разочарованием, когда оказалось, что персептроны не способны обучиться решению ряда простых задач. Минский строго проанализировал эту проблему и показал, что имеются жесткие ограничения на то, что могут выполнять однослойные персептроны, и, следовательно, на то, чему они могут обучаться. Так как в то время методы обучения многослойных сетей не были известны, исследователи перешли в более многообещающие области, и исследования в области нейронных сетей пришли в упадок. Недавнее открытие методов обучения многослойных сетей в большей степени, чем какой-либо иной фактор, повлияло на возрождение интереса и исследовательских усилий.

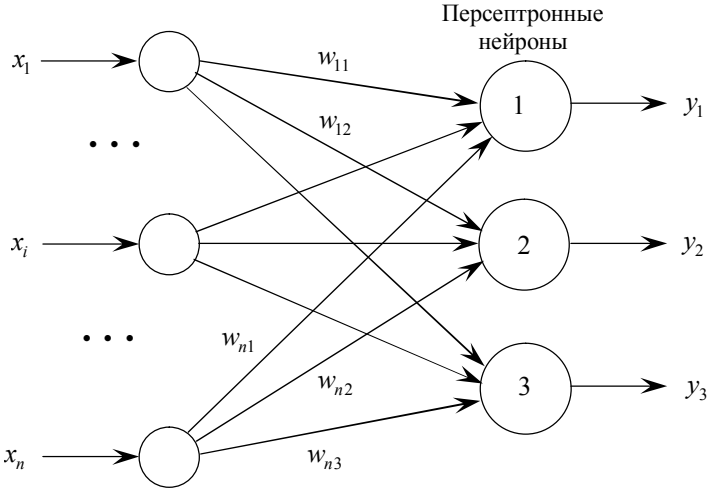



Рис. 5.16. Персептрон со многими выходами.

Работа Минского возможно и охладила пыл энтузиастов персептрона, но обеспечила время для необходимой консолидации и развития лежащей в основе теории. Важно отметить, что анализ Минского не был опровергнут. Он остается важным исследованием и должен изучаться, чтобы ошибки 60-х годов не повторились.

Несмотря на свои ограничения, персептроны широко изучались (хотя не слишком широко использовались). Теория персептронов является основой для многих других типов искусственных нейронных сетей, в силу чего они являются логической исходной точкой для изучения искусственных нейронных сетей.

Рассмотрим в качестве примера трехнейронный персептрон (рис. 5.16), нейроны которого имеют активационную функцию в виде единичного скачка.

На $n$ входов поступают входные сигналы, проходящие по синапсам, на три нейрона, образующие единственный слой этой сети и выдающие три выходных сигнала:

$$y_j = f_j\left(\sum_{i=1}^{n} x_i w_{ij}\right), \quad j = 1,2,3\,.$$

Очевидно, что все весовые коэффициенты синапсов одного слоя нейронов можно свести в матрицу $W$, в которой каждый элемент $w_{ij}$ задает величину $i$ - ый синаптической связи $j$ - го нейрона. Таким обра-

зом, процесс, происходящий в нейронной сети, может быть записан в матричной форме:

$$Y = \mathrm{f}(XW)$$

где $X$ и $Y$ – соответственно входной и выходной сигнальные векторы (здесь и далее под вектором понимается вектор-строка), $\mathrm{f}(S)$ – активационная функция, применяемая поэлементно к компонентам вектора $S$.

На рис. 5.17 представлен двухслойный персептрон, полученный из персептрона с рис. 5.16 путем добавления второго слоя, состоящего из двух нейронов.

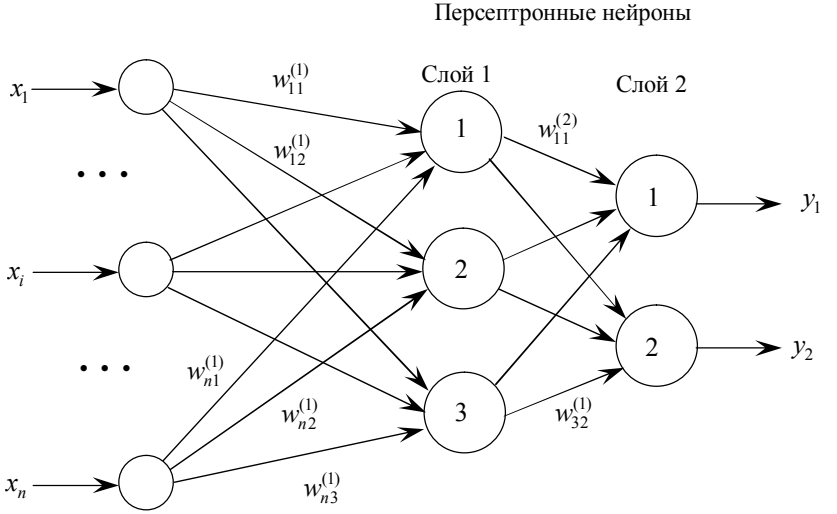



Рис. 5.17. Двухслойный персептрон.

Здесь уместно отметить важную роль нелинейности активационной функции, так как, если бы она не обладала данным свойством или не входила в алгоритм работы каждого нейрона, результат функционирования любой $Q$ - слойной нейронной сети с весовыми матрицами $W^{(q)}$ для каждого слоя $q = 1,...,Q$ сводился бы к перемножению входного вектора сигналов $X$ на матрицу:

$$W_{(\Sigma)} = W^{(1)}...W^{(q)}...W^{(Q)}.$$

Фактически такая $Q$ - слойная нейронная сеть эквивалентна сети с одним скрытым слоем и с весовой матрицей единственного слоя $W_{(\Sigma)}$:

$$Y = X\,W_{(\Sigma)}.$$

Работа персептрона сводится к классификации (обобщению) входных сигналов, принадлежащих $n$ - мерному гиперпространству, по некоторому числу классов. С математической точки зрения это происходит путем разбиения гиперпространства гиперплоскостями. Для случая однослойного персептрона

$$\sum_{i=1}^{n} x_i w_{ij} = \theta_j, \quad j = 1, 2, ..., m.$$

Каждая полученная область является областью определения отдельного класса. Число таких классов для персептрона не превышает $2^n$, где $n$ – число его входов. Однако не все из классов могут быть разделимы данной нейронной сетью. Например, однослойный персептрон, состоящий из одного нейрона с двумя входами, не может реализовать логическую функцию *ИСКЛЮЧАЮЩЕЕ ИЛИ*, т.е. не способен разделить плоскость (двумерное гиперпространство) на две полуплоскости так, чтобы осуществить классификацию входных сигналов по классам $A$ и $B$ (см. табл. 5.3).

Уравнение сети для этого случая

$$x_1 w_1 + x_2 w_2 = \theta$$

является уравнением прямой (одномерной гиперплоскости), которая ни при каких условиях не может разделить плоскость так, чтобы точки из множества входных сигналов, принадлежащие разным классам, оказались по разные стороны от прямой (рис. 2.18). Невозможность реализации однослойным персептроном этой функции получила название проблемы *ИСКЛЮЧАЮЩЕГО ИЛИ*.

**Таблица 5.3**. *Логическая функция ИСКЛЮЧАЮЩЕЕ ИЛИ*

| $x_1$ \ $x_2$ | 0 | 1 |
|---|---|---|
| 0 | *B* | *A* |
| 1 | *A* | *B* |

Отметим, что функции, которые не реализуются однослойным персептроном, называются линейно неразделимыми. Решение задач, подпадающих под это ограничение, заключается в применении двух- и более слойных сетей или сетей с нелинейными синапсами, однако и тогда су-

ществует вероятность, что корректное разделение некоторых входных сигналов на классы невозможно.

Обучение персептрона сводится к формированию весов связей между первым и вторым (см. рис. 5.17) слоями в соответствии со следующим алгоритмом.

***Шаг 1.*** Проинициализировать элементы весовой матрицы (обычно небольшими случайными значениями).

***Шаг 2.*** Подать на входы один из входных векторов, которые сеть должна научиться различать, и вычислить ее выход.

***Шаг 3.*** Если выход правильный, перейти на шаг 4.

Иначе – вычислить разницу между идеальным $d$ и полученным $X$ значениями выхода:

$$\delta = d - Y .$$

Модифицировать веса в соответствии с формулой

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta x_i ,$$

где $t$ и $(t+1)$ – номера соответственно текущей и следующей итераций; $\eta$ – коэффициент скорости обучения; $0 < \eta < 1$; $i$ – номер входа; $j$ – номер нейрона в слое.

Очевидно, что если $d > Y$, то весовые коэффициенты будут увеличены и, тем самым, уменьшат ошибку. В противном случае они будут уменьшены, и $Y$ тоже уменьшится, приближаясь к $d$.

***Шаг 4***. Цикл с шага 2, пока сеть не перестанет ошибаться.

На втором шаге на разных итерациях поочередно в случайном порядке предъявляются все возможные входные векторы. К сожалению, нельзя заранее определить число итераций, которые потребуется выполнить, а в некоторых случаях и гарантировать полный успех.

Сходимость рассмотренной процедуры устанавливается теоремами, утверждающими, что для любой классификации обучающей последовательности можно подобрать такой набор (из бесконечного набора) элементарных нейронов, в котором будет осуществлено разделение обучающей последовательности при помощи линейного

решающего правила, и что, если относительно задуманной классификации можно найти набор элементов, в котором существует решение, то в рамках этого набора оно будет достигнуто в конечный промежуток времени.

***Нейронные сети встречного распространения.*** Объединение разнотипных нейронных структур в единой архитектуре зачастую приводит к свойствам, которых нет у них по отдельности. Причем именно каскадные соединения нейронных структур, специализирующихся на решении различных задач, позволяют решить проблему комплексно.

Нейронные сети встречного распространения, состоящие из входного слоя нейронов и так называемых слоев нейронов Кохонена и Гроссберга, по своим характеристикам существенно превосходят возможности сетей с одним скрытым слоем нейронов. Так, время их обучения задачам распознавания и кластеризации более чем в сто раз меньше времени обучения аналогичным задачам сетей с обратным распространением. Это может быть полезно в тех приложениях, где долгая обучающая процедура невозможна

Одними из определяющих характеристик сети встречного распространения являются ее хорошие способности к обобщению, позволяющие получать правильный выход даже при неполным или зашумленном входном векторе. Это позволяет эффективно использовать данную сеть для распознавания и восстановления образов, а также для усиления сигналов.

В процессе обучения сети встречного распространения входные векторы ассоциируются с соответствующими выходными векторами. Эти векторы могут быть двоичными или непрерывными. После обучения сеть формирует выходные сигналы, соответствующие входным сигналам. Обобщающая способность сети дает возможность получать правильный выход, когда входной вектор неполон или искажен.

Сеть встречного распространения имеет два слоя с последовательными связями. Первый слой – слой Кохонена, второй – слой Гроссберга. Каждый элемент входного сигнала подается на все нейроны слоя Кохонена. Каждый нейрон слоя Кохонена соединен со всеми нейронами слоя Гроссберга. Отличие сети встречного распространения от других многослойных сетей с последовательными связями состоит в операциях, выполняемых нейронами Кохонена и Гроссберга.

В режиме функционирования сети предъявляется входной сигнал $X$ и формируется выходной сигнал $Y$. В режиме обучения на вход сети подается входной сигнал и веса корректируются, чтобы сеть выдавала требуемый выходной сигнал.

*Функционирование сети.*

*Слой Кохонена.* В своей простейшей форме слой Кохонена функционирует по правилу «победитель получает все». Для данного входного вектора один и только один нейрон Кохонена выдает логическую единицу, все остальные выдают ноль. Выход каждого нейрона Кохонена является просто суммой взвешенных элементов входных сигналов:

$$s_j = w_{1j}x_1 + w_{2j}x_2 + \ldots + w_{nj}x_n = \sum_{i=1}^{n} x_i w_{ij} ,$$

где $s_j$ – выход $j$-го нейрона Кохонена, $W_j = \left(w_{1j}, w_{2j}, ..., w_{nj}\right)$ – вектор весов $j$-го нейрона Кохонена, $X = \left(x_1, x_2, ..., x_n\right)$ – вектор входного сигнала, или в векторно-матричной форме:

$$S = X W$$

где $S$ – вектор выходов слоя Кохонена.

Нейрон Кохонена с максимальным значением $s_j$ является «победителем». Его выход равен единице, у остальных он равен нулю.

*Слой Гроссберга.* Слой Гроссберга функционирует в сходной манере. Его выход является взвешенной суммой выходов слоя Кохонена (т.е. он является слоем нейронов с линейными активационными функциями).

Если слой Кохонена функционирует таким образом, что лишь один выход равен единице, а остальные равны нулю, то каждый нейрон слоя Гроссберга выдает величину веса, который связывает этот нейрон с единственным нейроном Кохонена, чей выход отличен от нуля.

*Предварительная обработка входных сигналов.* Рассматриваемая НС требует предварительной обработки входных векторов путем их нормализации. Такая нормализация выполняется путем деления каждой компоненты входного вектора на длину вектора (квадратный корень из суммы квадратов компонент вектора). Это превращает входной вектор в единичный вектор с тем же направлением, т. е. в вектор единичной длины в $n$-мерном пространстве.

*Обучение слоя Кохонена.* Слой Кохонена классифицирует входные векторы в группы схожих. Это достигается с помощью такой подстройки весов слоя Кохонена, что близкие входные векторы активируют один и тот же нейрон данного слоя (затем задачей слоя Гроссберга является получение требуемых выходов).

Слой Кохонена обучается без учителя (самообучается). В результате обучения слой приобретает способность разделять несхожие входные векторы. Какой именно нейрон будет активироваться при предъявлении конкретного входного сигнала, заранее трудно предсказать.

При обучении слоя Кохонена на вход подается входной вектор и вычисляются его скалярные произведения с векторами весов всех нейронов. Скалярное произведение является мерой сходства между входным вектором и вектором весов. Нейрон с максимальным значением скалярного произведения объявляется «победителем» и его веса подстраиваются (весовой вектор приближается к входному).

Уравнение, описывающее процесс обучения, имеет вид

$$w_н = w_c + \eta(x - w_c),$$

где $w_н$ – новое значение веса, соединяющего входную компоненту $x$ с выигравшим нейроном, $w_c$ – предыдущее значение этого веса, $\eta$ – коэффициент скорости обучения.

Каждый вес, связанный с выигравшим нейроном Кохонена, изменяется пропорционально разности между его величиной и величиной входа, к которому он присоединен. Направление изменения минимизирует разность между весом и соответствующим элементом входного сигнала.

Коэффициент скорости обучения $\eta$ вначале обычно полагается равным 0.7 и может затем постепенно уменьшаться в процессе обучения. Это позволяет делать большие начальные шаги для быстрого грубого обучения и меньшие шаги при подходе к окончательной величине.

Если бы с каждым нейроном Кохонена ассоциировался один входной вектор, то слой Кохонена мог бы быть обучен с помощью одного вычисления на вес( $\eta = 1$ ). Как правило, обучающее множество включает много сходных между собой входных векторов, и сеть должна быть обучена активировать один и тот же нейрон Кохонена для каждого из них. Веса этого нейрона должны получаться усреднением входных векторов, которые должны его активировать.

*Обучение слоя Гроссберга.* Выходы слоя Кохонена подаются на входы нейронов слоя Гроссберга. Выходы нейронов вычисляются, как при обычном функционировании. Далее каждый вес корректируется лишь в том случае, если он соединен с нейроном Кохонена, имеющим ненулевой выход. Величина коррекции веса пропорциональна разности между весом и требуемым выходом нейрона Гроссберга.

Обучение слоя Гроссберга – это обучение с учителем, алгоритм использует заданные желаемые выходы.

В полной модели сети встречного распространения имеется возможность получать выходные сигналы по входным сигналам и наоборот. Этим двум действиям соответствуют прямое и обратное распространение сигналов.

Области применения: распознавание образов, восстановление образов (ассоциативная память), сжатие данных (с потерями).

*Недостатки.* Сеть не дает возможности строить точные аппроксимации (точные отображения). В этом сеть значительно уступает сетям с обратным распространением ошибки.

К недостаткам модели также следует отнести слабую теоретическую проработку модификаций сети встречного распространения.

*Преимущества.* Сеть встречного распространения проста. Она дает возможность извлекать статистические свойства из множеств входных сигналов. Кохоненом показано, что для полностью обученной сети веро-

ятность того, что случайно выбранный входной вектор (в соответствии с функцией плотности вероятности входного множества) будет ближайшим к любому заданному весовому вектору, равна $1/p$, $p$ – число нейронов Кохонена.

Сеть быстро обучается. Время обучения по сравнению с обратным распространением может быть в 100 раз меньше. По своим возможностям строить отображения сеть встречного распространения значительно превосходит однослойные персептроны.

Сеть полезна для приложений, в которых требуется быстрая начальная аппроксимация.

Сеть дает возможность строить функцию и обратную к ней, что находит применение при решении практических задач.

*Модификации.* Сети встречного распространения могут различаться способами определения начальных значений синаптических весов. Так, кроме традиционных случайных значений из заданного диапазона, могут быть использованы значения в соответствии с известным методом выпуклой комбинации.

Для повышения эффективности обучения применяется добавление шума к входным векторам.

Еще один метод повышения эффективности обучения – наделение каждого нейрона «чувством справедливости». Если нейрон становится победителем чаще, чем $1/p$, ($p$ – число нейронов Кохонена), то ему временно увеличивают порог, давая тем самым обучаться и другим нейронам.

Кроме «метода аккредитации», при котором для каждого входного вектора активируется лишь один нейрон Кохонена, может быть использован «метод интерполяции», при использовании которого целая группа нейронов Кохонена, имеющих наибольшие выходы, может передавать свои выходные сигналы в слой Гроссберга. Этот метод повышает точность отображений, реализуемых сетью, и реализован в так называемых *самоорганизующихся картах.*

***Нейронные сети Хопфилда и Хэмминга.*** Среди различных конфигураций искусственных нейронных сетей (НС) встречаются такие, при классификации которых по принципу обучения, строго говоря, не подходят ни обучение с учителем, ни обучение без учителя. В таких сетях весовые коэффициенты синапсов рассчитываются только однажды перед началом функционирования сети на основе информации об обрабатываемых данных, и все обучение сети сводится именно к этому расчету. С одной стороны, предъявление априорной информации можно расценивать как помощь учителя, но с другой, – сеть фактически просто запоминает образцы до того, как на ее вход поступают реальные данные,

и не может изменять свое поведение, поэтому говорить о звене обратной связи с «миром» (учителем) не приходится. Из сетей с подобной логикой работы наиболее известны *сеть Хопфилда* и *сеть Хэмминга* (представляющие собой разновидности сетей с обратными связями), которые обычно используются для организации ассоциативной памяти. Далее речь пойдет именно о них. Структурная схема сети Хопфилда приведена на рис. 2.19. Она состоит рис. 2.19. Структурная схема сети Хопфилда из единственного слоя нейронов, число которых является одновременно числом входов и выходов сети. Каждый нейрон связан синапсами со всеми остальными нейронами, а также имеет один входной синапс, через который осуществляется ввод сигнала. Выходные сигналы, как обычно, образуются на аксонах.

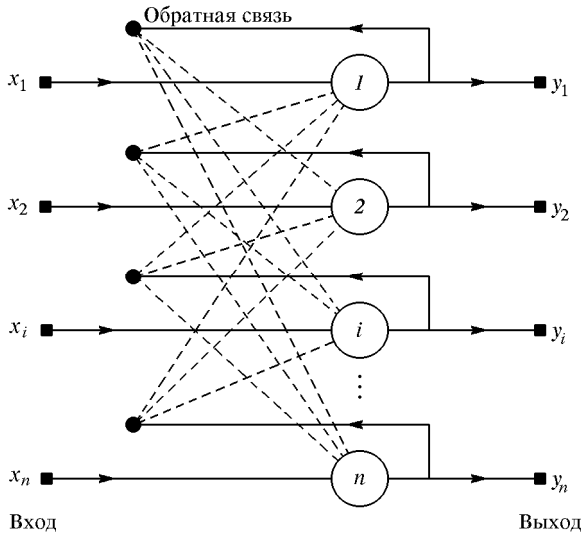



Рис.5.19. Структурная схема сети Хопфилда

Задача, решаемая данной сетью в качестве ассоциативной памяти, как правило, формулируется следующим образом. Известен некоторый набор двоичных сигналов (изображений, звуковых оцифровок, прочих данных, описывающих некие объекты или характеристики процессов), которые считаются образцовыми. Сеть должна уметь из произвольного неидеального сигнала, поданного на ее вход, выделить («вспомнить» по частичной информации) соответствующий образец (если такой есть) или «дать заключение» о том, что входные данные не соответствуют ни одному из образцов. В общем случае, любой сигнал может быть описан вектором $X = \{x_i : i = 1,2,...,n\}$, $n$ – число нейронов в сети и размерность

входных и выходных векторов. Каждый элемент $x_i$ равен либо $(+1)$, либо $(-1)$. Обозначим вектор, описывающий $k$ - й образец, через $X^k$, а его компоненты соответственно $x_i^k$, $k = 1,2,...,m$, где $m$ – в данном случае число образцов. Когда сеть распознает (или «вспомнит») какой-либо образец на основе предъявленных ей данных, ее выходы будут содержать именно его, т.е. $Y = X^k$, где $Y$ – вектор выходных значений сети: $Y = \{y_i : i = 1,2,...,n\}$. В противном случае, выходной вектор не совпадет ни с одним образцовым.

Если, например, сигналы представляют собой некие изображения, то, отобразив в графическом виде данные с выхода сети, можно будет увидеть картинку, полностью совпадающую с одной из образцовых (в случае успеха), или же «вольную импровизацию» сети (в случае неудачи).

На стадии инициализации сети весовые коэффициенты синапсов устанавливаются следующим образом:

$$w_{ij} = \begin{cases} \sum_{k=1}^{m} x_i^k x_j^k, \; i \neq j, \\ 0, \quad i = j. \end{cases}$$

Здесь $i$ и $j$ – индексы, соответственно, предсинаптического и постсинаптического нейронов; $x_i^k$, $x_j^k$ – $i$ - й и $j$ - й элементы вектора $k$ -го образца.

Алгоритм функционирования сети следующий ($t$ – номер итерации):

***1.*** На входы сети подается неизвестный сигнал. Фактически его ввод осуществляется непосредственной установкой значений аксонов:

$$y_i(0) = x_i, \quad i = 1,2,...,n,$$

поэтому обозначение на схеме сети входных синапсов в явном виде носит чисто условный характер. Ноль в скобке справа от $y_i$ означает нулевую итерацию в цикле работы сети.

***2.*** Рассчитывается новое состояние нейронов

$$s_j(t+1) = \sum_{i=1}^{n} w_{ij} y_j(t), \quad j = 1,2,...,n,$$

и новые значения аксонов

$$y_j(t+1) = f\left(s_j(t+1)\right),$$

где $f$ – пороговая активационная функция с порогом $\theta = 0$ (см. табл. 2.1).

***3.*** Проверка, изменились ли выходные значения аксонов запоследнюю итерацию. Если изменились, то переход к пункту 2, иначе (если выходы застабилизировались) – *конец*. При этом выходной вектор представляет собой образец, наилучшим образом сочетающийся с входными данными.

Таким образом, когда подается новый вектор, сеть переходит из вершины в вершину, пока не стабилизируется. Устойчивая вершина определяется сетевыми весами и текущими входами. Если входной вектор частично неправилен или неполон, сеть стабилизируется в вершине, ближайшей к желаемой.

Показано, что достаточным условием устойчивой работы такой сети является выполнение условий

$$w_{ij} = w_{ji}, \quad w_{ii} = 0.$$

Как говорилось выше, иногда сеть не может провести распознавание и выдает на выходе несуществующий образ. Это связано с проблемой ограниченности возможностей сети. Для сети Хопфилда число запоминаемых образов $N$ не должно превышать величины, примерно равной $0.15n$. Кроме того, если два образа $A$ и $B$ сильно похожи, они, возможно, будут вызывать у сети перекрестные ассоциации, т.е. предъявление на входы сети вектора $A$ приведет к появлению на ее выходах вектора $B$ и наоборот.

Когда нет необходимости, чтобы сеть в явном виде выдавала образец, т.е. достаточно, скажем, получать номер образца, ассоциативную память успешно реализует сеть Хэмминга. Данная сеть характеризуется, по сравнению с сетью Хопфилда, меньшими затратами на память и объемом вычислений, что становится очевидным из ее структуры (рис. 5.20).

Сеть состоит из двух слоев. Первый и второй слои имеют по $m$ нейронов, где $m$ – число образцов. Нейроны первого слоя имеют по $n$ синапсов, соединенных с входами сети (образующими фиктивный нулевой слой). Нейроны второго слоя связаны между собой ингибиторными (отрицательными обратными) синаптическими связями. Единственный синапс с положительной обратной связью для каждого нейрона соединен с его же аксоном.

Идея работы сети состоит в нахождении расстояния Хэмминга от тестируемого образа до всех образцов (расстоянием Хэмминга называется число отличающихся битов в двух бинарных векторах). Сеть должна выбрать образец с минимальным расстоянием Хэмминга до неизвестного входного сигнала, в результате чего будет активизирован только один выход сети, соответствующий этому образцу.

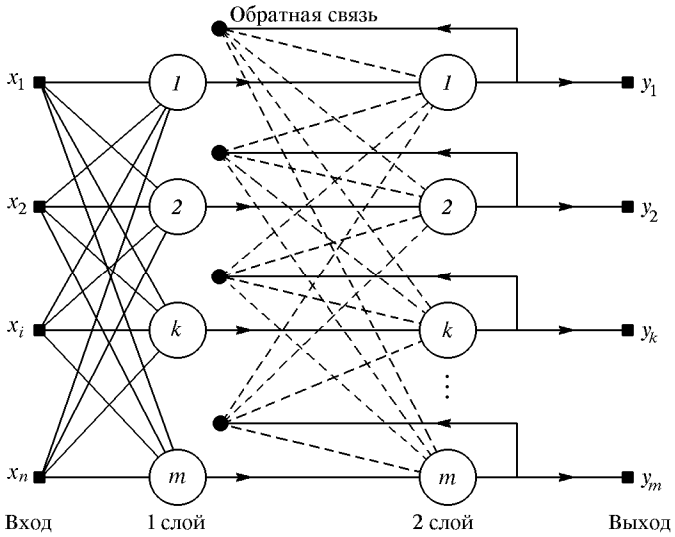



Рис. 5.20. Структурная схема сети Хэмминга.

На стадии инициализации весовым коэффициентам первого слоя и порогу активационной функции присваиваются следующие значения:

$$w_{ik} = \frac{x_i^k}{2}, \quad \theta_k = \frac{n}{2}, \quad i = 1,2,...,n, \quad k = 1,2,...,m.$$

Здесь $x_i^k$ – $i$ - й элемент $k$ - го образца.

Весовые коэффициенты тормозящих синапсов во втором слое берут равными некоторой величине $0 < \varepsilon < 1/m$. Синапс нейрона, связанный с его же аксоном, имеет вес $(+1)$.

Алгоритм функционирования сети Хэмминга следующий:

***1.*** На входы сети подается значение неизвестного вектора $X = \{x_i : i = 1,2,...,n\}$, исходя из которого рассчитываются состояния нейронов первого слоя (верхний индекс в скобках указывает номер слоя):

$$y_j^{(1)} = s_j^{(1)} = \sum_{i=1}^{n} w_{ij} x_i + \theta_j, \quad j = 1,2,...,m.$$

После этого полученными значениями инициализируются значения аксонов второго слоя:

$$y_j^{(2)} = y_j^{(1)}, \quad j = 1,2,...,m .$$

***2.*** Вычисляются новые состояния нейронов второго слоя:

$$s_j^{(2)}(t+1) = y_j(t) - \varepsilon \sum_{k=1}^{m} y_k^{(3)}(t), \quad k \neq j, \quad j = 1,2,...,m ,$$

и значения их аксонов:

$$y_j^{(2)}(t+1) = f\left(s_j^{(1)}(t+1)\right), \quad j = 1,2,...,m .$$

***3.*** Проверяется, изменились ли выходы нейронов второго слоя за последнюю итерацию. Если «да», то перейди к шагу 2. Иначе – *конец*.

Из оценки алгоритма видно, что роль первого слоя весьма условна: воспользовавшись один раз на шаге 1 значениями его весовых коэффициентов, сеть больше не обращается к нему, поэтому первый слой может быть вообще исключен из сети (заменен на матрицу весовых коэффициентов).

В заключение можно сделать следующее обобщение. Сети Хопфилда и Хэмминга позволяют просто и эффективно разрешить задачу воссоздания образов по неполной и искаженной информации. Невысокая емкость сетей (число запоминаемых образов) объясняется тем, что сети не просто запоминают образы, а позволяют проводить их обобщение например, с помощью сети Хэмминга возможна классификация по критерию максимального правдоподобия. Вместе с тем, легкость построения программных и аппаратных моделей делают эти сети привлекательными для многих применений.

***Сеть с радиальными базисными элементами (RBF).*** В общем случае под термином Radial Basis Function Network (сеть *RBF*) понимается двухслойная сеть без обратных связей, которая содержит скрытый слой радиально симметричных скрытых нейронов (шаблонный слой). Для того чтобы шаблонный слой был радиально-симметричным, необходимо выполнение следующих условий:

– наличие центра, представленного в виде вектора во входном пространстве; обычно этот вектор сохраняется в пространстве весов от входного слоя к слою шаблонов;

– наличие способа измерения расстояния входного вектора от центра; обычно это стандартное евклидово расстояние;

– наличие специальной функции прохождения от одного аргумента, которая определяет выходной сигнал нейрона путем отображения функции расстояния; обычно используется функция Гаусса

$$\varphi(s) = e^{-s^2} .$$

Другими словами, выходной сигнал шаблонного нейрона – это функция только от расстояния между входным вектором $X$ и сохраненным центром $C$:

$$f(X) = \varphi\left(\frac{\| X - C \|}{\sigma}\right).$$

Выходной слой сети является линейным, так что выходы сети определяются выражением

$$y_i = \sum_{i=1}^{K} w_{ij}\, \varphi\left(\frac{\| X - C_i \|}{\sigma_i}\right), \quad j = 1, 2, ..., m,$$

где $C_i$ – центры, $\sigma_i$ – отклонения радиальных элементов.

Обучение *RBF* - сети происходит в несколько этапов. Сначала определяются центры и отклонения для радиальных элементов; после этого оптимизируются параметры $w_{ij}$ линейного выходного слоя.

Расположение центров должно соответствовать кластерам, реально присутствующим в исходных данных. Рассмотрим два наиболее часто используемых метода.

*Выборка из выборки*. В качестве центров радиальных элементов берутся несколько случайно выбранных точек обучающего множества. В силу случайности выбора они «представляют» распределение обучающих данных в статистическом смысле. Однако, если число радиальных элементов невелико, такое представление может быть неудовлетворительным.

*Алгоритм K - средних*. Этот алгоритм стремится выбрать оптимальное множество точек, являющихся центроидами кластеров в обучающих данных. При $K$ радиальных элементах их центры располагаются таким образом, чтобы:

– каждая обучающая точка «относилась» к одному центру кластера и лежала к нему ближе, чем к любому другому центру;

– каждый центр кластера был центроидом множества обучающих точек, относящихся к этому кластеру.

После того как определено расположение центров, нужно найти отклонения. Величина отклонения (ее также называют сглаживающим фактором) определяет, насколько «острой» будет гауссова функция. Если эти функции выбраны слишком острыми, то сеть не будет интерполировать данные между известными точками, и потеряет способность к обобщению. Если же гауссовы функции взяты чересчур широкими, сеть не будет воспринимать мелкие детали. На самом деле сказанное – еще одна форма проявления дилеммы переобучения (или недообучения). Как правило, отклонения выбираются таким образом, чтобы «колпак» каж-

дой гауссовой функций захватывал несколько соседних центров. Для этого имеется несколько методов:

*Явный.* Отклонения задаются пользователем.

*Изотропный.* Отклонение берется одинаковым для всех элементов и определяется эвристически с учетом количества радиальных элементов и объема покрываемого пространства.

*K - ближайших соседей.* Отклонение каждого элемента устанавливается (индивидуально) равным среднему расстоянию до его $K$ ближайших соседей. Тем самым отклонения будут меньше в тех частях пространства, где точки расположены густо, – здесь будут хорошо учитываться детали, – а там, где точек мало, отклонения будут большими (и будет производиться интерполяция).

После того как выбраны центры и отклонения, параметры выходного слоя оптимизируются с помощью стандартного метода линейной оптимизации – алгоритма псевдообратных матриц (сингулярного разложения).

Могут быть построены различные гибридные разновидности сетей с радиальными базисными функциями. Например, выходной слой может иметь нелинейные функции активации, и тогда для его обучения используется какой-либо из алгоритмов обучения многослойных сетей, например метод обратного распространения. Можно также обучать радиальный (скрытый) слой с помощью алгоритма обучения сети Кохонена – это еще один способ разместить центры так, чтобы они отражали расположение данных.

Сети *RBF* имеют ряд *преимуществ* перед рассмотренными многослойными сетями прямого распространения (хотя их структура и соответствует приведенной на рис. 2.5). Во-первых, они моделируют произвольную нелинейную функцию с помощью всего одного промежуточного слоя, тем самым избавляя нас от необходимости решать вопрос о числе слоев. Во-вторых, параметры линейной комбинации в выходном слое можно полностью оптимизировать с помощью хорошо известных методов линейной оптимизации, которые работают быстро и не испытывают трудностей с локальными минимумами, так мешающими при обучении с использованием алгоритма обратного распространения ошибки. Поэтому сеть *RBF* обучается очень быстро (на порядок быстрее, чем с использованием алгоритма обратного распространения).

*Недостатки* сетей *RBF*: данные сети обладают плохими экстраполирующими свойствами и получаются весьма громоздкими при большой размерности вектора входов.

***Вероятностная нейронная сеть (PNN).*** Задача оценки плотности вероятности по имеющимся данным имеет давнюю историю в математической статистике.

Обычно при этом предполагается, что плотность имеет некоторый определенный вид (чаще всего, – что она имеет нормальное распределение). После этого оцениваются параметры модели. Нормальное распределение часто используется потому, что тогда параметры модели (среднее и стандартное отклонение) можно оценить аналитически.

Заметим, что предположение о нормальности далеко не всегда оправдано.

Другой подход к оценке плотности вероятности основан на так называемых *ядерных оценках*. Можно рассуждать так: тот факт, что наблюдение расположено в данной точке пространства, свидетельствует о том, что в этой точке имеется некоторая плотность вероятности. Кластеры из близко лежащих точек указывают на то, что в этом месте плотность вероятности большая. Вблизи наблюдения имеется большее доверие к уровню плотности, а по мере отдаления от него доверие убывает и стремится к нулю. В методе ядерных оценок в точке, соответствующей каждому наблюдению, помещается некоторая простая функция, затем все они складываются, и в результате получается оценка для общей плотности вероятности. Чаще всего в качестве ядерных функций берутся гауссовы функции (с формой колокола). Если обучающих примеров достаточное количество, то такой метод дает достаточно хорошее приближение к истинной плотности вероятности.

Метод аппроксимации плотности вероятности с помощью ядерных функций во многом похож на метод радиальных базисных функций, и таким образом мы естественно приходим к понятиям вероятностной нейронной сети (*PNN*) и обобщенно-регрессионной нейронной сети (*GRNN*). *PNN*- сети предназначены для задач классификации, а *GRNN* – для задач регрессии. Сети этих двух типов представляют собой реализацию методов ядерной аппроксимации, оформленных в виде нейронной сети.

Сеть *PNN* имеет, по меньшей мере, три слоя: входной, радиальный и выходной. Радиальные элементы берутся по одному на каждое обучающее наблюдение. Каждый из них представляет гауссову функцию с центром в этом наблюдении. Каждому классу соответствует один выходной элемент. Каждый такой элемент соединен со всеми радиальными элементами, относящимися к его классу, а со всеми остальными радиальными элементами он имеет нулевое соединение. Таким образом, выходной элемент просто складывает отклики всех элементов, принадлежащих к его классу. Значения выходных сигналов получаются пропорциональными ядерным оценкам вероятности принадлежности соответствующим классам, и, пронормировав их на единицу, мы получаем окончательные оценки вероятности принадлежности классам.

Выход рассматриваемой сети, соответствующий какому-либо классу, описывается выражением

$$y_i = \frac{1}{N\sigma^n}\sum_{i=1}^{N}\varphi\left(\frac{\| X - X^k \|}{\sigma}\right),$$

где $n$ – размерность входного вектора, $N$ – объем обучающей выборки, $X^k$ – элемент (вектор) этой выборки, соответствующий отмеченному классу.

Базовая модель *PNN*-сети может иметь две модификации.

Вероятностная нейронная сеть имеет единственный управляющий параметр обучения, значение которого должно выбираться пользователем, отклонение гауссовой функции $\sigma$ (параметр сглаживания). Как и в случае *RBF*-сетей, этот параметр выбирается из тех соображений, чтобы «шапки» определенное число раз перекрывались: выбор слишком маленьких отклонений приведет к «острым» аппроксимирующим функциям и неспособности сети к обобщению, а при слишком больших отклонениях будут теряться детали. Требуемое значение несложно найти опытным путем, подбирая его так, чтобы контрольная ошибка была как можно меньше. К счастью, *PNN*-сети не очень чувствительны к выбору параметра сглаживания.

Наиболее важные *преимущества PNN*-сетей состоят в том, что выходное значение имеет вероятностный смысл (и поэтому его легче интерпретировать), и в том, что сеть быстро обучается. При обучении такой сети время тратится практически только на то, чтобы подавать ей на вход обучающие наблюдения, и сеть работает настолько быстро, насколько это вообще возможно.

Существенным *недостатком* таких сетей является их объем. *PNN* – сеть фактически вмещает в себя все обучающие данные, что требует большой объем памяти и соответственно замедляет скорость работы.

*PNN*-сети особенно полезны при пробных экспериментах (например, когда нужно решить, какие из входных переменных использовать), так как благодаря короткому времени обучения можно быстро проделать большое количество пробных тестов.

***Обобщенно-регрессионная нейронная сеть (GRNN).*** Данная сеть устроена аналогично вероятностной нейронной сети (PNN), но она предназначена для решения задач регрессии, а не классификации. Как и в случае *PNN*-сети, в точку расположения каждого обучающего наблюдения помещается гауссова ядерная функция. Мы считаем, что каждое наблюдение свидетельствует о некоторой нашей уверенности в том, что поверхность отклика в данной точке имеет определенную высоту, и эта уверенность убывает при отходе в сторону от точки. *GRNN*-сеть копирует внутрь себя все обучающие наблюдения и использует их для

оценки отклика в произвольной точке. Окончательная выходная оценка сети получается как взвешенное среднее выходов по всем обучающим наблюдениям:

$$y = \frac{\sum_{i=1}^{K} y^k \, \varphi\left(\| X - X^k \| / \sigma\right)}{\sum_{i=1}^{K} \varphi\left(\| X - X^k \| / \sigma\right)}$$

где $X^k$, $y^k$ – точки обучающей выборки.

Первый промежуточный слой сети *GRNN* состоит из радиальных элементов. Второй промежуточный слой содержит элементы, которые помогают оценить взвешенное среднее. Каждый выход имеет в этом слое свой элемент, формирующий для него взвешенную сумму. Чтобы получить из взвешенной суммы взвешенное среднее, эту сумму нужно поделить на сумму весовых коэффициентов. Последнюю сумму вычисляет специальный элемент второго слоя. После этого в выходном слое производится собственно деление (с помощью специальных элементов «деления»). Таким образом, число элементов во втором промежуточном слое на единицу больше, чем в выходном слое. Как правило, в задачах регрессии требуется оценить одно выходное значение, и, соответственно, второй промежуточный слой содержит два элемента.

Можно модифицировать GRNN-сеть таким образом, чтобы радиальные элементы соответствовали не отдельным обучающим случаям, а их кластерам. Это уменьшает размеры сети и увеличивает скорость обучения. Центры для таких элементов можно выбирать с помощью любого предназначенного для этой цели алгоритма (выборки из выборки, $K$ - средних или Кохонена).

*Достоинства и недостатки* у сетей *GRNN* в основном такие же, как и у сетей *PNN*, – единственное различие в том, что *GRNN* используются в задачах регрессии, а *PNN* – в задачах классификации. *GRNN* -сеть обучается почти мгновенно, но может получиться большой и медленной (хотя здесь, в отличие от *PNN*, не обязательно иметь по одному радиальному элементу на каждый обучающий пример, их число все равно будет большим). Как и сеть *RBF*, сеть *GRNN* не обладает способностью экстраполировать данные.

***Линейные НС.*** Согласно общепринятому в науке принципу, если более сложная модель не дает лучших результатов, чем более простая, то из них следует предпочесть вторую. В терминах аппроксимации отображений самой простой моделью будет линейная, в которой аппроксимирующая (подгоночная) функция определяется гиперплоскостью. В задаче классификации гиперплоскость размещается таким образом, чтобы

она разделяла собой два класса (линейная дискриминантная функция); в задаче регрессии гиперплоскость должна проходить через заданные точки. Линейная модель обычно задается уравнением

$$Y = X W + B ,$$

где $W$ – матрица весов сети, $B$ – вектор смещений.

На языке нейронных сетей линейная модель представляется сетью без промежуточных слоев, которая в выходном слое содержит только линейные элементы (т.е. элементы с линейной функцией активацией). Веса соответствуют элементам матрицы, а пороги – компонентам вектора смещения. Во время работы сеть фактически умножает вектор входов на матрицу весов, а затем к полученному вектору прибавляет вектор смещения.

### *Эффективность нейронных сетей*

Эффективность нейронных сетей устанавливается рядом так называемых теорем о полноте. Ранее в нестрогой формулировке была приведена одна из них. Рассмотрим еще одну подобную теорему.

В 1989 г. Funahashi показал, что бесконечно большая нейронная сеть с единственным скрытым слоем способна аппроксимировать любую непрерывную функцию, сформулировав данное утверждение в форме следующей теоремы.

**Теорема**. Пусть $\psi(x)$ – непостоянная, ограниченная и монотонно возрастающая непрерывная функция. Пусть, далее, $U \subset R^n$ – ограниченное множество и

$$f : U \to R$$

– вещественная непрерывная функция, определенная на $U$ .

Тогда для произвольного $\varepsilon > 0$ существует целое $L$ и вещественные константы $w_i$ , $w_{ij}$ такие, что аппроксимация

$$\overline{f}(x_1, x_2, ..., x_n) = \sum_{i=1}^{L} w_i \, \psi \left( \sum_{j=1}^{n} w_{ij} x_j \right)$$

удовлетворяет неравенству

$$\| f - \overline{f} \|_\infty = \sup_{x \in U} | f(x) - \overline{f}(x) | \le \varepsilon .$$

Другими словами, любое непрерывное отображение может быть аппроксимировано в смысле однородной топологии на $U$ двухслойной нейронной сетью с активационными функциями $\psi(x)$ для нейронов скрытого слоя и линейными активационными функциями для нейронов выходного слоя. На рис. 5.21 представлена НС Funahashi для аппроксимации скалярной функции векторного аргумента.

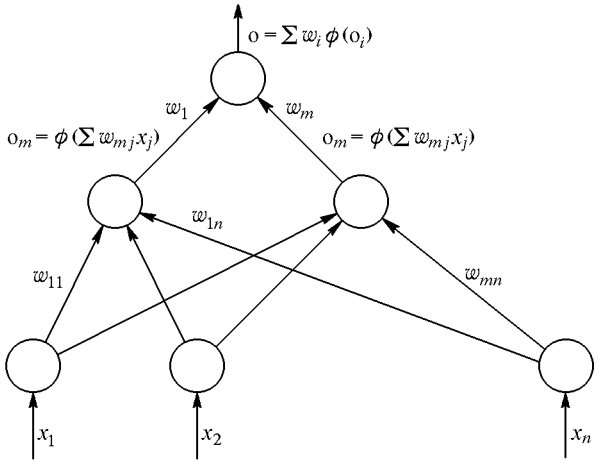



Рис. 5.21. Нейронная сеть Funahashi

Отметим еще раз, что приведенная теорема о полноте является далеко не единственной из известных.

*Основными недостатками* аппарата нейронных сетей являются:

– отсутствие строгой теории по выбору структуры НС;

– практическая невозможность извлечения приобретенных знаний из обученной НС (нейронная сеть практически всегда – «вещь в себе», черный ящик для исследователя).

# ЛИТЕРАТУРА


1. *Барский А. Б.* Нейронные сети: распознавание, управление, принятие решений. — М.: Финансы и статистика, 2004. — 176 с: ил. — (Прикладные информационные технологии).

2. *Галушкин А.И.* Теория нейронных сетей / Сер. Нейрокомпьютеры и их применение. – М.: Изд-во Радиотехника. –2000. – 480 с.

3. *Девятков В.В.* Системы искусственного интеллекта: Учеб. пособие для вузов. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. –352 с. (Серия Информатика в техническом университете).

4. Искусственный интеллект: В 3 кн. Справочник / Под ред. Э.В. Попова. – М.: Радио и связь, 1990. – 464 с.

5. *Калан Р.* Основные концепции нейронных сетей.: Пер . с англ. – М.: Издательский дом «Вильямс», 2001. – 290 с.

6. *Конышева Л.К., Назаров Д.М.* Основы теории нечетких множеств: Учебное пособие. – СПб.: Питер, 2011. – 192 с.

7. *Кофман А.* Введение в теорию нечетких множеств: Пер. с франц. – М.: Радио и связь, 1982. – 432 с.

8. *Люггер, Дж. Ф.* Искусственный интеллект: стратегии и методы решения сложных проблем. Изд 4-е.: Пер. с англ. – М.: Издательский дом «Вильямс», 2003. – 864 с.

9. Нейронные сети. STATISTICA Neural Networks: Методология и технологии современного анализа данных/ Под ред. В.П. Боровикова. 2-е изд., перераб. и доп. – М.: Горячая линия-Телеком, 2008. –392 с.

10. *Рассел С., Норвиг П.* Искусственный интеллект: современный подход, 2-е изд.: Пер. с англ. - М.: Издательский дом «Вильямс», 2006. - 1408 с.

11. *Рутковская Д., Пилиньский М., Рутковский Л.* Нейронные сети, генетические алгоритмы и нечеткие системы; Пер. с польск. И.Д. Рудинского. – М.: Горячая линия – Телеком, 2006. – 452 с.

12. *Смолин Д.В.* Введение в искусственный интеллект: конспект лекций. – М.: ФИЗМАТЛИТ, 2004. – 208 с.

13. *Хайкин С.* Нейронные сеи: полный курс, 2-е издание: Пер. с англ. – М.: Издательский дом «Вильямс», 2006. – 1104 с.

14. *Шамис А.Л.* Пути моделирования мышления: Активные синергетические сети, мышление и творчество, формальные модели поведения и «распознавание с пониманием». – М.: КомКнига, 2006. –336 с.

15. *Яхъяева Г.Э.* Нечеткие множества и нейронные сети: Учебное пособие. – М.: Интернет-Университет Информационных Технологий, БИНОМ, Лаборатория знаний, 2006. –316 с. – (Серия «Основы информационных технологий»).

Навчальне видання

***НЕФЬОДОВ ЮРІЙ МИХАЙЛОВИЧ***

# ОСНОВИ МОДЕЛЮВАННЯ ИНТЕЛЕКТУАЛЬНИХ СИСТЕМ

(російською мовою)

Оригінал-макет автора

Формат 60х84 1/16. Гарнітура Times.
Умов. друк. арк. 14,82. Обл. вид. арк. 15,6

Видавництво
Східноукраїнського національного університету
імені Володимира Даля
91034, м. Луганськ, кв. Молодіжний, 20а

**Адрес издательства:** 91034, г. Луганск, кв. Молодежный, 20а
**Телефон:** 8 (0642) 41-34-12, **факс**. 8 (0642) 41-31-60
E-mail uni@snu.edu.ua http: www.snu.edu.ua