МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФИЛИАЛ ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «МЭИ» В Г. СМОЛЕНСКЕ

В.В. БОРИСОВ, А.В. БОБРЯКОВ, А.Е. МИСНИК

ЭКСПЕРТНЫЕ СИСТЕМЫ

Учебное пособие

Смоленск 2021

УДК 004.89
Утверждено учебно-методическим Советом филиала НИУ МЭИ в г. Смоленске в качестве учебно-теоретического пособия для студентов, по направлению 09.03.01 «Информатика и вычислительная техника» (уровень подготовки – бакалавриат).

Подготовлено на кафедре вычислительной техники

Рецензенты
докт. техн. наук, профессор Военной академии ВПВО ВС РФ **В.О. Волосенков**
канд. техн. наук, доцент филиала НИУ МЭИ в г. Смоленске **А.В. Полячков**

Б 82 **Борисов В.В., Бобряков А.В., Мисник А.Е.** Экспертные системы. Учебное пособие по направлению «Информатика и вычислительная техника» [Текст]: учебное пособие. – Смоленск: Универсум, 2021. – 110 с.

ISBN 978-5-91412-464-6

Содержатся учебные и методические материалы, соответствующие образовательной программе общенаучного цикла по дисциплине «Интеллектуальные системы», а также специального цикла по дисциплинам «Теория принятия решений», «Системы искусственного интеллекта», обучающихся по направлению «Информатика и вычислительная техника» (уровень подготовки – бакалавриат). Пособие может быть полезно и для студентов других специальностей, занимающихся созданием и использованием интеллектуальных информационных систем и технологий.

ISBN 978-5-91412-464-6

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Учебное пособие «Экспертные системы» посвящено основам проектирования и разработки экспертных систем. В пособии содержатся учебные и методические материалы, соответствующие образовательной программе общенаучного цикла по дисциплине «Интеллектуальные системы», а также специального цикла по дисциплинам «Теория принятия решений», «Системы искусственного интеллекта», обучающихся по направлению «Информатика и вычислительная техника» (уровень подготовки – бакалавриат). Пособие может быть полезно и для студентов других специальностей, занимающихся созданием и использованием интеллектуальных информационных систем и технологий.

В данном пособии рассмотрены особенности экспертных систем, представлена технология их проектирования, описаны подходы к работе с экспертами в процессе разработки.

Рассмотрены различные технологии, применяющиеся для разработки экспертных систем: таблицы решений, продукционные правила, семантические сети, фреймы, байесовские сети доверия. Описаны интеллектуальные технологии извлечения новых знаний: системы символьного обучения, эволюционное программирование, эволюционные стратегии и генетические алгоритмы. Особое внимание в пособии обращено на технологии разработки искусственных нейронных сетей и их применению в экспертных системах.

1 Системы и их модели

1.1 Понятие и определения системы

Разнообразие определений системы обусловлено:

- различными концепциями теории систем и вариантами системного подхода, отличающимися по составу и содержанию понятий и принципов;
- ориентацией, как на разные типы, так и на различное назначение систем.

Можно выделить 5 групп определений системы.

Первую группу составляют определения системы в виде выбираемой исследователем совокупности переменных, свойств или сущностей. В этом случае системой могут оказаться произвольно выбранные объекты, не имеющие или имеющие слабые взаимосвязи.

Вторая группа определений основана на понимании системы как множества связанных между собой элементов. Однако это определение системы допускает возможность различных, в том числе произвольных, ее разбиений на подмножества элементов, каждое из которых также является множеством.

Третью группу составляют наиболее общие определения системы как комплекса элементов, находящихся во взаимодействии. Однако в этом случае к категории «система» могут быть отнесены любые, даже очень слабо взаимодействующие объекты, которые рассматриваются с системных позиций.

Четвертую группу составляют определения системы, связывающие ее с целенаправленной активностью. Система определяется как относительно устойчивая организованная совокупность взаимодействующих и взаимосвязанных элементов, а также комплекс средств достижения общей цели. Развитие и совершенствование такой системы зависит от взаимодействия с окружающей средой.

Пятая группа определений характеризует систему через указание признаков, которыми должен обладать объект, чтобы его можно было отнести к категории «система». Данные признаки вводятся через понятия совокупности, взаимосвязи и целого. Здесь под системой понимается совокупность элементов и процессов, находящихся в отношениях и взаимосвязях между собой, образующих единое целое и характеризующихся интегративным, или системным, свойством, отличающим данную совокупность от среды и приобщающим к этому свойству каждый из ее компонентов.

1.2 Классификация систем

По происхождению системы классифицируются делятся на:

- естественные (существующие в объективной действительности: живые; неживые экологические, социальные и др.);
- концептуальные (продукт человеческого мышления: теории, гипотезы);
- искусственные (созданные человеком: механизмы, машины, автоматы);

- смешанные, т.е. объединяющие искусственные и естественные подсистемы: автоматизированные, организационно-технические (в которых совместно функционируют человеческие коллективы и техника).

По целевому предназначению системы делятся на:

- ценностноориентированные, которым не присуща внутренняя цель, целевая функция задается извне, а процесс функционирования оценивается по некоторым критериям ценностей;
- целеориентированные, которым присуща внутренняя цель, а основой функционирования и развития являются факторы целесообразности и целеполагания.

По типу переменных системы могут быть:

- с количественным описанием переменных, допускающим их дискретное и непрерывное описание, а также смешанный случай;
- с качественным описанием переменных, допускающим представление как средствами естественного языка, так и на основе формализации;
- с качественным описанием переменных.

Эти описания могут носить детерминированный, стохастический, нечеткий или смешанный характер.

По типу управления системы можно разделить на:

- управляемые извне;
- самоуправляемые;
- управление которыми частично осуществляется извне, частично – изнутри.

По способу управления бывают:

- без обратной связи (программным управлением);
- регулированием (автоматическим регулированием);
- управлением по параметрам (параметрической адаптацией);
- управлением по структуре (структурной адаптацией).

По ресурсной обеспеченности моделирования системы делятся на малые и большие. Под большими понимаются системы, моделирование которых затруднительно вследствие их размерности, а также наличия компонентов, не поддающихся точному и подробному описанию.

По достаточности информации для моделирования системы можно разделить на простые и сложные. Сложной системой называется система, состоящая из подсистем, являющихся, в свою очередь, простыми системами, и в модели которой не хватает информации для эффективного управления. Степень сложности больше зависит от разнообразия связей и элементов, чем от их количества. При этом сложной является система, обладающая определенным набором свойств, например, большим числом неоднородных элементов, иерархией, агрегированием параметров, многофункциональностью, гибкостью, адаптацией, надежностью, безопасностью, стойкостью и т.д.

1.3 Статические модели систем

Модель типа *«черный ящик»* имеет следующие особенностями:

- отражает свойства целостности системы и ее обособленности от внешней среды;
- предполагается, что моделируемый процесс может быть «аппроксимирован»;
- структура модели не отражает структуру реальной системы;
- источником информации для построения модели являются данные типа «вход–выход» или «стимул–реакция»;
- неизвестно, какие из параметров являются значимыми и какой моделью описываются закономерности их влияния на целевую характеристику;
- выбор параметров выполняется проверкой эмпирических гипотез на разнокачественных данных.

Ограничениями этой модели являются:

- опасность неполноты перечня входов и выходов, вследствие чего важные из них могут быть сочтены несущественными либо быть неизвестны;
- структуру и параметры этих моделей сложно интерпретировать; они не могут быть промасштабированы при изменении масштаба исследуемого процесса.

Модель *состава системы* характеризуется следующими особенностями:

- детализация моделируемой системы на неделимые элементы;
- границы разбиения модели состава системы на подсистемы и элементы.

Модель *структуры системы* характеризуется тем, что она отображает отношения (причинности, подобия, сходства, включения, подчиненности и др.) между компонентами модели состава системы.

Модель типа *«белый ящик»* (или структурная схема системы):

- образуется из моделей «черного ящика», состава и структуры системы;
- структурная схема системы является наиболее подробной и полной моделью системы и содержит все элементы системы, все связи между элементами внутри системы и связи определенных элементов с окружающей средой (входы и выходы системы);
- модель может быть представлена графически, в виде теоретико-множественных описаний, в виде матриц, графов, с помощью языков моделирования структур.

1.4 Динамические модели систем

Для *динамической модели типа «черный ящик»* выход $y(t)$ является реакцией на управляемые $u(t)$ и неуправляемые $v(t)$ входы $x(t) = \{u(t), v(t)\}$, выражаемой совокупностью двух процессов:

$$X^T = \{x(t)\} \text{ и } Y^T = \{y(t)\}, t \in T.$$

Динамическая модель типа «черный ящик» предполагает восстановление преобразования F, при котором $y(t) = F(x(t))$. Нахождение F может быть как самостоятельной задачей, так и являться задачей перехода от модели «черного ящика» к *модели «белого ящика» в динамике*. Этапы перехода:

- модель типа «черного ящика» – связь между входными и выходными параметрами считается вообще неизвестной;
- непараметризированная модель – информация о связи входных и выходных параметрах является априорной и настолько общей, что нельзя сделать конкретных выводов о функциональном виде этой зависимости;
- параметризированная модель – в явной форме известна зависимость $y(t) = F(x(t))$ с точностью до конечного числа параметров $F = (F1, \ldots, Fk)$;
- структурная схема системы в динамике, в которой параметры системы заданы точно (например, сетевыми графиками).

Сложности построения динамических моделей:

- в параметризированном случае неизвестны параметры функции F;
- в непараметризированном – вид функции F неизвестен;
- неизвестны сведения о свойствах функции F (непрерывности, гладкости, монотонности, симметричности и др.);
- входы и выходы наблюдаются с помехами или искажениями;
- необходимо учитывать состояние системы в предыдущие моменты;
- система функционирует в условиях неопределенности.

Наибольшая общность динамической модели системы достигается введением понятия *состояния системы*, как внутренней характеристики системы. Выход системы зависит от ее состояния с течением времени. То есть, существует такое отображение

$$\eta: S \times T \rightarrow Y,$$

что $y(t) = \eta\big(t, s(t)\big), t \in T$, где $s(t)$ – состояние системы в момент t.

Таким образом, наиболее общая динамическая модель системы задает множества входов, состояний и выходов, а также связи между ними:

$$X \overset{\sigma}{\rightarrow} S \overset{\eta}{\rightarrow} Y.$$

Конкретизируя множества X, Y и S и отображения σ и η, можно перейти к примерам моделей различных систем.

В зависимости от того, дискретно или непрерывно множество T, модели являются *дискретными* и *непрерывными по времени*.

Если же множества X, Y и S дискретной во времени системы имеют конечное число элементов, то такую модель называют конечным автоматом.

Если множества X, Y и S – линейные пространства, а σ и η – линейные операторы, то и модель системы называется линейной.

Если для пространств линейной системы заданы топологические структуры (определена метрика и сходимость последовательностей), а σ и η непрерывны в этой топологии, то модели являются гладкими.

Если свойства систем не меняются со временем, то системы и их модели называют стационарными. Стационарность означает независимость функции η от t и инвариантность функции σ к сдвигу во времени.

Вопросы для самопроверки

1. Сформулируйте определение системы.
2. Назовите основные группы определений системы.
3. Какие системы называют смешанными?
4. Назовите основные причины разнообразий определений системы.
5. Какие существуют группы классификации систем?
6. Что такое целеориентированные системы?
7. Какую систему можно назвать сложной?
8. Какими свойствами может обладать сложная система?
9. Назовите особенности модели типа «черный ящик».
10. Какие ограничения модели типа «черный ящик» существуют?
11. Опишите модель системы типа «белый ящик».
12. Как называются системы, свойства которых не меняются со временем?

2 Экспертные системы

2.1 Данные и знания

Данные – отдельные факты, характеризующие объекты, процессы и явления в предметной области, а также их свойства.

Знания – выявленные закономерности предметной области (принципы, связи, законы), позволяющие решать задачи в этой области.

Экспертная система – система, предназначенная для решения трудно формализуемых задач, у которых отсутствует алгоритм решения, алгоритм решения которых не известен; или обладает достаточно большой размерностью.

Ключевая особенность экспертных систем – включает в себя базу знаний с набором правил и позволяет на основании предоставляемых пользователем фактов распознать ситуацию, поставить диагноз, сформулировать решение или дать рекомендацию для выбора действия. *Отличие экспертных систем от систем математического моделирования* – моделирует механизм рассуждения применительно к решению задач в конкретной проблемной области.

Цель разработки экспертной системы – разработка программной системы (средств) для решения трудно формализуемых задач, не уступающих по качеству и эффективности решениям, которые принимает эксперт.

2.2 Выбор: экспертная система или эксперт?

Экспертная система будет делать не более (а скорее даже менее) того, чем может эксперт, создавший данную систему. Можно построить самообучающуюся экспертную систему в области, в которой вообще нет экспертов, либо объединить в одной экспертной системе знания нескольких экспертов, и получить в результате систему, которая может то, чего ни один из ее создателей не может.

Преимущества экспертных систем

- простота «транслирования» данных и знаний;
- устойчивость и воспроизводимость результатов;
- достаточно низкая затратность создания.

Разрабатывать экспертную систему целесообразно, если:

- не могут быть построены строгие алгоритмы или процедуры, но существуют эвристические методы решения;
- есть эксперты, которые способны решить задачу;
- по своему характеру задачи относятся к области диагностики, интерпретации или прогнозирования.

Разрабатывать экспертную систему нецелесообразно, если:

- имеются эффективные алгоритмические методы;
- отсутствуют эксперты или их число недостаточно;
- задачи носят вычислительный характер;
- задачи решаются процедурными методами, с помощью аналогии или интуитивно.

2.3 Назначение экспертных систем

Интерпретация – процесс определения смысла данных, результаты которого должны быть согласованными и корректными.

Планирование – заранее намеченный порядок, последовательность осуществления какой-либо программы, работы, проведения мероприятий.

Прогнозирование – обоснованное описание последовательности событий, с возможностью обнаружения новых факторов.

Мониторинг – непрерывное оповещение о состоянии системы, приложения или процесса.

Проектирование – процесс создания новой информации об объекте, системе (имеется возможность исключения профессионала из процесса проектирования).

Диагностика – процесс распознавания состояния на основе имеющихся факторов

Обучение – обучение пользователя, а также самообучение системы, как на этапе приобретения знаний, так и в процессе работы ЭС (пополнение базы знаний (БЗ) ЭС новыми цепочками вывода).

Может быть предложена следующая классификация экспертных систем:

- в зависимости от масштаба времени (реального времени, псевдореального времени, статические);
- по виду знаний (детерминированные, неопределённые).
- по источнику знаний (один, несколько).

Возможны следующие режимы функционирования экспертной системы:

- приобретение знаний;
- консультации;
- комбинированный.

Вопросы для самопроверки

1. Сформулируйте определение экспертной системы.
2. Перечислите ключевые особенности экспертных систем.
3. Что является основной целю разработки экспертной системы?
4. Перечислите преимущества экспертных систем.
5. Какие существуют условия, при которых нецелесообразно разрабатывать экспертную систему?
6. Что такое интерпретация?
7. Что такое мониторинг?
8. Что такое проектирование?
9. Перечислите группы экспертных систем по масштабу времени.
10. Перечислите режимы функционирования экспертной системы.

3 Технология разработки экспертных систем

3.1 Этапы создания экспертных систем

Как правило, при создании экспертных систем реализуются следующие этапы:

- идентификация;
- концептуализация;
- формализация;
- выполнение;
- тестирование;
- опытная эксплуатация.

На рисунке 3.1 проиллюстрирована взаимосвязь этих этапов.

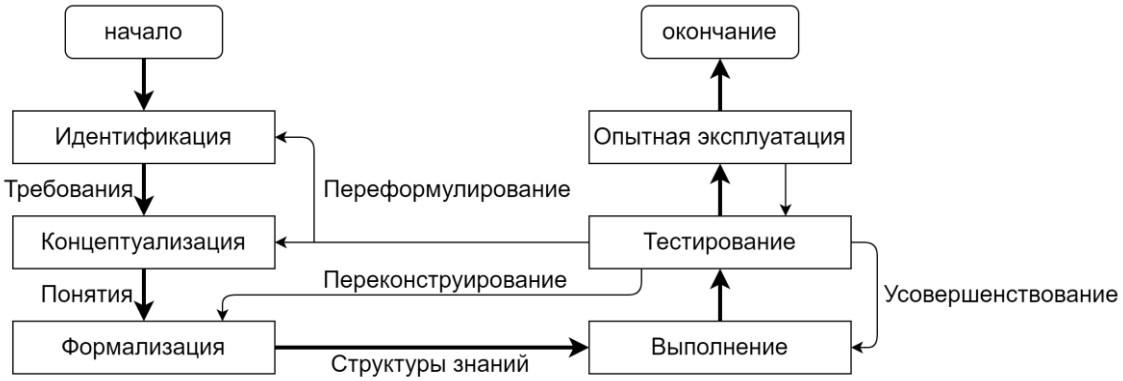


Рисунок 3.1 – Взаимосвязь этапов разработки экспертных систем

Рассмотрим кратко перечисленные этапы создания экспертных систем.

3.2 Идентификация

На этапе идентификации экспертной системы определяются:

- участники процесса разработки;
- задачи;
- ресурсы;
- цели.

Идентификация участников и их ролей представляет собой определение количества экспертов и инженеров по знаниям, а также формы их взаимоотношений. Обычно в основном цикле разработки экспертной системы используются следующие роли: эксперт; инженер по знаниям; программист.

Идентификация задачи заключается в составлении вербального описания решаемой задачи. Указываются: общие характеристики задачи; подзадачи, выделяемые внутри данной задачи; ключевые понятия, характеристики и отношения; входные и выходные данные; предположительный вид решения; знания, релевантные решаемой задаче; примеры решения задачи.

Цель идентификации задачи – охарактеризовать задачу и структуру поддерживающих ее знаний и приступить к работе по созданию базы знаний.

Если исходная задача оказывается слишком сложной с учетом имеющихся ресурсов, то этап идентификации может потребовать нескольких итераций.

В процессе идентификации задачи необходимо ответить на ряд вопросов:

- Какие задачи предлагается решать экспертной системе?
- Как эти задачи могут быть охарактеризованы и определены?
- На какие подзадачи разбивается задача, какие данные используются?
- Какие ситуации препятствуют решению?
- Как эти препятствия будут влиять на экспертную систему?

Начальное описание задачи экспертом влечет за собой вопросы инженера по знаниям для уточнения терминов и ключевых понятий. Уточняется описание задачи, объясняется, как ее решить, какие рассуждения лежат в основе решения.

После нескольких циклов, уточняющих описание, эксперт и инженер по знаниям получают окончательное неформальное описание задачи.

Идентификация ресурсов состоит из определения:

- источников знаний;
- времени разработки;
- вычислительных средств;
- объема финансирования.

Идентификация целей – это формулирование в явном виде целей построения экспертной системы. При этом важно отличать цели, ради которых строится система, от задач, которые она должна решать.

Примеры целей экспертных систем:

- формализация неформальных знаний экспертов;
- улучшение качества решений, принимаемых экспертом;
- автоматизация рутинных аспектов работы эксперта;
- тиражирование знаний эксперта.

В завершении этапа идентификации инженер по знаниям должен ответить на основной вопрос: *«Подходят ли методы инженерии знаний для решения предложенной задачи?»*. Для положительного ответа на данный вопрос необходимо, чтобы задача относилась к достаточно узкой, специальной области знаний и не требовала для своего решения использования того, что принято называть здравым смыслом, поскольку методы искусственного интеллекта не дают возможности формализовать это понятие.

3.3 Концептуализация

На этапе концептуализации эксперт и инженер по знаниям выделяют:

- ключевые понятия;
- отношения и характеристики, необходимые для описания процесса решения задачи.

Затем выполняется детализация:

- типов доступных данных;
- исходных и выводимых данных;
- подзадач общей задачи;
- используемых стратегий и гипотез;
- видов взаимосвязей между объектами проблемной области;
- типов используемых отношений (иерархия, причина/следствие, часть/целое и т.п.);
- процессов, используемых в ходе решения задачи;
- типовых ограничений, накладываемых на процессы и используемых в ходе решения;
- состава знаний, используемых для решения.

Для определения перечисленных характеристик задачи целесообразно составить детальный *протокол действий и рассуждений эксперта* в процессе решения хотя бы одной конкретной задачи. Такой протокол обеспечивает инженера по знаниям *словарем терминов* (объектов) и некоторым приблизительным *представлением о тех стратегиях, которые использует эксперт*.

Адекватным средством для выделения ключевых понятий, отношений и характеристик являются *диаграммы*. Диаграммы используются как средства проектирования, сопровождения и документирования, а также для организации взаимодействия между различными участниками процесса создания системы.

Видами таких диаграмм являются:

- контекстные диаграммы (структурно-функциональные схемы), например, нотация IDEF0;
- диаграммы «сущность-связь», например, нотация IDEF1X;
- диаграммы потоков данных, например, нотация DFD;
- диаграммы «состояния-переходы», например, нотация UML.

Контекстные диаграммы – описывают систему на высоком уровне. Ими представляется сама система в виде системного процесса, ее основные части (подсистемы), включая операторы и основные блоки оборудования (измерения и управления), объекты внешнего окружения и основные потоки между ними.

Диаграммы «сущность-связь» используются для систем со сложными связями между объектами, где важно более детально представлять взаимоотношения между объектами.

Диаграммы потоков данных показывают, как система взаимодействует с внешним окружением. Ими представляются внешние объекты, хранилища данных в системе, потоки данных, входящие, выходящие и проходящие внутри системы, и системные процессы, обрабатывающие эти потоки.

*Диаграммы «состояний-переходов» д*емонстрируют динамическое поведение системы, какие события происходят в системе, как система на них реагирует и в какие состояния она попадает.

3.4 Формализация

Все ключевые понятия и отношения, выявленные на этапе концептуализации, выражаются на некотором формальном языке, выбранном инженером по знаниям. Он определяет, подходят ли имеющиеся инструментальные средства для решения рассматриваемой проблемы или необходим выбор другого инструментария, или требуются оригинальные разработки.

Основные задачи формализации:

- структуризация общей задачи на связанные подзадачи;
- структуризация предметной области на основе иерархии классов;
- структуризация знаний на декларативные и процедурные;
- структуризация приложения на основе иерархии «часть/целое».

Структуризация общей задачи на связанные подзадачи

Разбиение приложения на модули существенно ускоряет разработку (так как независимые группы разработчиков могут одновременно разрабатывать различные модули), снижает затраты на сопровождение и поддержку, упрощает повторное использование модулей базы знаний в последующих разработках.

Разбиение прикладной экспертной системы на модули несколько повышает накладные расходы на загрузку и сборку прикладной системы, например, восстановление после сбоев и перезапуск системы.

Структуризация предметной области на основе иерархии классов

Применение объектно-ориентированного подхода в современных экспертных системах естественным образом реализует возможность декомпозиции задачи на совокупность подзадач. Знания при этом подходе организованы в классы. Каждый класс определяется специфическим набором атрибутов. Классы организуются в иерархию классов. Каждый класс в иерархии наследует атрибуты и ограничения своего родительского класса. Обычно производный класс определяет дополнительные специфические атрибуты и (или) ограничения.

Основными механизмами структурирования проблемно-ориентированной иерархии классов являются два противоположно направленных, но взаимосвязанных процесса: обобщение и специализация (конкретизация).

Обобщение – создание родительских классов для обобщения свойств, присущих более чем одному классу объектов в приложении.

Специализация заключается во введении новых классов для описания объектов, отличающихся значениями характеристик, их набором и поведением от уже описанных.

Структуризация знаний на декларативные и процедурные

Декларативные знания – записаны в памяти интеллектуальной системы так, что они непосредственно доступны для использования после обращения к соответствующему полю памяти. Обычно декларативные знания используются для представления информации о свойствах и фактах предметной области.

Процедурные знания – это знания, хранящиеся в памяти интеллектуальной системы в виде описания процедур, с помощью которых их можно получить. Обычно процедурные знания используются для представления информации о способах решения задач в проблемной области, а также различные инструкции, методики и т.п.

Структуризация приложения на основе иерархии «часть/целое»

Модульный принцип создания приложения предоставляет разработчику различные возможности разбиения приложения на подсистемы, легче поддающиеся сопровождению и модификации. Разбиение приложения на модули упрощает процесс тестирования за счет использования групповой работы над тестируемой системой. Модульность также обеспечивает базовые возможности для повторного использования фрагментов системы.

3.5 Реализация

По результатам этапов тестирования и опытной эксплуатации создается конечный продукт, пригодный для промышленного использования. Разработка прототипа состоит в программной реализации его компонентов и наполнении базы знаний.

Реализация – частая ошибка

Процесс приобретения знаний откладывают до полного понимания структуры базы знаний и всех тестовых примеров. Тем самым эта наиболее трудоемкая часть работы отодвигается на поздние этапы.

Процесс накопления знаний позволяет уточнить используемые понятия и отношения, поэтому необходимо начинать приобретение знаний, как только составлены или выбраны инструментальные средства, позволяющие работать с простейшим представлением знаний и простейшими управляющими структурами. Такой подход позволяет как можно раньше начать выполнение отдельных подзадач и обнаружить, что в ряде случаев для их решения необходимы дополнительные знания.

Реализация – первый прототип

Прототип должен обеспечить проверку адекватности идей, выбранных при построении данной экспертной системы, решаемым задачам. Цель состоит в том, чтобы получить решение задачи, пока не заботясь об эффективности.

Создание первого прототипа должно подтвердить, что выбранные методы решений и способы представления пригодны для успешного решения по крайней мере ряда задач из области экспертизы.

При разработке первого прототипа обычно оставляют в стороне вопросы, требующие значительных трудозатрат:

- построение сложных моделей;
- учет сложных временных, причинных и модальных отношений;
- понимание намерений пользователей;
- моделирование рассуждений, содержащих неточные понятия.

Реализация – приобретение знаний

В ходе приобретения знаний инженер по знаниям должен *получить* знания от эксперта, *структурировать* их и *представить* в виде, понятном экспертной системе. Процесс извлечения знаний сложен и длителен, так как эксперт часто или не осознает, какими знаниями он пользуется, или не может их вербализовать (содержательно выразить).

Реализация – структурирование знаний

Наиболее важным средством для структурирования знаний является иерархия классов, описывающих понятия промежуточного уровня. Во многих случаях эти понятия могут явно не упоминаться (а возможно, и не осознаваться) экспертом.

Задача инженера по знаниям – выделить такие понятия, обнаружив сходные действия эксперта при обработке различных ситуаций.

Представление правил в виде, понятном экспертной системе.

Особое внимание следует уделять трем ситуациям:

- некоторое правило слишком громоздко;
- имеется много похожих правил;
- используются частные, а не общие правила.

Громоздкость правила

Может объясняться тем, что в нем отражено несколько фактов из данной проблемной области.

Если это так, то правило надо разбить на несколько более мелких.

Имеется много похожих правил

Имеет место тогда, когда в проблемной области существует понятие, явно не указанное экспертом, а возможно, и не имеющее имени. В этом случае новое понятие необходимо ввести в явном виде, присвоить ему специальное имя и, используя это понятие, сформулировать одно правило взамен группы подобных.

Используются частные, а не общие правила

Имеет место тогда, когда эксперт не использует возможности, предоставляемые объектно-ориентированным программированием, позволяющим скрыть специфику объектов в иерархии классов и ссылаться в правилах на классы, а не на конкретные объекты.

Реализация – итеративный процесс

Итеративная разработка заключается в подходе к реализации системы как серии удачных приближений прототипов к конечной цели, а не как к единой, монолитной, интегрированной системе. Итеративная разработка особенно эффективна при создании систем с недостаточно четко определенными спецификациями, к которым прежде всего относятся экспертные системы.

Поскольку подобные проекты обычно недостаточно проработаны с точки зрения системного анализа, разработчики обычно обнаруживают новые требования к системе после начала проекта.

Если принят итеративный подход к разработке, то на адаптацию системы и коррекцию дальнейшего плана работ требуются относительно небольшие затраты.

3.6 Тестирование

Тестирование включается в каждую стадию прототипирования прикладной системы. Обычно тестирование рассматривают в качестве заключительной фазы процесса разработки, однако операционное прототипирование, характеризующееся возможностью изменения целей проектирования в процессе разработки, предъявляет особые требования к *доказательству корректности (верификации – verification)* и *соответствия разрабатываемой системы предъявляемым требованиям (концептуальное тестирование – validation).*

Верификация и концептуальное тестирование должны выполняться параллельно с процессом разработки экспертной системы.

Можно интерпретировать процесс верификации (логического тестирования) как альфа-тестирование программной системы, а концептуальное тестирование – как этап бета-тестирования. Для тестирования экспертной системы необходимо привлекать эксперта в данной предметной области.

Три аспекта тестирования экспертных систем:

- тестирование исходных данных;
- логическое тестирование базы знаний;
- концептуальное тестирование прикладной системы.

Тестирование исходных данных включает в себя проверку фактографической информации, служащей основой для проведения экспертизы. Очевидно, что наборы данных, используемых при тестировании, должны покрывать область возможных ситуаций, распознаваемых экспертной системой.

Логическое тестирование базы знаний заключается в обнаружении логических ошибок в системе продукций, не зависящих от предметной области:

- избыточные, циклические и конфликтные правила;
- пропущенные и пересекающиеся правила;
- несогласуемые и терминальные клаузы (несогласуемые условия).

Формальный характер этих ошибок позволяет автоматизировать процесс логического тестирования. Однако в ряде случаев, когда цепочки правил, используемых в процессе вывода, небольшие (от 3 до 10 правил), целесообразно проводить процесс верификации вручную.

Концептуальное тестирование проводится для проверки общей структуры системы и учета в ней всех аспектов решаемой задачи. На этом этапе проведение тестирования невозможно без привлечения конечных пользователей прикладной системы.

3.7 Опытная эксплуатация и внедрение

Проверяется пригодность экспертной системы для конечного пользователя. Здесь система занимается решением всех возможных задач при работе с различными пользователями. Целесообразно организовать работу системы не на стенде разработчика, а на месте работы пользователей. К этому этапу следует переходить лишь после того, как система, по мнению эксперта, будет успешно решать все требуемые задачи, чтобы ошибки в решениях не создавали у пользователя отрицательное представление о системе. Пригодность системы для пользователя определяется в основном удобством работы с ней и ее полезностью.

Полезность системы – способность системы в ходе диалога определить потребность пользователя, выявить и устранить причины неудач в работе и удовлетворить потребности пользователя (т.е. решить поставленные задачи).

Удобство работы с системой заключается в:

- естественности взаимодействия в привычном, не утомляющем пользователя виде;
- способности настраиваться на различных пользователей, а также учитывать изменения в квалификации одного и того, же пользователя;
- устойчивости к ошибкам, способности не выходить из строя при ошибочных действиях неопытного пользователя.

По результатам *опытной эксплуатации* может потребоваться не только модификация правил и данных, но и изменение устройств ввода-вывода из-за их неприемлемости для пользователя. По результатам этого же этапа принимается решение о *внедрении* системы. После завершения этапа опытной эксплуатации и использования экспертной системы различными пользователями она может классифицироваться как промышленная экспертная система.

Вопросы для самопроверки

1. Перечислите этапы разработки экспертных систем
2. В чем заключается этап идентификации разработки экспертных систем?
3. В чем заключается процесс идентификации задач?
4. Что такое идентификация целей?
5. Какие виды диаграммы используются при разработке экспертных систем и для чего?
6. В чем заключается этап формализации разработки экспертных систем?
7. Назовите основные задачи формализации?
8. Что представляет собой этап обобщения?
9. В чем заключается различие декларативных и процедурных знаний?
10. Для чего создаются прототипы экспертной системы?
11. Назовите три аспекта тестирования экспертных систем
12. В чем заключается этап эксплуатации и внедрения экспертной системы?

4 Выявление знаний от экспертов

4.1 Основная идея

Эффективность начальных этапов разработки экспертных систем (этапов идентификации и концептуализации) во многом определяется успешным формированием авторитетной группы экспертов и получением от них качественных знаний, составляющих основу любой экспертной системы.

4.2 Экспертное оценивание как процесс измерения

Суть процесса выявления знаний заключается в организации проведения экспертами интуитивно-логического анализа проблемной области с количественной оценкой формулируемых ими суждений. На этом этапе эксперты:

- формируют объекты и понятия предметной области (цели, решения, альтернативные ситуации и т.д.);
- производят измерение характеристик (вероятности свершения событий, коэффициенты значимости целей, предпочтение решений и т.д.).

Экспертное оценивание представляет собой процесс измерения, который можно определить как процедуру сравнения объектов по выбранным показателям (признакам). В этом определении фигурируют три понятия: объект, показатель (признак) и процедура сравнения.

Объекты – предметы, явления, решения.

Показатели сравнения – пространственно-временные, физические, психические и другие свойства и характеристики объектов.

Процедура сравнения включает в себя:

- определение причинно-следственной связи между объектами;
- установление степени влияния одних объектов на другие.

Введение конкретных показателей сравнения позволяет экспертам устанавливать отношения между объектами, например, «больше», «лучше», «более чем», «хуже», «одинаковы», «предпочтительнее» и т.д.

Эмпирическая система – это совокупность интересующих нас объектов вместе с системой связывающих их отношений.

При экспертном оценивании предметной области важным является возможность использования для эмпирической системы с отношениями построения числовой системы с отношениями, описывающими влияние объектов и отношения между ними с помощью чисел.

4.3 Методы измерения степени влияния объектов

Типовыми методами измерения степени влияния объектов являются:

- ранжирование;
- парное сравнение;
- непосредственная оценка.

Ранжирование – процедура упорядочения объектов по степени их влияния на результат. На основе своих знаний и опыта эксперт располагает объекты в порядке предпочтения, руководствуясь одним или несколькими показателями сравнения. В зависимости от вида отношений между объектами возможны различные варианты упорядочения объектов.

Парное сравнение представляет собой процедуру установления предпочтения объектов при сравнении всех возможных пар. В отличие от ранжирования, при котором осуществляется упорядочение всех объектов сразу, парное сравнение представляет для экспертов более простую задачу.

Непосредственная оценка представляет собой процедуру приписывания объектам числовых значений в шкале интервалов. Эти значение соответствует степени влияния того или иного объекта на наблюдаемый результат. В процессе выявления знаний эксперт должен сопоставить каждому объекту значение, например, из диапазона [0, 1]. Естественно потребовать, чтобы сходным объектам приписывалось бы одно и то же значение. Измерение предпочтения в шкале интервалов можно выполнить с высокой степенью доверия только при хорошей информированности экспертов о свойствах объектов и предметной области. В ряде случаев, для ослабления этих условий за счет уменьшения точности измерения, рассматривают оценку, которая использует 5-, 10-, 100-бальные шкалы. Однако непосредственная оценка не всегда должна использовать числовые шкалы. Например, цвет объекта невозможно представить в виде какого-либо числового значения, а переход к значениям частот спектра во многих случаях затруднителен для эксперта.

4.4 Характеристики экспертов и группы экспертов

Основными *характеристиками экспертов* являются:

- компетентность;
- креативность;
- отношение к экспертизе;
- конформизм;
- коллективизм и самокритичность.

Основными *характеристиками группы экспертов* является:

- достоверность экспертизы;
- затраты на экспертизу.

Обе эти характеристики определяют количество экспертов в группе и их качество (рисунок 4.1).

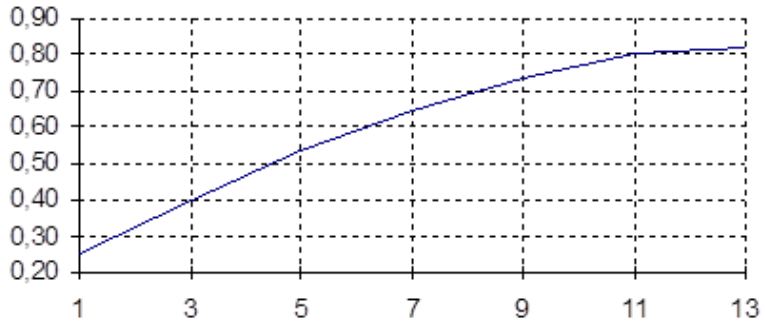


Рисунок 4.1 – Зависимость достоверности экспертизы от числа экспертов

Наряду с компетентностью, каждый из экспертов может характеризоваться достоверностью суждений. Эта характеристика определяется на основе информации о прошлом опыте его участия в решении проблем.

Для решения проблем с высоким уровнем информационного потенциала знаний, увеличение количества экспертов в группе приводит, как это следует из теории обработки наблюдений, к монотонному возрастанию достоверности экспертизы.

Опрос – это основной этап совместной работы группы экспертов.

При коллективной экспертизе используются следующие виды опроса:

- дискуссия;
- анкетирование и интервьюирование;
- метод коллективной генерации идей;
- мозговой штурм.

Результат опроса – информация, выражающая предпочтение экспертов и содержательное обоснование этих предпочтений.

Наличие, как числовых данных, так и содержательных высказываний экспертов, приводит к необходимости применения качественных и количественных методов обработки результатов группового экспертного оценивания.

Вопросы для самопроверки

1. Что такое экспертное оценивание, для чего он необходимо?
2. Что включает в себя процедура сравнения?
3. Дайте определение эмпирической системы.
4. Какие существуют методы для измерения степени влияния объектов?
5. В чем заключается процедура ранжирования объектов?
6. Что такое непосредственная оценка объектов?
7. Назовите основные характеристики экспертов.
8. Какие виды опросов используются при коллективной экспертизе?

5 Таблицы решений

Для представления знаний была предложена табличная форма, получившая название *таблицы решений*. Идея этого подхода очень проста и заключается в использовании табличной формы описания проблемы и способа ее решения (см. таблицу 5.1).

Таблица 5.1 – Структура таблицы решений

Условия	**Объекты**			
	O_1	O_2	...	O_K
$У_1$	1	0	…	1
…	…	…	…	…
$У_I$	1	1	…	0
Действия				
$Д_1$	0	1	…	1
…	…	…	…	…
$Д_J$	1	0	…	1

В этой таблице предполагается наличие двух основных частей: строк условий $У_i$ и строк действий $Д_j$. Каждое условие и действие соотносится с набором определенных объектов O_k.

Если некоторое условие или действие имеет место для данного объекта, то в соответствующую ячейку таблицы заносится 1, в противном случае используется символ 0.

В общем случае в ячейках таблицы решений могут использоваться любые символы, отражающие суть условия или действия: значения и диапазоны значений, коды и номера, символьные данные и т.п. В качестве действия может выступать также набор инструкций, подлежащих выполнению в случае выполнения условий.

Структура таблицы тоже может быть модифицирована под конкретную задачу. Например, может добавиться столбец с кодом операции, которую необходимо выполнять для проверки заданного условия. Возможно использование упрощенного варианта таблицы, состоящей из пар «условие-действие» без указания каких-либо объектов.

Не представляет труда и расширение таблицы на новые классы объектов – необходимо лишь добавить в конец таблицы недостающие столбцы. А добавление новых строк, представляется несколько более сложным, так как потребуется внести изменения для всех существующих записей (столбцов).

Алгоритм поиска решения по таблице решений тривиален и прост в реализации. Интерпретирующая программа получает на вход конкретизированный набор условий для заданной ситуации и осуществляет последовательный перебор, подыскивая столбцы, где этот набор выполняется. Если такие столбцы находятся, то выполняется действие, указанное для этой ситуации.

Основное достоинство таблиц решений – высокая степень формализации, наглядности процесса принятия решений. Они строятся регулярным образом и могут наращиваться практически до бесконечности, то есть являются универсальным средством решения задач, для которых возможно описание ситуаций с помощью ограниченного набора условий, например, проектирование деталей, представляющих собой поверхности вращения.

Недостаток – если различные ситуации характеризуются разными условиями, то таблицы решений становятся сильно разреженными и делают данный подход малоэффективным.

Рассмотрим пример использования таблицы решений, реализованный в системе *STRIPS* (*Stanford Research Institute Problem Solver*). Программа предназначалась для решения проблемы формирования плана поведения робота, перемещающего предметы через множество комнат.

Текущее состояние окружающей среды – помещений и предметов в них – представляется набором выражений предикат-аргумент, которые в совокупности образуют модель мира. Общепринятая конструкция предикат-аргумент, определяется следующим образом:

$$< \text{предикат} > :: =< \text{предикатный символ} > (< \text{аргумент}_1 >, \dots, < \text{аргумент}_n >).$$

Различают одноместные или *n*-местные предикаты. В случае одноместного предиката считается, что аргумент обладает свойством, выраженным предикатным символом. *N*-местный предикат описывает отношение между объектами, которые заданы аргументами.

Примером ситуации, описываемой с помощью двухместного предиката, может быть нахождение робота в определенной комнате:

$$at(robot, roomA).$$

Данный предикат означает, что объект *robot* находится в комнате *roomA*.

Порядок следования аргументов в предикате определяется разработчиком, но должен быть неизменным везде, где этот предикат используется.

Можно было бы написать *at*(*roomA, robot*) и придать этой записи аналогичный смысл, но использовать одновременно в пределах одной базы знаний записи *at*(*robot*, *roomA*) и *at*(*roomA*, *robot*) нельзя.

С помощью множества предикатов можно описать текущую модель мира, то есть набор конкретных объектов, их свойств и отношений.

Например, исходная ситуация может описываться множеством:

$$W1 = \{at(robot, roomA), at(box1, roomB), at(box2, roomC)\}.$$

Конечная ситуация также задается множеством предикатов:

$$WK = \{at(box1, roomA), at(box2, roomB)\}.$$

Для описания действий, которые может выполнять робот, используются операторы, применяемые к текущей модели мира. Эти операторы позволяют добавлять или изымать некоторые факты из текущей модели.

Например, действие «*Переместить робота из комнаты A в комнату B*» в модели *W*1 приводит к формированию новой модели *W*2.

При этом факт *at*(*robot, roomA*) будет изъят из модели, а *at*(*robot, roomB*) – добавлен.

Множество допустимых операций, таких как перемещение робота или перенос предметов, кодируются в таблице операторов, которая близка по структуре к таблице решений, но предполагает обязательное наличие модели мира.

Таблица 5.2 – Пример таблицы операторов

Оператор:	*move* (*X*, *Y*)	*push*(*X*, *Y*, *Z*)
Предварительный условия:	{*at*(*robot*, *X*)}	{*at*(*robot, Y*), *at*(*X*, *Y*)}
Список удалений:	{*at*(*robot*, *X*)}	{*at*(*robot, Y*), *at*(*X*, *Y*)}
Список добавлений:	{*at*(*robot*, *Y*)}	{*at*(*robot, Z*), *at*(*X*, *Z*)}

Задачей системы *STRIPS* является формирование плана действий робота для достижения цели. Таким образом, результатом работы системы должна быть последовательность операторов, применение которых к исходной модели мира позволяет достичь целевой модели. Зная целевое состояние среды можно было бы перебирать последовательно или случайно комбинации операторов, пока цель не будет достигнута. Но экспоненциальный рост количества вариантов при каждой новой проверке делает такой подход неприемлемым на практике.

Для предотвращения экспоненциального роста вариантов возможных решений в качестве основы для работы системы предложен *метод анализа целей и средств*, идея которого состоит в том, чтобы с каждой новой операцией отличие между текущим состоянием и целевым уменьшалось. Это предполагает наличие меры оценки «расстояния» в пространстве решений.

Например, если очередная подцель сформулирована в виде предиката *at*(*box1, roomA*), а ящик *box1* находится в комнате *roomB*, то перемещение робота из комнаты *roomA* в комнату *roomC* не приблизит текущее состояние к целевому (для модели *W*1). А перемещение из комнаты *roomA* в комнату *roomB*, наоборот, уменьшит «расстояние» между текущим и целевым состоянием, так как позволит на следующем шаге переместить ящик *box1* в комнату *roomA*.

Алгоритм поиска требуемых операторов системы *STRIPS* основан на сопоставлении очередной подцели и списков добавления в таблице операторов. Новые подцели выбираются из списка предварительного условия найденного оператора. Так, например, цель *at*(*box1, roomA*) соответствует элементу *at*(*X, Z*) в списке добавлений оператора *push*(*X, Y, Z*).

Сопоставление этих двух предикатов (*X* соответствует *box1*, а *Z* – *roomA*) позволяет выбрать из предварительного условия оператора новые подцели: *at*(*robot*, *Y*), *at*(*box1*, *Y*).

Далее необходимо найти в модели мира предикат, содержащий объект (в данном случае это комната), который можно сопоставить с символом *Y*.

Таким предикатом может быть, например, *at*(*box1*, *roomB*). Поставив в соответствие с символом *Y* объект *roomB* можно окончательно сформулировать очередные подцели: *at*(*robot*, *roomB*) и *at*(*box1*, *roomB*).

Теперь первый элемент в этом списке указывает желаемое (целевое) положение робота, а второй элемент уже присутствует в модели мира *W*1.

Следовательно, после применения оператора *push* к модели мира *W*1, необходимо добавить *at*(*robot*, *roomB*) и удалить *at*(*robot*, *roomA*). В результате получится новая модель:

$$W2 = \{at(robot, roomB), at(box1, roomB), at(box2, roomC)\}.$$

Описанная процедура повторяется до тех пор, пока очередная модель не будет соответствовать целевой.

Так как таблица операторов, модель мира и цели представлены с помощью одного и того же синтаксиса в виде конструкций предикат-аргумент, то, применяя описанную выше схему сопоставления, программа довольно просто находит, какие именно операции нужно выполнить для достижения поставленной цели. Всё, что нужно для этого сделать, – просмотреть списки добавлений в описании операторов и найти в них элемент, соответствующий заданной цели.

Вопросы для самопроверки

1. В чем заключается основная идея таблицы решений?
2. Назовите основное достоинство алгоритма поиска решений?
3. В чем заключается недостаток алгоритма поиска решений?
4. Для чего применяется система STRIPS?
5. Что является основной задачей системы STRIPS?

6 Продукционные правила

Системы представления знаний, использующие выражения вида «**ЕСЛИ** *условие,* **ТО** *действие*», получили название систем продукций или систем, основанных на правилах.

Правила обеспечивают формальный способ представления рекомендаций, указаний или стратегий. Они хорошо подходят в тех случаях, когда знания предметной области возникают из эмпирических ассоциаций, накопленных за годы работы по решению задач в данной области.

Представления знаний в виде продукций наиболее распространено в экспертных системах, так как запись знаний фактически ведется на подмножестве естественного языка.

Следствием этого является то, что правила легко читаются, их просто понять и модифицировать, эксперты без труда могут сформулировать новое правило или указать на ошибочность какого-либо существующего.

В качестве *условия* и *действия* в правилах может быть, например, предположение о наличии того или иного свойства, принимающее значение истина или ложь. При этом термин *действие* следует трактовать широко: это может быть как директива к выполнению какой-либо операции, рекомендация, или модификация базы знаний – предположение о наличии какого-либо производного свойства. Примером продукции может служить следующее выражение:

ЕСЛИ *клиент работает на одном месте более двух лет*,
ТО *клиент имеет постоянную работу*.

Как условие, так и действие правила могут учитывать несколько выражений, объединенных логическими связками **И**, **ИЛИ**, **НЕ**, например:

ЕСЛИ *клиент имеет постоянную работу* **И** *клиенту более 18 лет*
И *клиент* **НЕ** *имеет финансовых обязательств*,
ТО *клиент может претендовать на получение кредита*.

Помимо продукционных правил база знаний должна включать и простые факты, поступающие в систему через интерфейс пользователя или выводимые в процессе поиска решения задачи. Факты являются простыми утверждениями типа «*клиент работает на одном месте более двух лет*». И когда в процессе интерпретации правил машиной вывода какой-либо факт согласуется с частью правила **ЕСЛИ**, то выполняется действие, определяемое частью **ТО** этого правила.

Новые факты, добавляемые в базу знаний в результате действий, описанных в правилах, также могут быть использованы для сопоставления с частями **ЕСЛИ** других правил. Последовательное сопоставление частей правил **ЕСЛИ** фактами порождает *цепочку вывода* (см. рисунок 6.1).

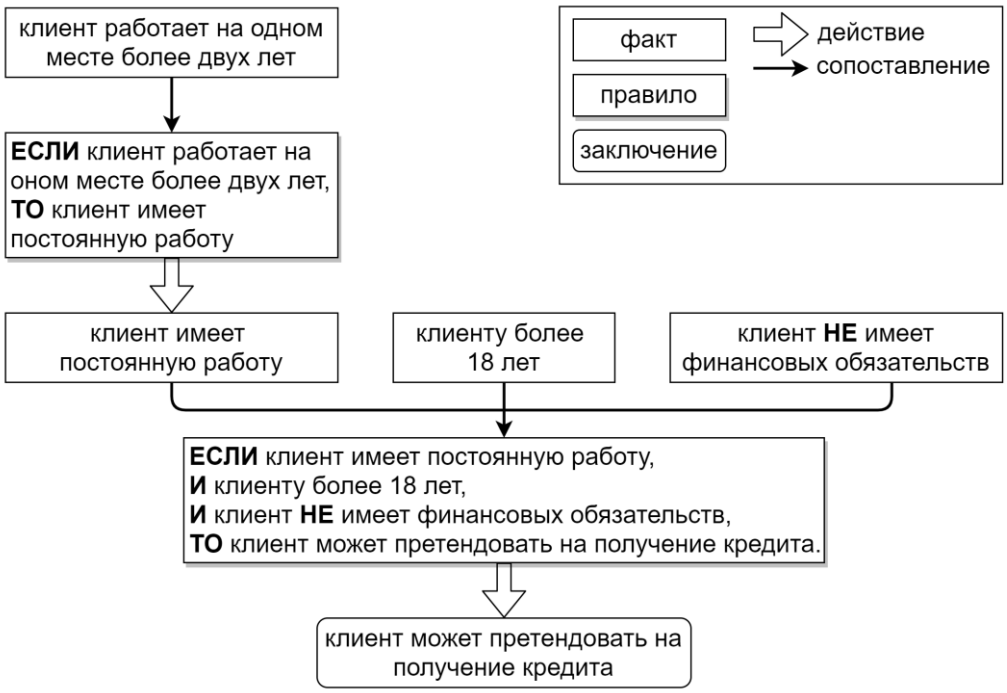


Рисунок 6.1 – Пример цепочки вывода

Эта цепочка вывода показывает, как на основании правил и исходных фактов выводит заключение о возможности получения кредита. Цепочки вывода экспертной системы могут быть предъявлены пользователю и помогают понять, как было получено решение.

Существуют два основных способа выполнения правил в системе: прямая цепочка рассуждений или *прямой вывод* и обратная цепочка рассуждений или *обратный вывод*. В приведенном выше примере использовалась прямая цепочка рассуждений – от исходных фактов к цели (заключению). Этот вид вывода применим в задачах, где на основании имеющихся фактов необходимо определить тип (класс) объекта или явления, выдать рекомендацию, определить диагноз и т.п.

Если же задача формулируется иначе – для заданной ситуации необходимо определить условия к ней приводящие, то используется обратный вывод. Подобная задача может формулироваться, например, следующим образом: «*Определить условия, при которых клиент может получить кредит*».

Обратная цепочка применима также, если требуется объяснить, как было получено решение.

В процессе вывода решения все правила системы равнозначны и самодостаточны, то есть все необходимое для активизации правила содержится в его условии, и одни правила не могут непосредственно вызывать другие. Но иногда для получения решения требуется вмешательство в стандартный процесс вывода. Для этих целей некоторые продукционные системы позволяют вводить в базу знаний специальные правила для управления процессом вывода – *метаправила*.

Метаправила не принимают непосредственного участия в процессе формирования рассуждений, а определяют приоритет выполнения или исключают из рассмотрения обычные правила и выполняются в первую очередь. Ниже приведен пример метаправила, сокращающего цепочку вывода:

ЕСЛИ *кредитный рейтинг клиента высокий*
И *клиент является клиентом банка*,
ТО *сначала применить правила для льготных условий предоставления кредита*.

Можно также сформулировать пример метаправила, касающегося общей стратегии вывода и не связанного с какой-либо конкретной предметной областью:

ЕСЛИ *существуют правила, в условиях которых не упоминается текущая цель*,
И *существуют правила, в условиях которых упоминается текущая цель*,
ТО *сначала следует активизировать первые из перечисленных правил*.

Последний пример не связан с предметной областью и, поэтому, подобные метаправила можно применять в системах различного назначения. Интерес к такого рода обобщенным формулировкам знаний достаточно высок. Идея использования метаправил является весьма продуктивной, но, тем не менее, метаправилами следует пользоваться осмотрительно, учитывая возможные исключительные ситуации.

Продукционные правила обеспечивают естественный способ описания процессов, управляемый сложной и изменяющейся средой. Через правила можно описать ход решения задачи, не имея заранее алгоритма этого решения. Более того, можно корректировать способ решения, добавляя новые правила, не изменяя существующих, что обеспечивает высокую модульность базы знаний.

Однако, несмотря на то, что с помощью продукционных правил можно представить решение любой задачи, при большом количестве правил становится сложно отслеживать непротиворечивость базы знаний.

Вопросы для самопроверки:

1. Какие системы представления знаний получили название «системы продукций»?
2. Что представляет собой цепочка вывода?
3. Назовите основные способы выполнения правил в экспертной системе.
4. Что такое метаправила?
5. Для чего необходимы продукционные правила?

7 Семантические сети

Семантика устанавливает отношения между символами и объектами, которые они обозначают, т.е. определяет смысл знаков.

Семантические сети впервые были предложены Р. Квиллианом в 1968 году в качестве как способа описания феноменов человеческой памяти.

Первоначально семантические сети использовались для анализа значения слов в предложениях. В последующем они успешно применялись для решения задач, связанных с представлением и извлечением знаний за счет возможности учитывать не только форм утверждений, но и их семантику.

Такие знания позволяют экспертным системам получить приемлемый вывод при работе с неоднозначными фактами.

Семантическая сеть – это ориентированный граф, вершины которого представляют понятия, а дуги – различные отношения между ними (см. рисунок 7.1).

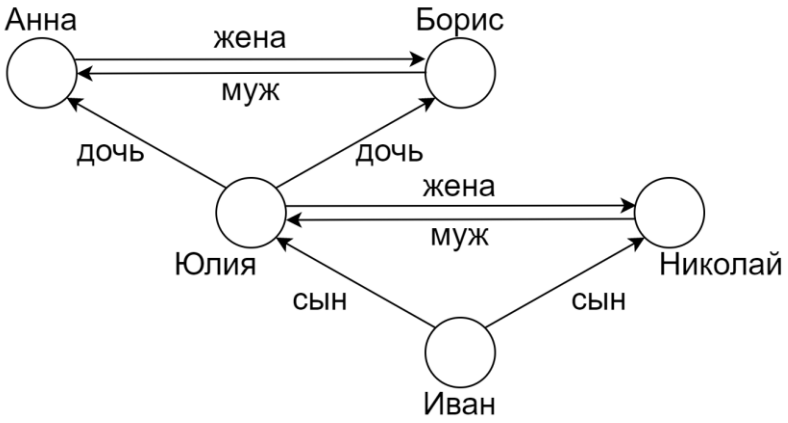


Рисунок 7.1 – Пример семантической сети

Понятия семантической сети – это абстрактные или конкретные объекты, а отношения – связи типа: *is-a, has-a, a-kind-of, cause*.

Связь *is-a* – означает, что отдельный объект *является экземпляром* определенного класса. Примером такого типа связей может быть отнесения клиентов банковских учреждений к определенному классу (рисунок 7.2).

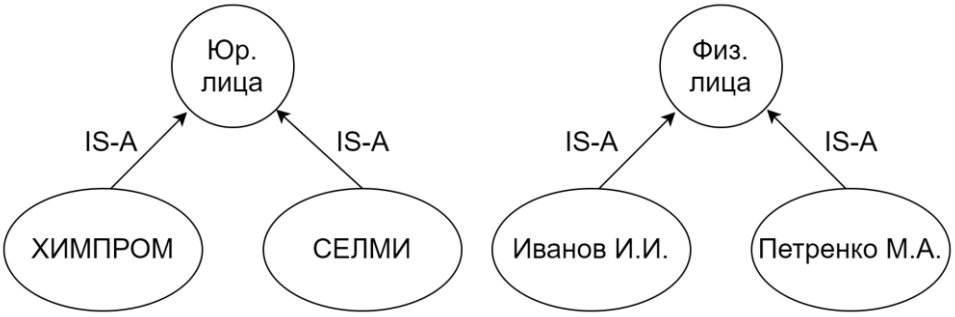


Рисунок 7.2 – Связь *is-a*

Связь *is-a* используется для обозначения отношений между отдельными объектами через принадлежность их к общему классу, благодаря тождественности атрибутов.

Связь *a-kind-of* определяет отношение между самими родовыми классами. Следует отметить, что общий класс, на который указывает стрелка, называется суперклассом (рисунок 7.3).

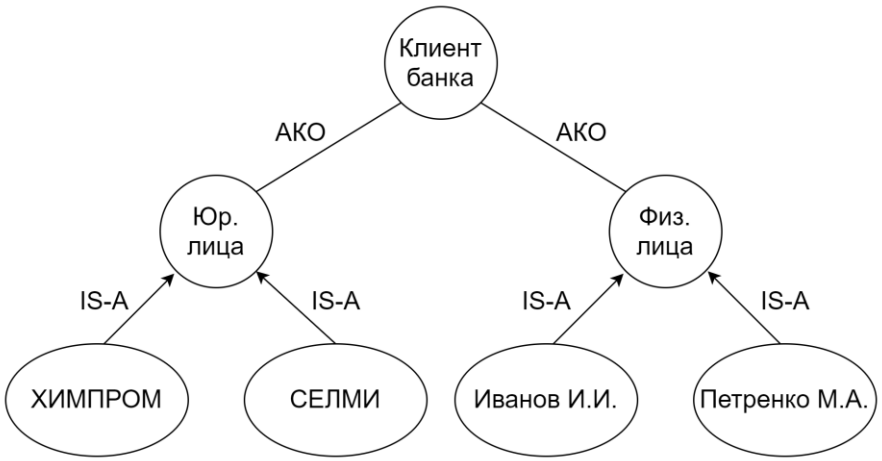


Рисунок 7.3 – Связь *a-kind-of*

В случае если суперкласс имеет связь *a-kind-of*, что указывает на другой узел, то он, вместе с тем, является классом суперкласса.

Таким образом, в семантических сетях есть три основных типа отношений:

- класс – элемент класса;
- свойство – значение;
- пример элемента класса.

По типу отношений семантические сети бывают:

- однородные (с единственным типом отношений);
- неоднородные (с различными типами отношений).

По типам отношений:

- бинарные (в которых отношения связывают два объекта);
- *n*-арные (в которых есть специальные отношения, связывающие более двух понятий).

Часто используемыми отношениями являются:

- связи типа «часть-целое» («класс-подкласс», «элемент-множество» и т.п.);
- функциональные связи («производит», «влияет» и т.п.);
- количественные («больше», «меньше», «равно» и т.п.);
- пространственные («далеко от», «близко от», «за», «над» и т.п.);
- временны*е* («раньше», «позже», «в течение» и т.п.);
- атрибутивные связи («иметь свойство», «иметь значение», и т.п.);
- логические связи (И, ИЛИ, НЕ) и др.

На рисунке 7.4 приведен пример семантической сети.

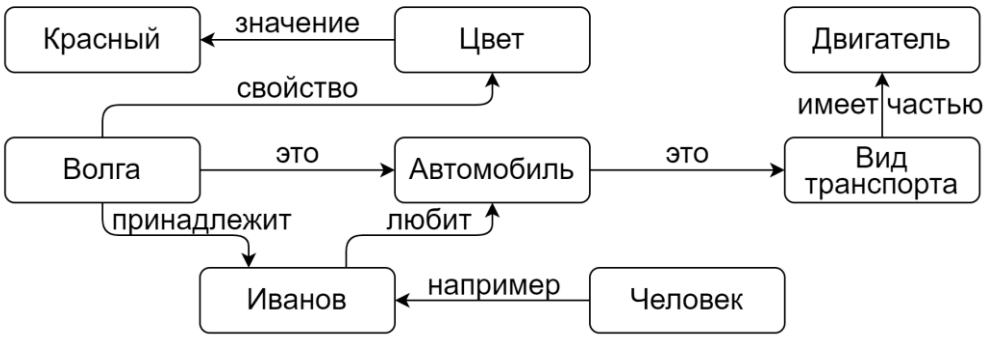


Рисунок 7.4 – Пример семантической сети

Поиск решения на основе знаний, представленных семантическими сетями, типа семантической сети сводится к задаче поиска фрагмента сети (подсети), соответствующей поставленному вопросу.

Достоинство семантических сетей – соответствие представлениям об организации долговременной памяти человека, а недостаток – сложность вывода.

Вопросы для самопроверки

1. Дайте определения семантики.
2. Что такое семантическая сеть?
3. Что представляет собой тип связи *is-a*?
4. Что представляет собой тип связи *a-kind-of*?
5. Для чего используется связь *is-a*?
6. Назовите типы отношений в семантических сетях?
7. Какие бывают семантические сети по типу отношений?
8. Что представляет собой *n*-арные семантические сети?
9. Назовите основные типы отношений, которые используются в семантических сетях?

8 Фреймы

Фреймы предложены М. Минским в 70-е годы XX века как структура знаний для восприятия пространственных сцен. Эта модель, как и семантическая сеть, имеет глубокое психологическое обоснование. Под фреймом понимается абстрактный образ или ситуация.

В психологии и философии известно понятие абстрактного образа. Например, слово «комната» представляет образ «жилого помещения с 4-мя стенами, полом, потолком, окнами и дверью, площадью до 30 кв.м.», из описания которого ничего нельзя убрать и «слоты» которого заполняются значениями атрибутов – количеством окон, цветом стен, высотой потолка, покрытием пола и др.

Такой образ, а точнее, его формализованная модель, и называется фреймом. В таблице 8.1 показана структура фрейма.

Таблица 8.1 – Структура фрейма

Имя фрейма			
Имя слота	Тип слота	Значение слота	Присоединенная процедура
…	…	…	…

Дополнительные столбцы предназначены для описания типа слота и возможного присоединения к тому или иному слоту специальных процедур. В качестве значения слота может выступать имя другого фрейма, так образуют сети фреймов.

Различают фреймы-образцы, или прототипы, хранящиеся в базе знаний, и фреймы-экземпляры, которые создаются для отображения реальных ситуаций на основе поступающих данных.

Модель фрейма позволяет отобразить знания через:

- фреймы-структуры, для обозначения объектов и понятий (например, заем, залог, вексель);
- фреймы-роли (например, менеджер, кассир, клиент);
- фреймы-сценарии (например, банкротство, собрание акционеров, празднование именин);
- фреймы-ситуации (например, тревога, авария, рабочий режим устройства) и др.

Важным свойством фреймов является заимствованное из теории семантических сетей наследование свойств. И во фреймах, и в семантических сетях наследование происходит по АКО-связям (*a-kind-of).* Слот АКО указывает на фрейм более высокого уровня иерархии, откуда неявно наследуются, т.е. переносятся, значения аналогичных слотов.

Основным достоинством фреймов как модели представления знаний является ее гибкость и наглядность. Так, на вопрос: «Любят ли ученики сладкое?» Следует ответ: «Да», так как этим свойством обладают все дети, что указано во фрейме «ребенок». Наследование свойств может быть частичным,

так, возраст для учеников не наследуется из фрейма «ребенок», поскольку указан явно в своем собственном фрейме (см. рисунок 8.1).

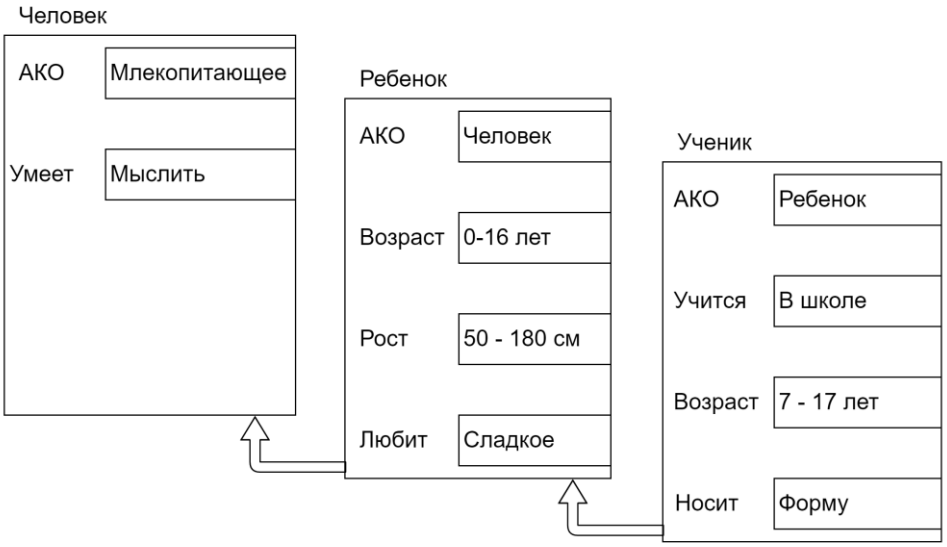


Рисунок 8.1 – Пример фрейма

Вопросы для самопроверки

1. Что такое фрейм?
2. Опишите структуру фрейма.
3. Для чего создаются фреймы-образцы и фреймы-экземпляры?
4. Какие преимущества дает создание фреймов?
5. Каким свойством обладает теория фреймов?
6. Приведите пример фрейма-сценария.
7. Приведите пример фрейма-ситуации.

9 Экспертные системы с неопределенными знаниями и байесовские сети доверия

9.1 Неопределенность в экспертных системах и проблемы, порождаемые ими

В практических задачах часто приходится оценивать гипотезы, относительно которых имеется неполная или недостаточная информация. Но несмотря на сложность точных оценок в условиях неопределенности, человеком принимаются разумные решения. Чтобы экспертные системы были полезными, они тоже должны уметь это делать.

Классическим примером таких задачи является медицинская диагностика. Всегда существуют некоторые сомнения в четкости проявления симптомов того или иного заболевания. Сомнения в наличии у пациента конкретного заболевания сохраняются даже в том случае, когда все его симптомы отчетливо выражены.

Как проявляется и учитывается неопределенность экспертной системой?

Пусть используется правило:

Если (*A*), **То** (*B*),

и предположим, никакие другие правила и посылки не имеют отношения к рассматриваемой ситуации.

При этом неопределенность может быть 2-х типов:

- неопределенность в истинности самой предпосылки (например, если степень уверенности в том, что *A* истинно составляет 90%, то какие значения примет *B*?);
- неопределенность самого правила (например, можно сказать, что в большинстве случаев, но не всегда, если есть *A*, то есть также и *B*).

Еще более сложная ситуация возникает в случае, если правило имеет вид:

Если (*A* **И** *B*), **То** (*C*),

где можно с некоторой степенью быть уверенными как в истинности каждой из посылок (*A*, *B*), а тем более их совместного проявления, так и в истинности самого вывода.

Типовыми проблемами при проектировании и создании экспертных систем являются следующие:

- Как количественно выразить степень определенности при установлении истинности (или ложности) некоторой части данных?
- Как выразить степень поддержки заключения конкретной посылкой?
- Как использовать совместно две (или более) посылки, независимо влияющие на заключение?
- Как быть в ситуации, когда нужно обсудить цепочку вывода для подтверждения заключения в условиях неопределенности?

Вероятность события определяется как отношение случаев, в которых данное событие происходит, к общему числу наблюдений.

Объективистский подход рассматривает вероятность отношения исходов ко всем наблюдениям в течении длительного времени. Этот подход основан на законе больших чисел, гарантирующим то, что при наличии достаточно большого количества наблюдений частота исходов, интересующего события будет стремиться к объективной вероятности.

Персонифицированный, субъективистский подход рассматривает вероятностную меру как степень доверия того, как отдельная личность судит об истинности некоторого высказывания (событию). При этом постулируют, что данная личность имеет в некотором смысле отношение к этому событию. Но это не отрицает возможности того, что две личности могут иметь различные степени доверия для одного и того же суждения.

Необходимый или логический подход расширяет вероятностную меру на множество утверждений, имеющих логическую связь такую, что истинность одного из них может выводиться из другого. Вероятность здесь характеризует степень доказуемости логически выверенного заключения. Такой взгляд можно рассматривать как расширение обычной логики.

Для вероятностных расчетов используются две направления:

- по Паскалю – байесовские правила для проверки и обработки мер доверия;
- по Бэкону – правила логики для доказательства или опровержения гипотез.

Общепринятые вероятности (по Паскалю) не могут быть получены из индуктивных вероятностей (по Бэкону) и, наоборот. Объективистский и субъективный взгляды используют расчеты по Паскалю. Те же, кто поддерживают логические выводы, используют расчеты по Бэкону.

Существуют экспертные системы, построенные в рамках каждого из этих направлений. Однако большинство современных экспертных систем, использующих теорию вероятностей, являются «байесовскими».

9.2 Основные понятия теории вероятностей

Пусть A некоторое событие.

Совокупность всех элементарных событий называется выборочным пространством или пространство событий (Ω).

Вероятность события A, обозначается $p(A)$.

Вероятность события A, обозначается $p(A)$ и каждая вероятностная функция p должна удовлетворять трем аксиомам:

1) Вероятность любого события A является неотрицательной

$$p(A) \geq 0 \text{ для } \forall A \in \Omega.$$

2) Вероятность всех событий выборочного пространства равна 1.

$$p(\Omega) = 1.$$

3) Если k событий являются взаимно независимыми (т.е. не могут произойти одновременно), то вероятность, по крайней мере, одного из этих событий равна сумме отдельных вероятностей.

$$p(A_1 \cup A_2 \cup \ldots \cup A_k) = \sum_{i=1}^{k} p(A_i).$$

Аксиомы 1) и 2) можно объединить

$$1 \geq p(A) \geq 0 \text{ для } \forall A \in \Omega.$$

Это утверждение показывает, что вероятность любого события – между 0 и 1. По определению, когда $p(A) = 0$, то событие A никогда не произойдет. В том случае и когда $p(A) = 1$, то событие A должно произойти обязательно.

Дополнение к A, обозначаемое $(\neg A)$, содержит совокупность всех событий за исключением A.

Так как A и $\neg A$ являются взаимонезависимыми, то из аксиомы 3) следует:

$$p(A) + p(\neg A) = p(A \cup \neg A) = p(\Omega) = 1.$$

Переписывая это равенство в виде:

$$p(\neg A) = 1 - p(A),$$

определяем способ для получения:

$$p(\neg A) \text{ из } p(A).$$

Предположим теперь, что B некоторое другое событие. Тогда вероятность того, что произойдет A при условии, что произошло B записывается в виде $p(A|B)$ и называется *условной вероятностью* события A при заданном событии B.

Вероятность того, что оба события A и B произойдут, называется *совместной вероятностью* событий A и B:

$$p(A \cap B).$$

Условная вероятность $p(A|B)$ равна отношению совместной вероятности к вероятности события B, при условии, что она не равна 0:

$$p(A|B) = \frac{p(A \cap B)}{p(B)}.$$

Аналогично условная вероятность события B при условии A:

$$p(B|A) = \frac{p(B \cap A)}{p(A)}.$$

Таким образом: $p(B \cap A) = p(B|A) \times p(A)$.

Так как совместная вероятность коммутативна, то

$$p(A \cap B) = p(B \cap A) = p(B|A) \times p(A).$$

Подставляя это равенство в ранее полученное выражение для условной вероятности $p(A|B)$ получим *правило Байеса*:

$$p(A|B) = \frac{p(B|A) \times p(A)}{p(B)}.$$

В ряде случае знание того, что произошло событие B, не влияет на вероятность события A (или наоборот A на B). Другими словами, вероятность события A не зависит от того, что произошло или нет событие B, так что

$$p(A|B) = p(A) \text{ и } p(B|A) = p(B).$$

В этом случае говорят, что события A и B являются независимыми.

9.3 Теорема Байеса

Приведенные выше соотношения предполагают определенную связь между теорией вероятностей и теорией множеств. Если A и B являются непересекающимися множествами, то объединение множеств соответствует сумме вероятностей, а пересечение – произведению вероятностей

$$p(A \cup B) = p(A) + p(B) \text{ и } p(A \cap B) = p(A) \times p(B).$$

Без предположения независимости эта связь является неточной и формулы должны содержать дополнительные члены включения.

Продолжая теоретико-множественную интерпретацию B, можно записать:

$$B = (B \cap A) \cup (B \cap \neg A).$$

Так как это объединение явно непересекающееся, то

$$p(B) = p\big((B \cap A) \cup (B \cap \neg A)\big) = p(B \cap A) + p(B \cap \neg A) = \\ = p(B|A)p(A) + p(B|\neg A)p(B).$$

Возвращаясь к обозначению событий, а не множеств, последнее равенство может быть подставлено в правило Байеса

$$p(A|B) = \frac{p(B|A) \times p(A)}{p(B|A) \times p(A) + P(B|\neg A) \times p(\neg A)}.$$

Это выражение является основой для использования положений теории вероятности при учете неопределенности. Оно определяет способ получения условной вероятности события B при условии A. Это соотношение позволяет экспертным системам «делать вывод вперед и назад».

9.4 Байесовские сети доверия

Байесовские сети доверия (*Bayesian Belief Network*) используются в задачах, которые характеризуются неопределенностью вследствие: неполного понимания задачи; неполных знаний; наличия случайных событий.

Для байесовских сетей доверия (БСД) используется ещё одно название – причинно-следственные сети, если случайные события в них соединены причинно-следственными связями. Такие связи наиболее часто делается в направлении от «наблюдателя» к «наблюдению», или от «эффекта» к «следствию» (рисунок 9.1).

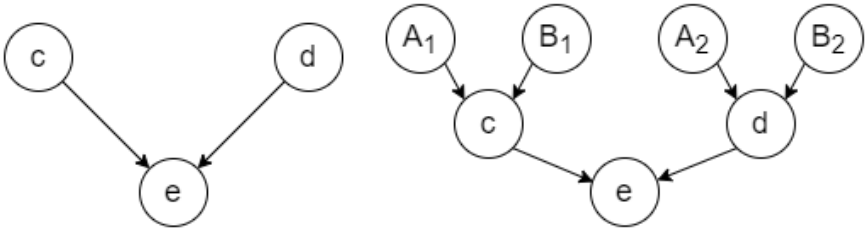


Рисунок 9.1 – Примеры байесовских сетей доверия

Вероятность пребывания вершины «*e*» в различных состояниях (e_k) зависит от состояний (c_i, d_j) вершин «*c*» и «*d*».

Байесовская сеть доверия представляется в виде направленного ациклического графа, обладающего следующими свойствами:

- каждая вершина представляет собой событие, описываемое случайной величиной, которая может иметь несколько состояний;
- все вершины, связанные с «родительскими» определяются таблицей условных вероятностей (ТУВ) или функцией условных вероятностей (ФУВ);
- для вершин без «родителей» вероятности их состояний являются безусловными (маргинальными).

Другими словами, в БСД вершины представляют собой случайные переменные, а дуги – вероятностные зависимости, которые определяются через ТУВ. Таблица условных вероятностей каждой вершины содержит вероятности состояний этой вершины при условии состояний её «родителей».

Рассмотрим пример построения простейшей БСД.

Однажды дерево лишилось листвы. Выясним, почему это случилось. Знаем, что листва часто опадает, если:

- дерево засыхает в результате недостатка влаги;
- или дерево болеет.

Данная ситуация может быть смоделирована байесовской сетью доверия, содержащей 3 вершины: «*Болеет*», «*Засохло*» и «*Облетело*» (рисунок 9.2).

Рисунок 9.2 – Пример простейшей БСД

Рассмотрим ситуацию, при которой каждая вершина может принимать всего лишь два возможных состояний и, как следствие находится в одном из них (таблица 9.1).

Таблица 9.1 – Табличная иллюстрация БСД

Вершина (событие) БСД	Состояние 1	Состояние 2
«*Болеет*»	«да»	«нет»
«*Засохло*»	«да»	«нет»
«*Облетело*»	«да»	«нет»

Вершина «*Болеет*» говорит о том, что дерево заболело, будучи в состоянии «болеет», в противном случае она находится в состоянии «нет». Аналогично для других двух вершин. Рассматриваемая БСД, моделирует тот факт, что имеется причинно-следственная зависимость от события «Болеет» к событию «Облетело» и от события «Засохло» к событию «Облетело». Это отображено стрелками на БСД.

Когда имеется причинно-следственная зависимость от вершины *A* к другой вершине *B*, то ожидаем, что когда *A* находится в некотором определённом состоянии, это оказывает влияние на состояние *B*. Следует быть внимательным, когда моделируется зависимость в БСД. Иногда совсем не очевидно, какое направление должна иметь стрелка.

В рассматриваемом примере, говорим, что имеется зависимость от «*Болеет*» к «*Облетело*», так как когда дерево болеет, это может вызывать опадание его листвы. Опадание листвы является следствием болезни, а не болезнь – следствием опадания листвы.

На рисунке дано графическое представление байесовской сети доверия. Однако, это только качественное представление байесовской сети доверия. Перед тем, как назвать это полностью байесовской сетью доверия необходимо определить количественное представление, то есть множество таблиц условных вероятностей (таблицы 9.2 и 9.3).

Таблица 9.2 – Таблицы условных вероятностей для родительских вершин

Априорная вероятность *p*(«*Болеет*»)			Априорная вероятность *p*(«Засохло»)	
Болеет = «да»	Болеет = «нет»		Засохло = «да»	Засохло = «нет»
0,1	0,9		0,1	0,9

Таблица 9.2 – Таблица условных вероятностей для вершины-потомка

Таблица условных вероятностей p(«*Облетело*») | «*Болеет*», «*Засохло*»)				
	Засохло = «да»		Засохло = «нет»	
	Болеет = «да»	Болеет = «нет»	Болеет = «да»	Болеет = «нет»
Облетело = «да»	0,95	0,85	0,90	0,02
Облетело = «нет»	0,05	0,15	0,10	0,98

Заметим, что все три таблицы показывают вероятность пребывания некоторой вершины в определённом состоянии, обусловленным состоянием её родительских вершин. Но так как вершины «*Болеет*» и «*Засохло*» не имеют родительских вершин, то их вероятности являются маргинальными, т.е. не зависят (не обусловлены) ни от чего.

В примере пусть известно, что дерево сбросило листву. Это свидетельство вводится выбором состояния «да» в вершине «*Облетело*». После этого можно узнать вероятности того, что дерево засохло. Для приведенных выше исходных данных:

$$p(\text{«Болеет» = «да» | «}\textit{Облетело}\text{» = «да»}) = 0{,}47;$$
$$p(\text{ «Засохло» = «да» | «Облетело» = «да»}) = 0{,}49.$$

Вопросы для самопроверки

1. Какие типы неопределенностей встречаются в экспертных системах?
2. Какие проблемы необходимо решать при проектировании и создании экспертных систем?
3. В чем заключается объективистский подход теории субъективных вероятностей?
4. Какие направления вероятностных расчетов существуют?
5. Что характерно для байесовских сетей доверия?
6. Назовите свойства направленного ациклического графа в байесовской сети доверия?
7. Что представляют собой вершины в байесовских сетях доверия?
8. Что представляют собой дуги в байесовских сетях доверия?

10 Интеллектуальные системы извлечения знаний, генетические алгоритмы

Интеллектуальные системы извлечения новых знаний (обучения и самообучения) ориентированы на автоматическое накопление и формирование знаний с использованием процедур анализа и обобщения данных. К ним относятся системы символьного, нейросетевого и эволюционного обучения.

Системы символьного обучения ориентированы на интеллектуальный анализ данных (*Data Mining*), поиск скрытых правил и закономерностей (*Knowledge Discovery*), автоматические рассуждения, доказательство теорем.

При поиске закономерностей задача и относящаяся к ней информация описывается в виде логических аксиом. В дальнейшем система рассматривает различные варианты задачи как теоремы, которые следует доказать.

Нейросетевые системы используют методы обучения, направленные на модификацию собственной структуры (структуры сети) и весовых коэффициентов связей между элементами.

Эволюционные системы построены на принципах генетических и эволюционных процессов (алгоритмов), когда из набора кандидатов (популяции), получаемого посредством операций скрещивания и мутаций, по принятому критерию отбираются лучшие, наиболее приспособленные для решения «особи». Они реализуют один из эффективных подходов к решению многомодальных (имеющих несколько локальных экстремумов) оптимизационных задач большой размерности. Эволюционные вычисления не гарантируют обнаружения глобального экстремума целевой функции (оптимального решения) за определенное время, однако, позволяют найти «хорошие» решения очень трудных задач.

Основными направлениями исследований в области эволюционных вычислений являются следующие:

- эволюционное программирование;
- эволюционные стратегии;
- генетические алгоритмы.

В основе *эволюционного программирования* лежит идея представления альтернатив решения задачи в виде универсальных конечных автоматов, которые способны реагировать на стимулы, поступающие из окружающей среды. Абстрактный автомат в теории алгоритмов (Л.Дж. Фогель, 1960 г.) – модель дискретного устройства, имеющего один вход, один выход и в каждый момент времени находящегося в одном состоянии из множества возможных. На вход поступают символы одного алфавита, на выходе формируются символы (в общем случае) другого алфавита. Автомат может быть представлен в виде графа, вершинами которого являются состояния, а дуги – переходами между ними. Как правило, автоматы используются для описания поведения программного обеспечения, аппаратных и технических средств.

Эволюционные стратегии (И. Рехенберг, 1964 г.) – это эвристические методы оптимизации, основанные на принципах адаптации и эволюции. Альтернативы представляется единым массивом, воздействие стратегией на которые осуществляется с учетом семантики альтернатив и направлено на улучшение значений входящих в них параметров. При поиске решения вначале происходит скрещивание особей для получения потомков и их мутация, а затем выполняется детерминированный отбор без повторений лучших особей из родителей и порожденных потомков.

Отличительной особенностью *генетических алгоритмов* (Дж. Холланд, 1975 г.) является представление любой альтернативы решения в виде кодовой (как правило, битовой) строки фиксированной длины, манипуляции с которой проводятся в отсутствие всякой связи с ее смысловой интерпретацией.

Рассмотрим более подробно основные вопросы построения и использования генетических алгоритмов. На рисунке 10.1 проиллюстрирована схема естественного отбора в природе.

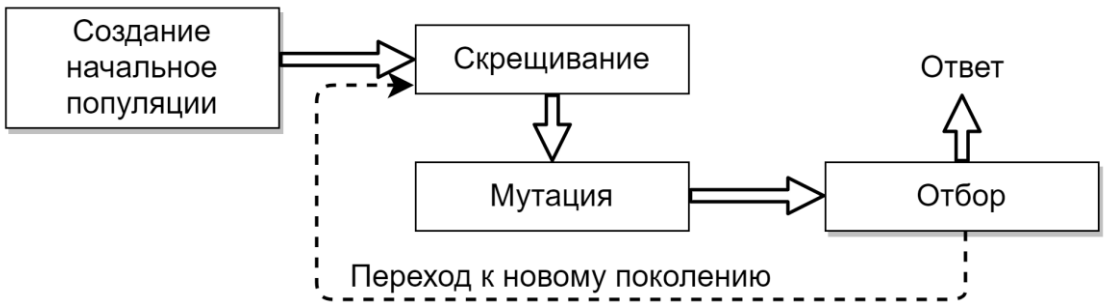


Рисунок 10.1 – Схема естественного отбора

Суть *естественного отбора*, как основного механизма эволюции, состоит в том, что более приспособленные особи имеют больше возможностей для выживания и размножения и, следовательно, приносят больше потомства, чем плохо приспособленные особи. При этом благодаря передаче генетической информации (генетическому наследованию) потомки наследуют от родителей основные их качества. Таким образом, потомки сильных индивидуумов также будут относительно хорошо приспособленными, а их доля в общей массе особей будет возрастать. После смены нескольких десятков или сотен поколений средняя приспособленность особей данного вида заметно возрастает.

Механизмы генетического наследования в природе основан на том, что в каждой клетке любого животного содержится вся генетическая информация этой особи. Эта информация записана в виде набора очень длинных молекул ДНК. Каждая молекула ДНК – это цепочка, состоящая из молекул нуклеотидов четырех типов, обозначаемых A, T, C и G. Собственно, информацию несет порядок следования нуклеотидов в ДНК.

Генетический код индивидуума – это просто очень длинная строка символов, где используются всего 4 буквы. В животной клетке каждая молекула ДНК окружена оболочкой такое образование называется хромосомой.

Каждое врожденное качество особи (цвет глаз, наследственные болезни, тип волос и т.д.) кодируется определенной частью хромосомы, которая называется геном этого свойства. Так, ген цвета глаз содержит информацию, которая кодирует определенный цвет глаз. Различные значения гена называются его аллелями.

При размножении животных происходит слияние двух родительских половых клеток и их ДНК взаимодействуют, образуя ДНК потомка. Основной способ взаимодействия кроссовер (crossover, скрещивание). При кроссовере ДНК предков делятся на две части, а затем обмениваются своими половинками.

При наследовании возможны мутации из-за радиоактивности или других влияний, в результате которых могут измениться некоторые гены в половых клетках одного из родителей. Измененные гены передаются потомку и придают ему новые свойства. Если эти новые свойства полезны, они, скорее всего, сохранятся в данном виде – при этом произойдет скачкообразное повышение приспособленности вида.

Рассмотрим основные понятия генетических алгоритмов.

Ген (свойство) – атомарный элемент хромосомы, может быть битом, числом или неким другим объектом.

Аллель – значение конкретного гена.

Локус – положение конкретного гена в хромосоме.

Хромосома (цепочка) – упорядоченная последовательность генов.

Генотип (код) – упорядоченная последовательность хромосом.

Особь (индивидуум) – конкретный экземпляр генотипа.

Фенотип – набор значений, соответствующих генотипу (представляет собой интерпретацию генотипа с точки зрения решаемой задачи).

Распространен случай, когда генотип состоит всего из одной хромосомы и представляется в виде битовой строки. Таким образом, ген – это бит; генотип (хромосома) – битовая строка, заданной размерности и с определенным положением битов; особь – конкретный набор битов (0 и 1).

На каждом шаге работы генетический алгоритм использует несколько точек поиска одновременно. Совокупность этих точек является набором особей, который называется *популяцией*. Количество особей в популяции называют *размером популяции*.

На каждом шаге генетический алгоритм обновляет популяцию путем создания новых особей и уничтожения ненужных. Чтобы отличать популяции на каждом из шагов и сами эти шаги, их называют *поколениями* и обычно идентифицируют по номеру. Например, популяция, полученная из исходной популяции после первого шага работы алгоритма, будет первым поколением, после следующего шага вторым и т. д.

При работе алгоритма генерация новых особей происходит на основе моделирования процесса размножения. При этом, естественно, порождающие особи называются *родителями*, а порожденные – *потомками*.

Выбор родителей для скрещивания выполняется случайным образом с помощью *оператора отбора*. При этом, один родитель может участвовать в нескольких скрещиваниях и необязательно, чтобы все родители участвовали в размножении.

На рисунке 10.2 показана типовая схема генетического алгоритма.

Рисунок 10.2 – Схема генетического алгоритма

Родительская пара, как правило, порождает пару потомков. Непосредственная генерация новых кодовых строк из двух выбранных происходит за счет работы *оператора скрещивания*, который также называют *кроссинговером* (англ. *crossover*).

Моделирование процесса мутации новых особей осуществляется за счет работы *оператора мутации*, применяемого к случайно выбранным потомкам за счет изменения случайного выбранного гена.

Поскольку размер популяции фиксирован, то порождение потомков должно сопровождаться уничтожением особей. Выбор лучших («жизнеспособных») особей из числа родителей и потомков выполняет *оператор редукции*, который уничтожает худшие («малоприспособленные») особи.

Основным правилом отбора является закон эволюции – «выживает сильнейший», который обеспечивает улучшение искомого решения. В некоторых источниках процесс приведения расширенной популяции к исходному размеру рассматривается не с точки зрения уничтожения худших особей (редукции), а с точки зрения выбора лучших (отбора).

Операторы отбора, скрещивания, мутации и редукции называют *генетическими операторами*. Отбор и мутация выполняются с использованием элементов случайности, скрещивание и редукция по строго детерминированным правилам.

Остановка генетического алгоритма происходит в случаях, когда:

- сформировано заданное число поколений;
- исчерпано время, отведенное на эволюцию;
- популяция достигла заданного качества – значение критерия одной, нескольких или всех особей превысило заданный порог;
- достигнут некоторый уровень сходимости – особи в популяции стали настолько подобными, что дальнейшее их улучшение происходит чрезвычайно медленно.

Основные отличия генетических алгоритмов от традиционных методов поиска решений:

- Генетические алгоритмы работают с кодовыми строками, от которых зависят значения аргументов целевой функции и, соответственно, значение самой целевой функции. Интерпретация этих кодов выполняется только в операторе редукции. В остальном работа алгоритма не зависит от смысловой интерпретации кодов.
- Для поиска лучшего решения генетический алгоритм на отдельном шаге использует сразу несколько точек поискового пространства (несколько вариантов решения задачи) одновременно, а не переходит от точки к точке, как это делается в традиционных методах. Это позволяет преодолеть один из их недостатков опасность попадания в локальный экстремум целевой функции, если она не является унимодальной (т.е. имеет несколько экстремумов). Использование нескольких точек одновременно значительно снижает такую возможность.
- Генетический алгоритм использует как вероятностные правила для порождения новых точек для анализа, так и детерминированные правила для перехода от одних точек к другим. Одновременное использование элементов случайности и детерминированности дает значительно больший эффект, чем раздельное.

Рассмотрим пример реализации генетического алгоритма. Пусть требуется найти глобальный минимум функции

$$y(x) = 5 - 24x + 17x^2 - \frac{11}{3}x^3 + \frac{1}{4}x^4$$

на отрезке [0, 7]

На этом отрезке функция принимает минимальное значение в точке $x = 1$. Очевидно, что в точке $x = 6$ функция попадает в локальный минимум. Если для нахождения глобального минимума использовать градиентные методы, то в зависимости от начального приближения можно попасть в данный локальный минимум.

Для простоты положим, что x принимает лишь целые значения, т.е. $x \in \{0, 1, 2, 3, 4, 5, 6, 7\}$.

Выберем случайным образом несколько чисел на отрезке [0, 7]: {2, 3, 5, 4}. Будем рассматривать эти числа в качестве пробных решений задачи.

Основной идеей генетических алгоритмов является организация «борьбы за существование» и «естественного отбора» среди этих пробных решений.

Запишем пробные решения в двоичной форме: {010, 011, 101, 100}.

Генетические алгоритмы используют биологические аналогии и принцип естественного отбора – в конкурентной борьбе выживает наиболее приспособленный. В нашем случае приспособленность особи определяется целевой функцией: чем меньше значение целевой функции, тем более приспособленной является особь, т.е. пробное решение, использовавшееся в качестве аргумента целевой функции (рисунок 10.3).

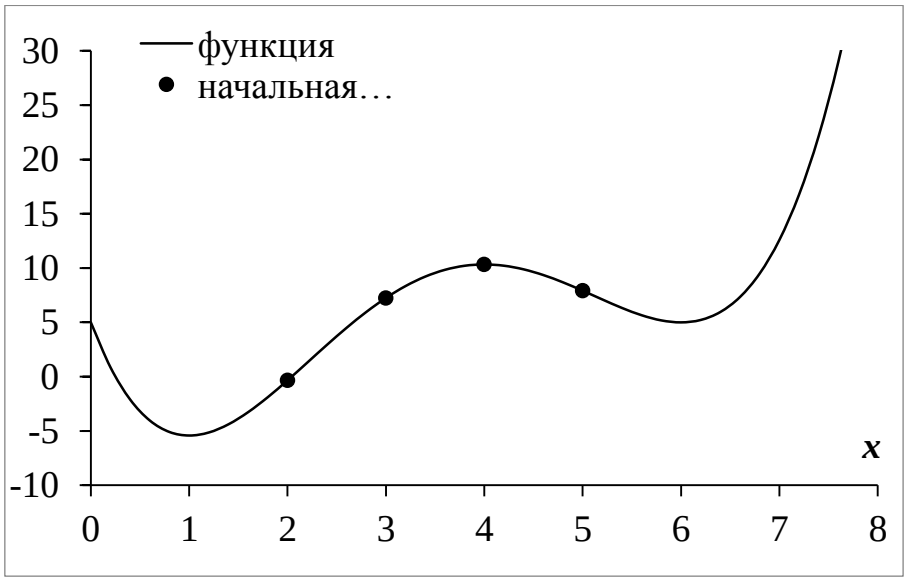


Рисунок 10.3 – График целевой функции с выбранными значениями пробных решений

Попробуем на основе исходной популяции создать новую, чтобы пробные решения в новой популяции были бы ближе к искомому глобальному минимуму целевой функции. Для этого сформируем из исходной популяции пары для скрещивания (таблица 10.1).

Таблица 10.1 – Исходная популяция

№, п/п	Особь		
	Целое число	Двоичное число	Приспособленность
1	2	010	–0,33
2	3	011	37,25
3	5	101	7,92
4	4	100	10,33

На этапе скрещивания поставим в соответствие каждой особи исходной популяции случайное целое число из диапазона от 1 до 4. Будем рассматривать эти числа как номера особей популяции. При таком выборе какие-то из особей популяции не будут участвовать в процессе скрещивания, так как образуют пару сами с собой. Какие-то особи популяции примут участие в процессе скрещивания неоднократно с различными особями популяции.

Процесс скрещивания (рекомбинация) заключается в обмене участками хромосом между родителями. Например, пусть скрещиваются две хромосомы 111111 и 000000. Определяем случайным образом точку разрыва хромосомы, пусть, это будет 3: 111|111 000|000. Теперь хромосомы обмениваются частями, стоящими после точки разрыва, и образуют двух новых потомков: 111000 и 000111 (таблица 10.2).

Таблица 10.2 – Одноточечный кроссинговер

№	Особь популяции	Выбранный номер	Вторая особь-родитель	Точка кроссинговера	Особи-потомки
1	010	1	010↔0|10	1	000
2	011	4	100↔1|00		110
3	101	3	101↔10|1	2	100
4	100	1	010↔01|0		011

Следующим этапом генетического алгоритма являются мутации, т.е. случайные изменения полученных в результате скрещивания хромосом. Пусть вероятность мутации равна 0,3. Для каждого потомка возьмем случайное число на отрезке [0, 1], и если это число меньше 0,3, то инвертируем случайно выбранный ген (заменим 0 на 1 или наоборот) (таблица 10.3, рисунок 10.4).

Таблица 10.3 – Мутация потомков

№	Особи-потомки	Случайное число	Выбранный ген для мутации	Потомок после мутации	Приспособленность потомка до мутации	Приспособленность потомка после мутации
1	000	0,1	3	001	5	-5,42
2	110	0,6	-	110	5	5
3	100	0,5	-	100	10,33	10,33
4	011	0,2	1	111	7,25	12,58

Как видно, мутации способны улучшить (1-й потомок) или ухудшить (4-й потомок) приспособленность особи-потомка.

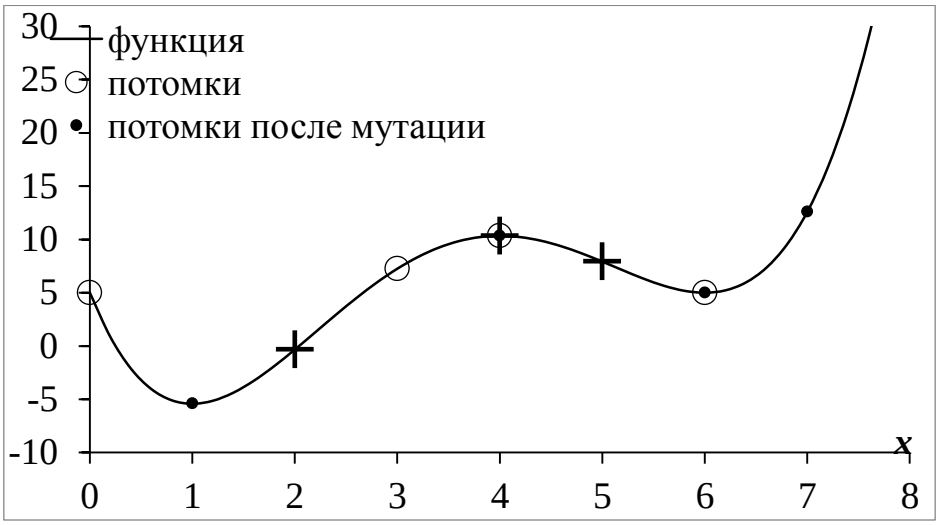


Рисунок 10.4 – График целевой функции с особями после мутации

В результате скрещивания хромосомы обмениваются «хвостами», т.е. младшими разрядами в двоичном представлении числа.

В результате мутаций изменению может подвергнуться любой разряд, в том числе, старший. Таким образом, если скрещивание приводит к относительно небольшим изменениям пробных решений, то мутации могут привести к существенным изменениям значений пробных решений.

Теперь из 4-х особей-родителей и 4-х полученных особей потомков необходимо сформировать новую популяцию. В новую популяцию отберем четыре наиболее приспособленных особей из числа «старых» особей и особей-потомков. Получившуюся популяцию можно будет вновь подвергнуть кроссинговеру, мутации и отбору особей в новое поколение. Таким образом, через несколько поколений мы получим популяцию из похожих и наиболее

приспособленных особей. Значение приспособленности наиболее «хорошей» особи (или средняя приспособленность по популяции) и будет являться решением нашей задачи.

Следуя этому, в данном случае, взяв наиболее приспособленную особь 001 во 2-м поколении, можно сказать, что минимумом целевой функции является значение −5,42, соответствующее аргументу $x = 1$ (таблица 10.4, рисунок 10.5). Тем самым попадания в локальный минимум удалось избежать!

Таблица 10.3 – Формирование новой популяции из особей-родителей и особей-потомков

№	Особи	Приспособленность	Новая популяция	Приспособленность особей в новой популяции
1	010	–0,33	001	–5,42
2	011	7,25	010	–0,33
3	101	7,92	110	5
4	100	10,33	011	7,25
5	001	–5,42		
6	110	5		
7	100	10,33		
8	111	12,58		

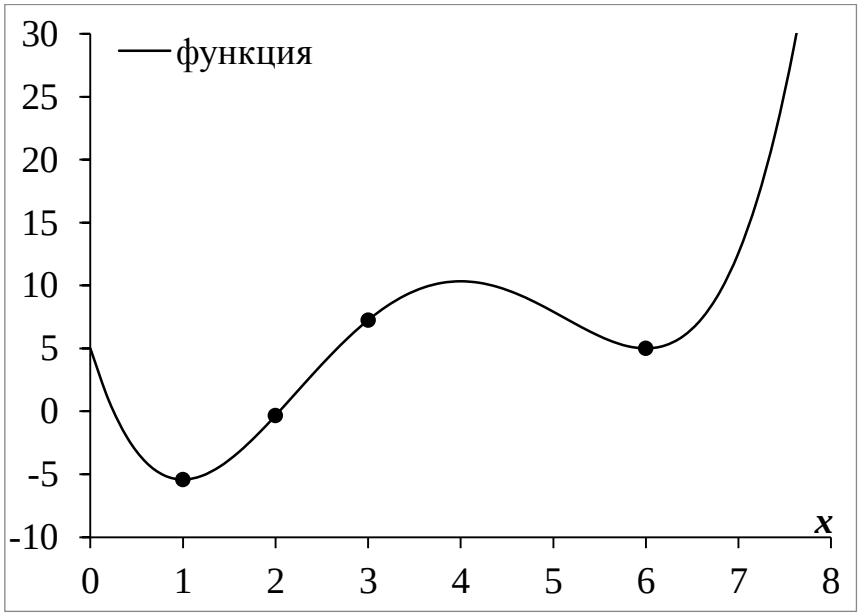


Рисунок 10.5 – График целевой функции после завершения работы генетического алгоритма

Рассмотрим пример работы генетического алгоритма при решении задачи коммивояжера. Коммивояжеру требуется посетить N городов. Для каждой пары городов по маршруту следования установлена стоимость (расстояние, время) проезда. Требуется найти путь минимальной стоимости, который начинается из некоторого города, обеспечивает посещение всех остальных городов ровно по одному разу и возврат в точку отправления (рисунок 10.6).

Задача коммивояжера относиться к категории *NP*-полных задач, т.е. задач решаемых методов полного перебора всех вариантов.

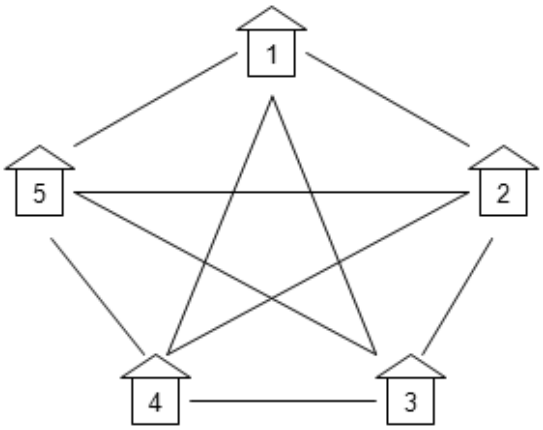


Рисунок 10.6 – Пример графического представления задачи коммивояжера

Ген – здесь число, характеризующее номер посещаемого города. Генотип – строка из чисел длиной *N*, описывающая порядок посещения городов. Особь – конкретная строка из чисел (допустимый вариант решения задачи).

Предположим, *N* = 9 и особи «231586479» и «147523869» примеры допустимых вариантов решения задачи. Классическое скрещивание приведет к генерации недопустимых вариантов, например, Родители Потомки:

23158 | 6479 23158 | 3869
14752 | 3869 14752 | 6479

так как в потомках посещение некоторых городов будет дублироваться или проигнорировано.

Предложены различные варианты решения данной проблемы, в частности, Л. Девис предлагает следующую модификацию оператора скрещивания.

Шаг 1. Случайным образом выбираются два сечения генотипа

P1 = 231 | 586 | 479
P2 = 147 | 523 | 869.

Шаг 2. Для потомков копируются участки кода, расположенные между сечениями

П1 = xxx | 586 | xxx,
П2 = xxx | 523 | xxx.

Шаг 3. Из родителей генерируются вспомогательные строки, у которых участки года после 2-го сечения перемещаются в начало

B1 = 479 231 586,
B2 = 869 147 523.

Шаг 4. Свободные гены потомков последовательно заполняются генами из перекрестных вспомогательных строк с пропуском уже имеющихся в потомке генов

П1 = 914 | 586 | 723,
П2 = 479 | 523 | 186.

Оператор мутации также может быть реализован различными способами, например:

- перестановка пары, случайным образом выбранных генов местами:

$$479523186 \rightarrow 473529186;$$

- инверсия случайным образом выбранной последовательности генов:

$$479 \mid 523 \mid 186 \rightarrow 479 \mid 325 \mid 186.$$

Вопросы для самопроверки

1. Какие типы систем извлечения новых знаний существуют?
2. На каких принципах построены нейросетевые системы?
3. Что представляют собой системы символьного обучения?
4. На каких принципах базируются эволюционные системы?
5. Назовите основные направления эволюционных вычислений.
6. Сформулируйте определение абстрактного автомата в теории алгоритмов.
7. Назовите основные направления эволюционных вычислений и их авторов.
8. В чем заключается основной механизм эволюции?
9. Что представляет собой генетический код индивидуума?
10. Назовите основные понятия, используемые в генетических алгоритмах.
11. Сформулируйте определения популяции и ее размера?
12. Какие операторы используются в генетических алгоритмах?
13. В каких случаях происходит остановка генетического алгоритма?
14. Назовите основные отличия генетических алгоритмов от традиционных методов поиска решений.

11 Искусственные нейронные сети

11.1 Краткая история исследований

1943 г. *У. Маккалох* (*W. McCulloch*) и *У. Питтс* (*W. Pitts*) предложили модель формального нейрона.

1949 г. *Д. Хебб* (*D. Hebb*) высказал идеи о характере соединений нейронов мозга и их взаимодействии (клеточные ансамбли, синаптическая пластичность). Впервые предложил правила обучения нейронной сети.

1957 г. *Ф. Розенблатт* (*F. Rosenblatt*) разработал принципы организации и функционирования персептронов, предложил вариант технической реализации первого в мире нейрокомпьютера Mark.

1959 г. *Д. Хьюбел* (*D. Hubel*) и *Т. Визель* (*T. Wiesel*) показали распределенный и параллельный характер хранения и обработки информации в биологических нейронных сетях.

1960–1968 г.г. Активные исследования в области искусственных нейронных сетей, АДАЛИНА и МАДАЛИНА *В. Уидроу* (*W. Widrow*) (1960–1962 гг.), ассоциативные матрицы *К. Штайнбуха* (*K. Steinbuch*) (1961 г.).

1969 г. Вышла книга *М. Минского* (*M. Minsky*) и *С. Пейперта* (*S. Papert*) «Персептроны», в которой доказывается ограниченность возможностей персептронов.

1974 г. *П. Вербос* разработал алгоритм обратного распространения ошибки (backpropagation) для обучения многослойных персептронов. Одновременно *А.И. Галушкин* опубликовал книгу, в которой обобщил подходы к обучению многослойных нейронных сетей. Заново алгоритм обратного распространения ошибки был «переоткрыт» в 1982 г. *Д. Паркером*.

1970–1976 гг. Активные разработки в области персептронов в СССР (основные заказчики – военные ведомства).

1975 г. *К. Фукушима* (*K. Fukushima*) предложил когнитрон – самоорганизующуюся сеть для инвариантного распознавания образов.

1980 г. *К. Фукушима* представил концепцию неокогнитрона.

1982–1985 г.г. *Дж. Хопфилд* (*J. Hopfield*) предложил семейство оптимизирующих нейронных сетей, моделирующих автоассоциативную память.

1982 г. *Т. Кохонен* (*T. Kohonen*) разработал обучающиеся без учителя самоорганизующиеся карты (Self-Organizing Maps).

1985 г. Первые коммерческие нейрокомпьютеры Mark III (TRW, США).

1987 г. *С. Гроссберг* (*S. Grossberg*) создал адаптивную резонансную теорию (АРТ) и модели нейронных сетей на ее основе.

1989 г. *Я. Лекун* (*Y. LeCun*) предложил сверточные нейронные сети для инвариантного распознавания изображений.

1990-е гг. Начало широкомасштабного финансирования разработок в области нейронных сетей в США, Японии и Западной Европе. Агентством DARPA (США) начато финансирование программы по созданию сверхбыстродействующих образцов нейрокомпьютеров.

1990-е гг. Активизация СССР в области искусственных нейронных сетей: Институт кибернетики им. Глушкова (Киев), Институт многопроцессорных вычислительных систем (Таганрог), Институт нейрокибернетики (Ростов-на-Дону), центры нейрокомпьютеров в Москве, Новосибирске, Санкт-Петербурге.

1997 г. *З. Хохрайтер* (*S. Hochreiter*) и *Ю. Шмидхубер* (*J. Schmidhuber*) предложили новый подход к созданию рекуррентных нейронных сетей – LSTM-сетей (Long Short-Term Memory).

2000-е гг. Решена проблема попадания искусственной нейронной сети в локальный минимум с применением стохастических методов обучения (больцмановское обучение, обучение Коши).

2007 г. *Дж. Хинтон* создал алгоритмы глубокого обучения многослойных нейронных сетей с использованием больцмановского обучения (RBM – Restricted Boltzmann Machine) для «нижних» слоев сети.

Искусственная нейронная сеть (ИНС) – структурированная совокупность искусственных нейронов, определенным образом соединенных друг с другом и с внешней средой с помощью связей, задаваемых весовыми коэффициентами.

Особенности искусственных ИНС:

- обучение на основе прецедентов (примеров);
- обобщение предыдущего опыта;
- извлечение значимой информации и закономерностей из избыточных и зашумленных данных;
- адаптивность к изменению условий функционирования.

Использование ИНС является целесообразным, если:

- отсутствует алгоритм решения задачи или неизвестен принцип ее решения, но имеются экспериментальные данные ее решения;
- задача характеризуется большими объемами информации;
- данные неполны, зашумлены, избыточны или противоречивы.

Постановка задачи для искусственных нейронных сетей: Необходимо построить такое отображение $X \to Y$, чтобы на каждый входной сигнал X формировался правильный выходной сигнал Y.

Области применения ИНС:

- классификация и распознавание образов (X – входной образ; Y – номер класса, к которому принадлежит входной образ);
- кластеризация/категоризация (X – входной вектор; Y – кластер, к которому относится входной вектор);
- аппроксимация функции (X – вектор входных переменных; Y – аппроксимированная выходная переменная);
- предсказание/прогноз (X – временные ряды на некотором интервале времени; Y – подмножество переменных входного сигнала);

- идентификация (*X* и *Y* представляют собой входные и выходные сигналы идентифицируемой системы, процесса)
- оптимизация;
- ассоциативная память.
- управление.

11.2 Искусственный нейрон

На рисунке 11.1 представлена типовая структура искусственного нейрона.

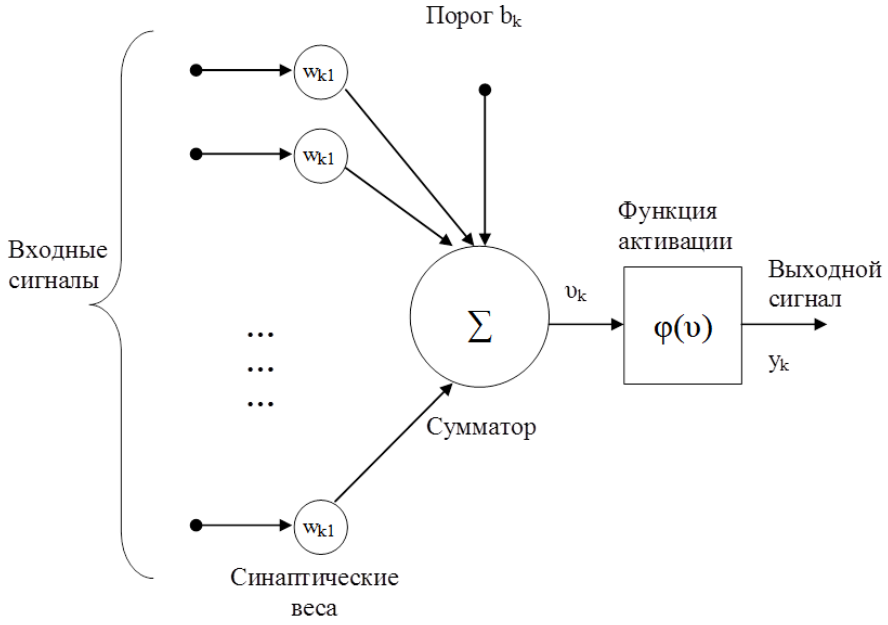


Рисунок 11.1 – Схема искусственного нейрона

Основные элементы нейрона:

- набор синапсов/связей, каждый из которых характеризуется; сигнал x_j на входе синапса j, связанного с нейроном $к$, умножается на вес w_{kj} (первый индекс относится к рассматриваемому нейрону, а второй – ко входному окончанию синапса, с которым связан данный вес); синаптический вес может быть как положительным, так и отрицательным;
- сумматор складывает входные сигналы, взвешенные относительно соответствующих синапсов нейрона;
- функция активации ограничивает амплитуду выходного сигнала нейрона; нормированный диапазон амплитуд выхода нейрона задается в диапазоне в интервале [0, 1] или [–1, 1].

Функционирование нейрона k описывается следующими выражениями:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j,$$

$$y_k = \varphi(u_k + b_k),$$

где $x_1, x_2, \ldots x_m$ – входные сигналы; $w_{k1}, w_{k2}, \ldots, w_{km}$ – синаптические веса k-го нейрона; u_k – линейная комбинация входных воздействий; b_k – пороговое смещение (обеспечивает эффект афинного преобразования выхода сумматора); φ – функция активации; y_k – выходной сигнал нейрона. Использование порога b_k.

В нейроне постсинаптический потенциал вычисляется следующим образом:

$$v_k = u_k + b_k.$$

В частности, в зависимости от того, какое значение принимает порог b_k, положительное или отрицательное, индуцированное локальное поле или потенциал активации u_k k-го нейрона изменяется так, как показано на рисунке 11.2.

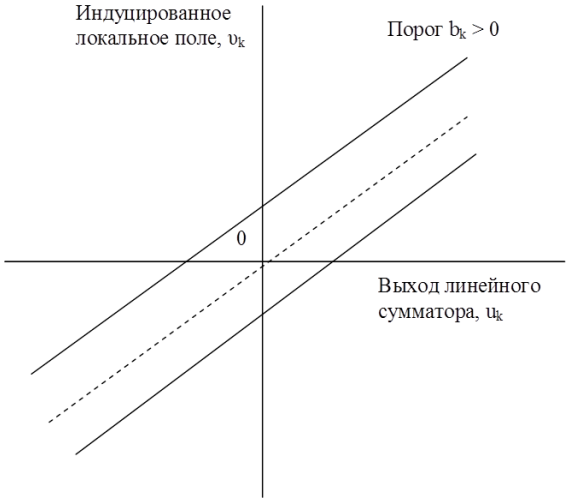


Рисунок 11.2 – Влияние порога на выход нейрона

На рисунке 11.3 проиллюстрированы примеры активационных функций искусственного нейрона:

а) функция единичного скачка: $f(s) = \begin{cases} 0, s < \Theta, \\ 1, s \geq \Theta; \end{cases}$;

б) кусочно-линейная функция: $f(s) = \begin{cases} 0, s < -a, \\ \dfrac{1}{2a}(s + a), -a < s < a, \\ 1, s > a; \end{cases}$

в) логистическая (сигмоидальная): $f(s) = \dfrac{1}{1 + e^{-as}}$;

г) гиперболический тангенс (сигмоидальная): $f(s) = \dfrac{e^{as} - e^{-as}}{e^{as} + e^{-as}}$.

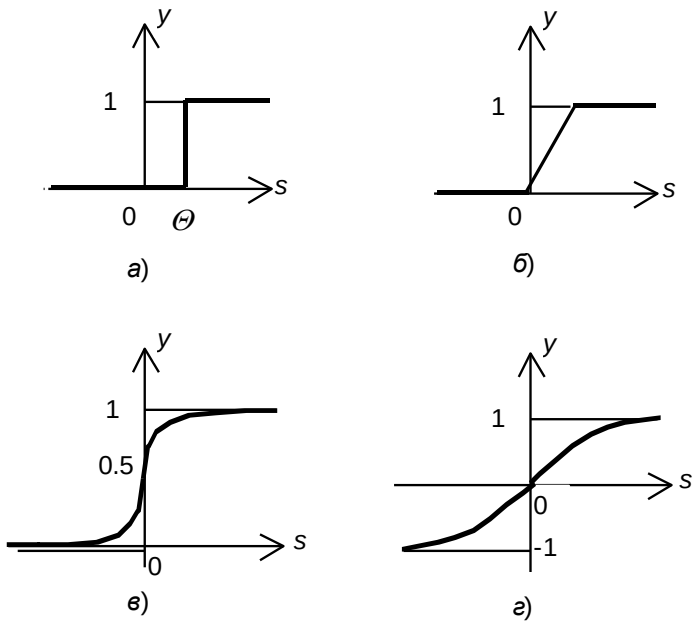


Рисунок 11.3 – Примеры активационных функций нейрона

Типы нейронов ИНС:

- *входные нейроны*, кодируют входные воздействия и передают «взвешенные» данные на промежуточные нейроны нейронной сети;
- *промежуточные нейроны* выполняют основные преобразования данных в нейронной сети;
- *выходные нейроны*, выполняют результирующие преобразования; их выходные значения являются выходами нейронной сети.

11.3 Классификация искусственных нейронных сетей и их свойства

В общем случае можно выделить три фундаментальных класса архитектур ИНС:

- однослойные сети прямого распространения;
- многослойные сети прямого распространения;
- рекуррентные сеть.

В *однослойных сетях прямого распространения (ациклических сетях)* имеется входной слой элементов, информация от которого передается на выходной слой нейронов (рисунок 11.4).

Многослойные сети прямого распространения характеризуются наличием одного или нескольких скрытых слоев нейронов (рисунок 11.5). Такая сеть позволяет выделять глобальные свойства данных с помощью локальных соединений за счет наличия дополнительных синаптических связей и повышения уровня взаимодействия нейронов. Способность скрытых нейронов выделять статистические зависимости высокого порядка особенно существенна, когда размерность признакового пространства (число нейронов входного слоя) достаточно велика.

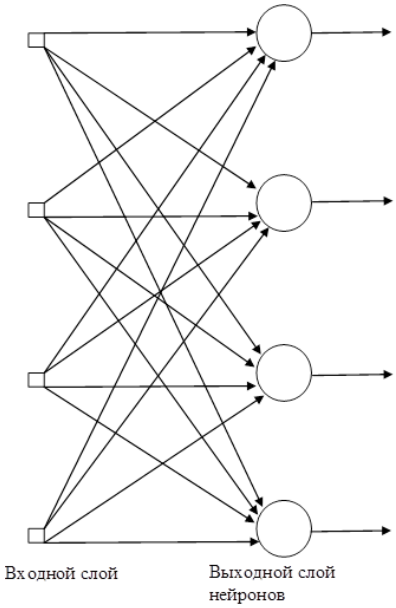


Рисунок 11.4 – Однослойная сеть прямого распространения

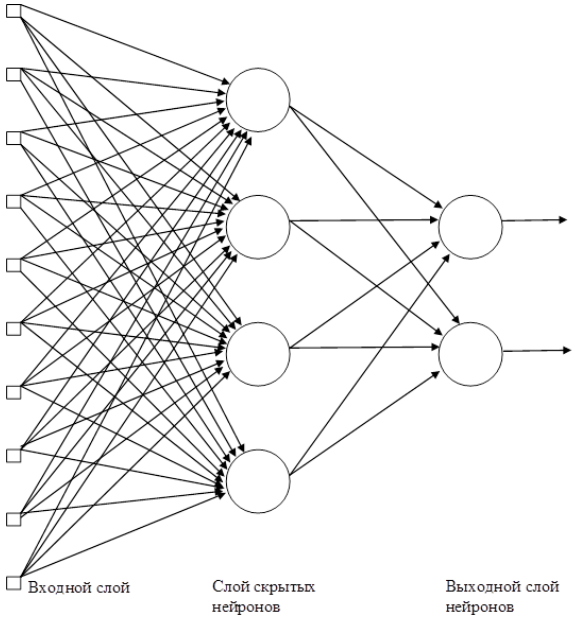


Рисунок 11.5 – Многослойная сеть прямого распространения

Элементы входного слоя сети формируют входные сигналы, поступающие на нейроны 2-го (скрытого) слоя и так далее, вплоть до выходного слоя. Обычно нейроны каждого из последующих слоев ИНС используют в качестве входных сигналов выходные сигналы нейронов только предыдущего слоя. Набор выходных сигналов нейронов выходного (последнего) слоя сети определяет общий отклик ИНС на вектор входных сигналов.

Нейронная сеть, показанная на рисунке 11.5, считается полносвязной в том смысле, что все нейроны каждого слоя соединены со всеми нейронами смежных слоев. Если отдельные синаптические связи отсутствуют, такая сеть считается неполносвязной.

Рекуррентная нейронная сеть отличается от сети прямого распространения наличием, по крайней мере, одной обратной связи. Например, рекуррентная сеть может состоять из единственного слоя нейронов, каждый из которых направляет свой выходной сигнал на входы всех остальных нейронов слоя (рисунок 11.6). На рисунке 11.7 показана рекуррентная сеть со слоем скрытых нейронов. Здесь обратные связи исходят как из скрытых, так и из выходных нейронов.

Наличие обратных связей в сетях оказывает непосредственное влияние на способность таких сетей к обучению и на их производительность. Более того, обратная связь подразумевает использование элементов единичной задержки (они обозначены как z^{-1}), что приводит к нелинейному динамическому поведению ИНС, если, конечно, в сети содержатся нелинейные нейроны.

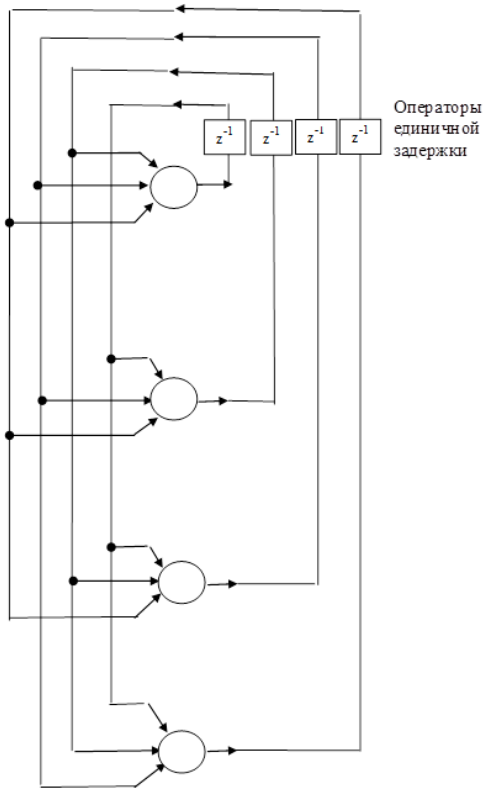


Рисунок 11.6 – Рекуррентная сеть без слоя скрытых нейронов

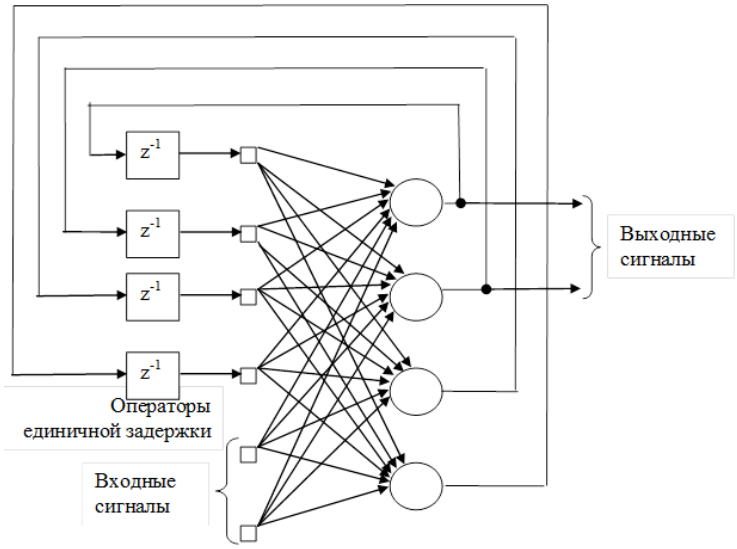


Рисунок 11.7 – Рекуррентная сеть со слоем скрытых нейронов

11.4 Обучение нейронных сетей

11.4.1 Основные понятия

Одним из основных свойств ИНС является их способность обучаться на основе поступающих данных. Обучение ИНС происходит посредством интерактивного процесса корректировки синаптических весов и других параметров ее нейронов. Тип обучения определяет способом подстройки этих параметров и, в обобщенном виде, предполагает следующую последовательность событий:

- в ИНС поступают воздействия из внешней среды;
- в результате этого изменяется внутренняя структура и параметры ИНС;
- после изменения внутренней структуры и параметров ИНС отвечает на внешние воздействия уже иным, нежели до обучения, образом.

Для обеспечения качественного обучения ИНС необходимо предварительно осуществить *нормализацию* исходных данных, т.е. приведение к некоторому диапазону, как правило, [0, 1] или [–1, 1].

Необходимость нормализации выборок данных обусловлена самой природой используемых входных переменных ИНС. Будучи разными по физическому смыслу, они зачастую могут сильно различаться между собой по абсолютным величинам. Нормализация данных позволяет привести все используемые числовые значения переменных к одинаковой области их изме-

нения, благодаря чему появляется возможность свести их вместе в единой нейросетевой модели.

Чтобы выполнить нормализацию данных, нужно точно знать пределы изменения значений соответствующих переменных (минимальное и максимальное возможные значения).

Наиболее распространенный способ нормализации входных и выходных переменных – *линейная нормализация*:

$$y = \frac{(x - x_{min})(d_2 - d_1)}{x_{max} - x_{min}} + d_1,$$

где x – значение, подлежащее нормализации; $[x_{\max}, x_{\min}]$ – интервал значений x; $[d_1, d_2]$ – интервал, к которому будет приведено значение x.

Пусть есть n входных данных из интервала [0, 10], тогда $x_{min} = 0$, а $x_{max} = 10$. Данные будем приводить к интервалу [0,1], тогда $d_1 = 0$, а $d_2 = 1$. Теперь, подставив все значения в формулу, можно вычислить нормализованные значения для любого x из n входных данных.

Примем следующие обозначения: x_{ik}, y_{jk} – i-е входное и j-е выходное значения k-го примера исходной выборки в традиционных единицах измерения, принятых в решаемой задаче; $\tilde{x}_{ik}$, $\tilde{y}_{jk}$ – соответствующие им нормализованные входное и выходное значения; N – количество примеров обучающей выборки.

Тогда переход от традиционных единиц измерения к нормализованным и обратно с использованием метода линейной нормализации осуществляется с использованием следующих расчетных соотношений:

- при нормализации и денормализации в диапазоне [0, 1]:

$$\tilde{x}_{ik} = \frac{x_{ik} - x_{mini_i}}{x_{max_i} - x_{mini_i}},$$

$$y_{jk} = y_{min_j} + \tilde{y}_{jk}\left(y_{max_j} - y_{min_j}\right),$$

- при нормализации и денормализации в диапазоне [–1, 1]:

$$\tilde{x}_{ik} = 2 \times \frac{x_{ik} - x_{mini_i}}{x_{max_i} - x_{mini_i}} - 1,$$

$$y_{jk} = y_{min_j} + \frac{(\tilde{y}_{jk} + 1)\left(y_{max_j} - y_{min_j}\right)}{2},$$

где $x_{min_i} = \min_{k=\overline{1,N}}(x_{ik})$; $x_{max_i} = \max_{k=\overline{1,N}}(x_{ik})$; $y_{min_j} = \min_{k=\overline{1,N}}(y_{jk})$; $y_{max_j} = = \max\limits_{k=\overline{1,N}}(y_{jk})$.

Если обучающая выборка не содержит примеров с потенциально возможными меньшими или большими выходными значениями, можно задаться шириной коридора экстраполяции для левой, правой или обеих границ в долях от длины всего первоначального интервала изменения переменной,

обычно не более 10 % от нее. В этом случае происходит переход от фактических границ из обучающей выборки к гипотетическим:

$$y_{min_j} = (\psi + 1)\min_{k=\overline{1,N}}(y_{jk}) - \psi \max_{k=\overline{1,N}}(y_{jk}),$$
$$y_{max_j} = (\psi + 1)\max_{k=\overline{1,N}}(y_{jk}) - \psi \min_{k=\overline{1,N}}(y_{jk}).$$

Еще один способ заключается в *нелинейной нормализации*, например, с использованием сигмоидной логистической функции или гиперболического тангенса. Переход от традиционных единиц измерения к нормализованным и обратно в данном случае осуществляется следующим образом:

- при нормализации и денормализации в диапазоне [0, 1]:

$$\tilde{x}_{ik} = \frac{1}{e^{-a(x_{ik}-x_{ci})} + 1},$$
$$y_{jk} = y_{cj} + \frac{1}{a}\ln\left(\frac{1}{\tilde{y}_{jk}} - 1\right),$$

где x_{ci}, y_{cj} – центры нормализуемых интервалов изменения входной и выходной переменных:

$$x_{ci} = \frac{x_{min_i} + x_{max_i}}{2},$$
$$y_{cj} = \frac{y_{min_j} + y_{max_j}}{2};$$

- при нормализации и денормализации в диапазоне [–1, 1]:

$$\tilde{x}_{ik} = \frac{e^{a(x_{ik}-x_{ci})} - 1}{e^{a(x_{ik}-x_{ci})} + 1},$$
$$y_{jk} = y_{cj} - \frac{1}{a}\ln\left(\frac{1 - \tilde{y}_{jk}}{1 + \tilde{y}_{jk}}\right).$$

Параметр a влияет на степень нелинейности изменения переменной в нормализуемом интервале. Кроме того, при использовании значений $a < 0{,}5$ нет необходимости дополнительно задаваться шириной коридора экстраполяции.

Возвращаясь к задаче обучения ИНС, отметим, что не существует универсального алгоритма обучения, подходящего для всех архитектур ИНС. Алгоритмы обучения отличаются способами взаимодействия ИНС с внешней средой и способами настройки синаптических весов нейронов.

Для того чтобы проиллюстрировать правило обучения, рассмотрим простейший случай k-го нейрона выходного слоя ИНС прямого распространения. Нейрон k работает под управлением вектора сигнала $x(n)$ (n – отсчет дискретного времени), поступающего со скрытого слоя нейронов, которые, в свою очередь, получают информацию из входного вектора, передаваемого нейронами входного слоя. Выходной сигнал k-го нейрона – $y_k(n)$. Он сравни-

вается с желаемым выходным значением $d_k(n)$, в результате чего получается сигнал ошибки $e_k(n)$:

$$e_k(n) = d_k(n) - y_k(n).$$

Сигнал ошибки инициализирует механизм управления (*control mechanism*), цель которого является применение последовательности корректирующих воздействий на синаптические веса k-го нейрона. Эти воздействия нацелены на пошаговое приближение значения $y_k(n)$ к желаемому $d_k(n)$.

Это цель достигается за счет минимизации функции ошибки (стоимости) $E(n)$, определяемой следующим образом:

$$E(n) = \frac{1}{2} e_k^2 n.$$

Пошаговая корректировка синаптических весов продолжается до тех пор, ИНС не достигнет устойчивого состояния, при котором синаптические веса практически стабилизируются. Процесс обучения на этом завершается.

Процесс, описанный выше, называется *обучением, основанным на коррекции ошибок*. Минимизация функции ошибки $E(n)$ выполняется по так называемому дельта-правилу, или правилу Видроу–Хоффа. Обозначим $w_{kj}(n)$ текущее значение синаптического веса w_{kj} нейрона k, соответствующего элементу $x_j(n)$ вектора $x(n)$, на шаге дискретизации n.

В соответствии с дельта-правилом изменение $\Delta w_{kj}(n)$, применяемое к синаптическому весу w_{kj} на этом шаге n, задается выражением:

$$\Delta w_{kj}(n) = \eta e_k(n) x_j(n),$$

где η – некоторая положительная константа, определяющая скорость обучения и используемая при переходе от одного шага процесса к другому.

Дельта-правило обучения ИНС формулируется следующим образом: корректировка, применяемая к синоптическому весу нейрона, пропорциональна произведению сигнала ошибки на входной сигнал, его вызвавший.

Дельта-правило предполагает возможность прямого измерения сигнала ошибки. Из рисунка 11.8 следует, что обучение на основе коррекции ошибки является локальнымдля в отдельного нейрона.

Вычислив значение изменения синаптического веса $\Delta w_{kj}(n)$, можно определить его новое значение для следующего шага обучения:

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n).$$

Можно записать:

$$w_{kj}(n) = z^{-1}\left[w_{kj}(n+1)\right],$$

где z^{-1} – оператор единичной задержки, представляющий собой элемент памяти.

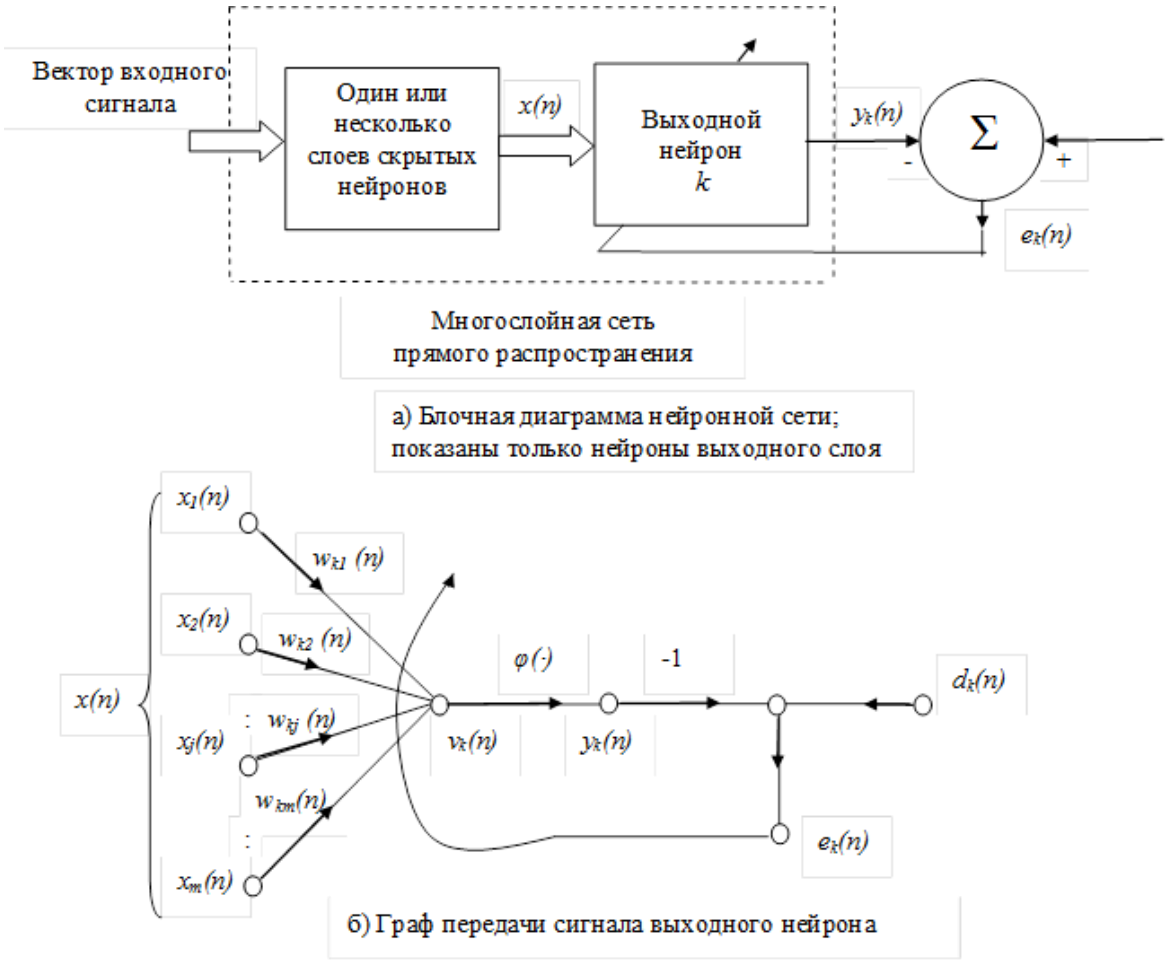


Рисунок 11.8 – Схема и граф передачи сигнала при обучении нейрона

Входной сигнал $x_к$ и индуцированное локальное поле v_k нейрона k представлены в виде предсинаптического и постсинаптического сигналов j-го синапса k-го нейрона.

На рисунке 11.8 видно, что обучение на основе коррекции ошибок — это пример замкнутой системы с обратной связью, устойчивость которой определяется параметрами этой обратной связи. В данном случае имеется одна обратная связь, а интересующим нас параметром является коэффициент скорости обучения η. Для обеспечения устойчивости или сходимости итеративного процесса обучения требуется тщательный подбор этого параметра.

11.4.2 Обучение на основе памяти

При *обучении на основе памяти* весь прошлый опыт накапливается в хранилище правильно классифицированных примеров вида вход-выход: $\{(x_i, d_i)\}^N_{i=1}$, где x_i – входной вектор, а d_i – соответствующий ему желаемый выходной сигнал.

Например, рассмотрим задачу бинарного распознавания образов или классификации на два класса, C_1 и C_2. В этом примере желаемый отклик системы d_i принимает значение 0 (или –1) для класса C_1 и значение +1 для

класса C_2. Если требуется классифицировать некоторый неизвестный вектор x_{test}, из базы данных выбирается выход, соответствующий входному сигналу, близкому к x_{test}.

Все алгоритмы обучения на основе памяти основываются на:

- критерии, используемом для определения окрестности вектора x_{test};
- правиле обучения, применяемом к примеру из окрестности тестового вектора.

В простейшем алгоритме обучения на основе памяти, получившем название *правила ближайшего соседа*, в окрестность включается пример, ближайший к тестовому.

Например, вектор

$$x'_N \in \{x_1, x_2, \ldots, x_N\}$$

считается ближайшим соседом вектора x_{test}, если выполняется условие

$$\min_i d(x_i, x_{test}) = d(x'_N, x_{test}),$$

где $d(x_i, x_{test})$ – евклидово расстояние между векторами x_i и x_{test}.

Класс, к которому относится ближайший сосед, считается также классом тестируемого вектора x_{test}. Это правило не зависит от распределения, используемого при генерировании примеров обучения.

При этом анализ основывается на двух следующих предположениях:

- классифицируемые примеры (x_i, d_i) независимы и равномерно распределены в соответствии с совместным распределением примера (x, d);
- число примеров обучающей выборки N бесконечно велико.

При этих двух предположениях вероятность ошибки классификации при использовании правила ближайшего соседа вдвое превышает *байесовскую вероятность ошибки*, т.е. минимальную вероятность ошибки на множестве всех правил принятия решения.

В этом контексте можно считать, что половина классификационной информации для обучающего множества бесконечного размера содержится в данных о ближайшем соседе. Это довольно неожиданный результат.

На рисунке 11.9 область в штриховой окружности содержит две точки, принадлежащие классу 1, и одну, принадлежащую классу 0. Точка d соответствует тестируемому вектору x_{test}. При $k = 3$ классификатор на основе метода k-ближайших соседей отнесет точку d к классу 1, несмотря на то, что она лежит ближе всего к «выбросу», относящемуся к классу 0.

Вариацией классификатора на основе ближайшего соседа является *классификатор k-ближайших соседей*. Он описывается следующим образом.

Находим k классифицированных соседей, ближайших к вектору x_{test}, где k – некоторое целое число. Вектор x_{test} относим к тому классу, который чаще других встречается среди k-ближайших соседей тестируемого вектора.

Таким образом, классификатор на основе k-ближайших соседей работает подобно устройству усреднения.

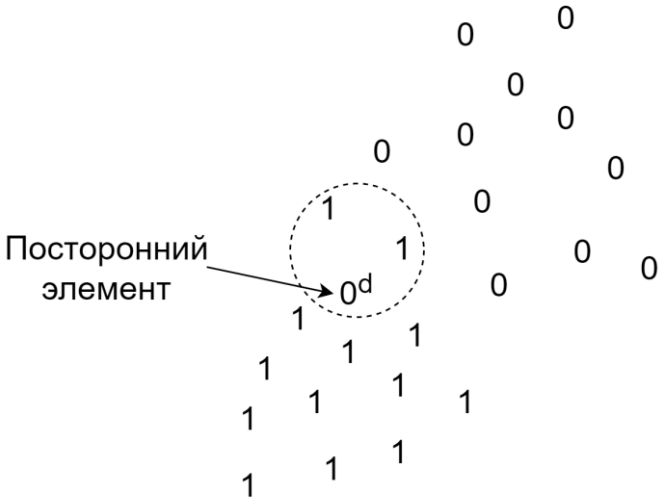


Рисунок 11.9 –Иллюстрация правила ближайшего соседа

11.4.3 Обучение Хебба

Постулат обучения Хебба является самым известным среди всех правил обучения: «если аксон клетки *A* находится на достаточно близком расстоянии от клетки *B* и постоянно или периодически участвует в ее возбуждении, наблюдается процесс метаболических изменений в одном или обоих нейронах, выражающийся в том, что эффективность нейрона *A* как одного из возбудителей нейрона *B* возрастает».

Хебб предложил положить это наблюдение в основу процесса ассоциативного обучения (на клеточном уровне), что должно было привести к постоянной модификации шаблона активности пространственно-распределенного «ансамбля нервных клеток». Это утверждение было сделано в нейробиологическом контексте, но его можно перефразировать в следующее правило, состоящее из двух частей:

- если два нейрона по обе стороны синапса (соединения) активизируются одновременно (т.е. синхронно), то прочность этого соединения возрастает;
- если два нейрона по обе стороны синапса активизируются асинхронно, то такой синапс ослабляется или вообще отмирает.

Функционирующий таким образом синапс называется *синапсом Хебба*. Если быть более точным, то синапс Хебба использует зависящий от времени, локальный механизм взаимодействия, изменяющий эффективность синаптического соединения в зависимости от корреляции между предсинаптической и постсинаптической активностью.

Из этого определения можно вывести следующие ключевые свойства синапса Хебба.

- *Зависимость от времени*. Синапс Хебба зависит от точного времени возникновения предсинаптического и постсинаптического сигналов.

- *Локальность*. По своей природе синапс является узлом передачи данных, в котором информационные сигналы (представляющие текущую активность предсинаптических и постсинаптических элементов) находятся в пространственно-временной близости. Эта локальная информация используется синапсом Хебба для выполнения локальных синаптических модификаций, характерных для данного входного сигнала.
- *Интерактивность*. Изменения в синапсе Хебба определяются сигналами на обоих его концах. Это значит, что форма обучения Хебба зависит от степени взаимодействия предсинаптического и постсинаптического сигналов (предсказание нельзя построить на основе только одного из этих сигналов). Заметим, что такая зависимость или интерактивность может носить детерминированный или статистический характер.
- *Корреляция*. Условием изменения эффективности синаптической связи является зависимость между предсинаптическим и постсинаптическим сигналами. Для модификации синапса необходимо обеспечить одновременность предсинаптического и постсинаптического сигналов. Именно по этой причине синапс Хебба иногда называют конъюнктивным синапсом.

Другая интерпретация постулата обучения Хебба использует характерные для синапса Хебба механизмы взаимодействия в статистических терминах. В частности, синаптические изменения определяются корреляцией предсинаптического и постсинаптического сигналов во времени. Учитывая это, синапс Хебба иногда называют *корреляционным синапсом*. Сама корреляция является основой обучения.

Приведенное выше определение синапса Хебба не учитывает дополнительные процессы, которые могут привести к ослаблению синаптической связи между парой нейронов. Исходя из этого, можно несколько обобщить концепцию модификации связей Хебба, используя тот факт, что положительно коррелированное функционирование приводит к усилению синаптической связи, а отрицательная корреляция или ее отсутствие – к ее ослаблению.

Ослабление синаптической связи может также носить и не интерактивный характер. В частности, условие взаимодействия для ослабления синаптической связи может носить случайный характер и не зависеть от предсинаптического и постсинаптического сигналов.

Систематизируя эти определения, можно выделить следующие модели модификации синаптических связей: *хеббовскую*, *антихеббовскую* и *нехеббовскую*. Следуя этой схеме, можно сказать, что синаптическая связь по модели Хебба усиливается при положительной корреляции предсинаптических и постсинаптических сигналов и ослабляется в противном случае.

И, наоборот, согласно антихеббовской модели, синаптическая связь ослабляется при положительной корреляции предсинаптического и постсинаптического сигналов и усиливается в противном случае. Однако в обеих

моделях изменение синаптической эффективности основано на зависящем от времени в высшей степени локальном и интерактивном по своей природе механизме.

В этом контексте антихеббовская модель все равно остается хеббовской по природе, но не по функциональности. Не-хеббовская модель вообще не связана с механизмом модификации, предложенным Хеббом.

Для того чтобы описать обучение Хебба в математических терминах, рассмотрим синаптический вес w_{kj} k-го нейрона с предсинаптическим и постсинаптическим сигналами x_j и $y_к$.

Изменение синаптического веса w_{kj} в момент времени n можно выразить следующим соотношением:

$$\Delta w_{kj}(n) = F\big(y_k(n), x_j(n)\big),$$

где F – некоторая функция, зависящая от предсинаптического и постсинаптического сигналов. Сигналы $x_j(n)$ и $y_к(n)$ часто рассматривают без учета размерности. Формула может быть записана в разных формах, каждая из которых все равно считается хеббовской.

Простейшая форма обучения Хебба имеет следующий вид:

$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n),$$

где η – положительная константа, определяющая скорость обучения.

Выражение подчеркивает корреляционную природу синапса Хебба. Ее иногда называют *правилом умножения активности*. Оно представляет зависимость изменения Δw_{kj}(n) от выходного сигнала (постсинаптической активности) $y_k(n)$.

На рисунке 11.20 видно, что регулярное приложение входного сигнала (предсинаптической активности) $x_j(n)$ ведет к увеличению $y_k(n)$ и *экспоненциальному росту* активности, приводящему к насыщению синаптической связи. В этой точке синапс уже не содержит никакой информации, и избирательность связей теряется.

Одним из способов преодоления ограничений гипотезы Хебба является использование *гипотезы ковариации*, согласно которой предсинаптический и постсинаптический сигналы в предыдущем выражении заменяются отклонениями этих сигналов от их средних значений на данном отрезке времени. Обозначим символами $\bar{x}$ и $\bar{y}$ *усредненные по времени значения* предсинаптического x_j и постсинаптического y_k сигналов.

Согласно гипотезе ковариации, изменение синаптического веса w_{kj} вычисляется по формуле

$$\Delta w_{kj} = \eta\big(x_j - \bar{x}\big)(y_k - \bar{y}).$$

Средние значения $\bar{x}$ и $\bar{y}$ содержат предсинаптический и постсинаптический пороги, определяющие знак синаптической модификации.

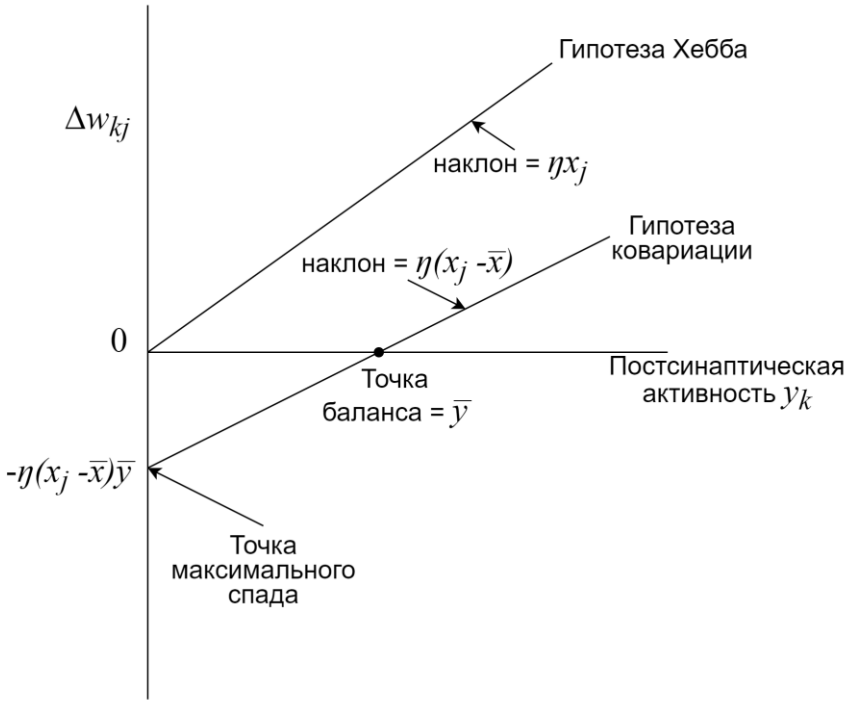


Рисунок 11.20 – Гипотеза Хебба, гипотеза ковариации

В частности, гипотеза ковариации обеспечивает:

- сходимость к нетривиальному состоянию, которое достигается при $x_k = x$ и $y_j = y$;
- прогнозирование усиления и ослабления синаптической связи.

На рисунке 11.20 показано отличие гипотезы Хебба от гипотезы ковариации. В обоих случаях зависимость величины Δw_{kj} от y_k является линейной, однако пересечение с осью y_k для гипотезы Хебба происходит в начале координат, а для гипотезы ковариации – в точке $y_k = y$

Можно сделать следующие выводы:

- синаптический вес связи усиливается при высоком уровне предсинаптического и постсинаптического сигналов, т.е. в том случае, когда одновременно удовлетворяются следующие условия: $x_j > x$ и $y_k > y$;
- синаптический вес ослабляется в следующих ситуациях: во-первых, предсинаптическая активность (т.е. $x_j > x$) не вызывает существенной постсинаптической активности ($y_k < y$); во-вторых, постсинаптическая активность ($y_k > y$) возникает при отсутствии существенной предсинаптической активности (т.е. $x_j < x$).

Такое поведение можно рассматривать как форму временно*й* конкуренции между входами.

11.4.4 Конкурентное обучение

В *конкурентном обучении* выходные нейроны нейронной сети конкурируют между собой за право быть активизированными. Если в нейронной сети, основанной на обучении Хебба, одновременно в возбужденном состоянии может находиться несколько нейронов, то в конкурентной сети в каждый момент времени может быть активным только один нейрон. Благодаря этому свойству конкурентное обучение очень удобно использовать для изучения статистических свойств в задачах классификации образов.

Правило конкурентного обучения основано на использовании трех основных элементов:

- множество одинаковых нейронов со случайно распределенными синаптическими весами, приводящими к различной реакции нейронов на один и тот же входной сигнал;
- предельное значение «силы» каждого нейрона;
- механизм, позволяющий нейронам конкурировать за право отклика на данное подмножество входных сигналов и определяющий единственный активный выходной нейрон (или по одному нейрону на группу). Принцип конкурентного обучения – «победитель забирает все».

Таким образом, каждый отдельный нейрон сети соответствует группе близких образов. При этом нейроны становятся *детекторами признаков* различных классов входных образов.

Простейшая нейронная сеть с конкурентным обучением содержит единственный слой выходных нейронов, каждый из которых соединен с входными узлами. В такой сети могут существовать обратные связи между нейронами. В подобной архитектуре обратная связь обеспечивает *латеральное торможение*, когда каждый нейрон стремится «затормозить» связанные с ним нейроны. Прямые синаптические связи в сети, являются *возбуждающими* (рисунок 11.21).

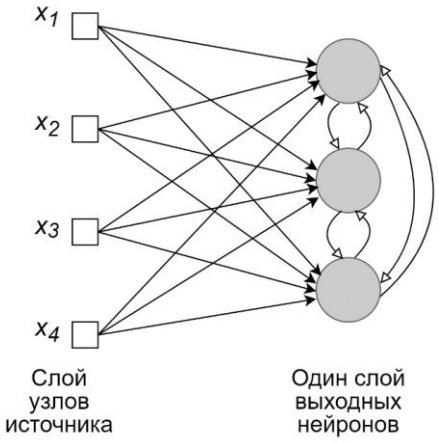


Рисунок 11.21 – Сеть с конкурентным обучением

Для того чтобы k-й нейрон победил в конкуренции, его индуцированное локальное поле v_k для заданного входного образа должно быть максимальным среди всех нейронов сети. Тогда выходной сигнал y_k нейрона-победителя принимается равным единице. Выходные сигналы остальных нейронов при этом устанавливаются в нуль.

Таким образом,

$$y_k = \begin{cases} 1, \text{если } v_k > v_j \text{ для всех } j, j \neq k, \\ 0 \text{ в остальных случаях,} \end{cases}$$

где индуцированное локальное поле v_k представляет сводное возбуждение нейрона k от всех входных сигналов и сигналов обратной связи.

Пусть w_{kj} – синаптический вес связи входного j-го нейрона с k-м нейроном. Предположим, что синаптические веса всех нейронов фиксированы (т.е. положительны), при этом

$$\sum_j w_{kj} = 1 \text{ для всех } k.$$

Тогда обучение этого нейрона состоит в смещении синаптических весов от неактивных к активным входным нейронам.

Если нейрон не формирует отклика на конкретный входной образ, то он и не обучается. Если некоторый нейрон выигрывает в конкурентной борьбе, то веса связей этого нейрона равномерно распределяются между его активными входными узлами.

Согласно *правилу конкурентного обучения* (*competitive learning rule*) изменение Δw_{kj} синаптического веса w_{kj} определяется следующим выражением:

$$\Delta w_{kj} - \begin{cases} \eta(x_j - w_{kj}), \text{если нейрон } k \text{ побеждает в соревновании,} \\ 0, \text{если нейрон } k \text{ проигрывает в соревновании.} \end{cases}$$

Это правило отражает смещение вектора синаптического веса w_k победившего k-го нейрона в сторону входного образа x. Точками на рисунке 11.22 представлены входные векторы, а крестиками – векторы синаптических весов трех выходных нейронов в исходном (слева) и конечном (справа) состоянии сети.

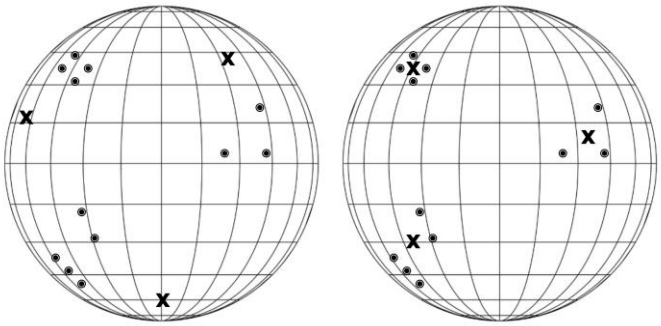


Рисунок 11.22 – Геометрическая интерпретация процесса конкурентного обучения

Предполагается, что все входные образы (векторы) x имеют постоянную евклидову норму и их можно изобразить в виде точек на N-мерной единичной сфере, где N – число входных нейронов, соответствующее размерности вектора синаптических весов w_k.:

$$\sum_j w_{kj}^2 = 1 \text{ для всех } k.$$

Если синаптические веса правильно масштабированы, они формируют набор векторов, которые проецируются на ту же N-мерную единичную сферу. На рисунке11.22 можно выделить три естественные группы (кластеры) точек, представляющие входные образы. На этом рисунке также показано вероятное начальное состояние сети (отмечено крестиками) до начала обучения, а также конечное состояние сети, полученное в результате конкурентного обучения, в котором синаптические веса каждого выходного нейрона смещены к центрам тяжести соответствующих кластеров.

11.4.5 Обучение с учителем, алгоритм обратного распространения ошибки

Участие учителя можно рассматривать как наличие знаний об окружающей среде, представленных в виде примеров *вход-выход* (рисунок 11.23).

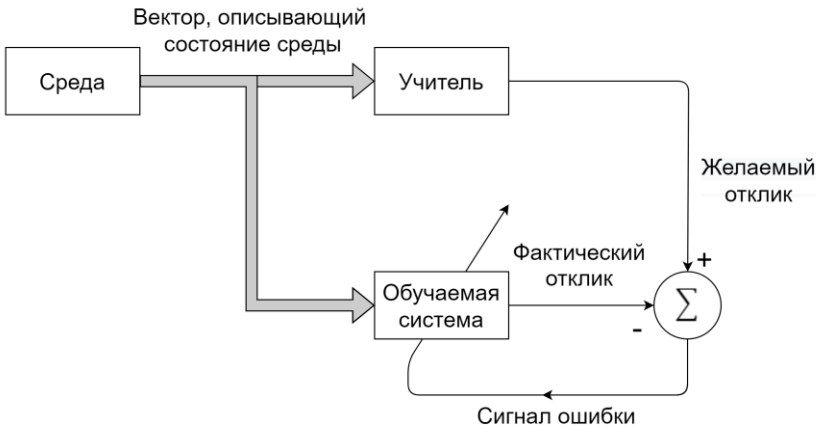


Рисунок 11.23 – Обучение с учителем

Теперь предположим, что учителю и обучаемой ИНС предъявляется обучающий пример. На основе встроенных знаний учитель может сформировать и передать обучаемой ИНС желаемый отклик, соответствующий данному входному вектору. Параметры ИНС корректируются с учетом обучающего вектора и сигнала ошибки. *Сигнал ошибки* – это разность между желаемым сигналом и текущим откликом ИНС.

Таким образом, в процессе обучения знания учителя передаются в сеть в максимально полном объеме. После окончания обучения учителя можно отключить и позволить ИНС работать самостоятельно.

Описанная форма обучения с учителем является ничем иным, как *обучением на основе коррекции ошибок*. Это замкнутая система с обратной связью, которая не включает в себя окружающую среду.

Эффективность такой системы можно оценивать в терминах среднеквадратической ошибки или суммы квадратов ошибок на обучающей выборке, представленной в виде функции от свободных параметров системы. Для такой функции можно построить многомерную *поверхность ошибки* в координатах свободных параметров. При этом реальная поверхность ошибки *усредняется* по всем возможным примерам, представленным в виде пар «вход-выход».

Любое конкретное действие системы с учителем представляется одной точкой на поверхности ошибок. Для повышения эффективности системы во времени значение ошибки должно смещаться в сторону минимума на поверхности ошибок. Этот минимум может быть как локальным, так и глобальным. Это можно сделать, если имеется полезная информация о *градиенте поверхности ошибок* – векторе, определяющем направление наискорейшего спуска по этой поверхности. В случае обучения с учителем на примерах вычисляется *моментальная оценка* вектора градиента, в которой входной вектор считается функцией времени. При использовании результатов такой оценки перемещение точки по поверхности ошибок обычно имеет вид «случайного блуждания». Тем не менее, при использовании соответствующего алгоритма минимизации функции ошибки, адекватном наборе обучающих примеров в форме «вход-выход» и достаточном времени для обучения ИНС способны эффективно решать такие задачи, как классификация образов и аппроксимация функций.

Алгоритм *обратного распространения ошибки (error back propagation)* – это итеративный градиентный алгоритм обучения для минимизации среднеквадратичного отклонения текущих от требуемых выходов многослойной ИНС. В алгоритме вычисляется вектор градиента поверхности ошибок, который указывает направление кратчайшего спуска по поверхности из данной точки для уменьшения ошибки. На рисунке 11.24 приведен пример обучаемой ИНС, а на рисунке 11.25 – пример поверхности ошибки ИНС.

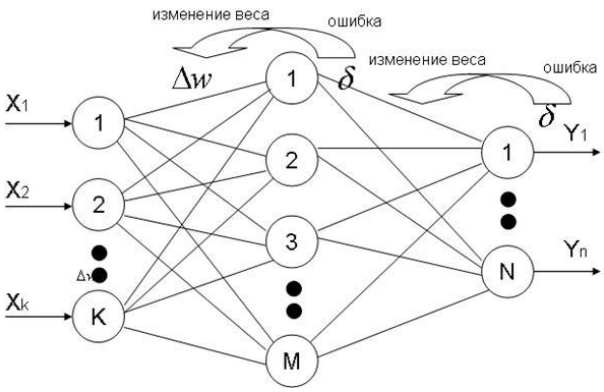


Рисунок 11.24 – Структура обучаемой ИНС

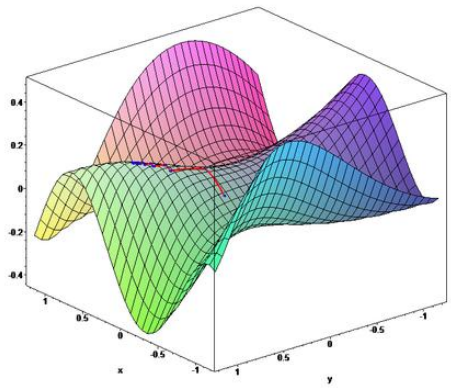


Рисунок 11.25 – Поверхность ошибки ИНС

Алгоритм обратного распространения ошибки состоит из следующих шагов.

Шаг 1. Подать на входы сети один из образов и в режиме функционирования сети, когда сигналы распространяются от входов к выходам, рассчитать значения выходов. Напомним, что:

$$s_j^{(q)} = \sum_{i=0}^{L} y_i^{(q-1)} w_{ij}^{(q)},$$

где L – число нейронов в слое $(q-1)$; $y_i^{(q-1)} w_{ij}^{(q)}$ – i-й вход нейрона j слоя q,

$$y_j^{(q)} = f(s_j^{(q)}),$$

где $f(\bullet)$ – сигмоид,

$$y_r^{(0)} = x_r,$$

где x_r – r-я компонента вектора входного образа.

Шаг 2. Рассчитать $\delta^{(Q)}$ для выходного слоя: $\delta_j^{(Q)} = (y_j^{(Q)} - d_j)\dfrac{dy_j}{ds_j}$.

Рассчитать изменения весов $\Delta w^{(q)}$ слоя q:

$$\Delta w_{ij}^{(q)} = -\eta \;\; \delta_j^{(q)} \;\; y_i^{(q-1)}.$$

Шаг 3. Рассчитать $\delta^{(q)}$ и $\Delta w^{(q)}$ для всех остальных слоев, $q = (Q-1), \ldots, 1$.

$$\delta_j^{(q)} = \left(\sum_r \delta_r^{(q+1)} \;\; w_{jr}^{(q+1)} \right) \frac{dy_j}{ds_j}, \qquad \Delta w_{ij}^{(q)} = -\eta \;\; \delta_j^{(q)} \;\; y_i^{(q-1)}.$$

Шаг 4. Скорректировать все веса нейронов в ИНС:

$$w_{ij}^{(q)}(t) = w_{ij}^{(q)}(t-1) + \Delta w_{ij}^{(q)}(t).$$

Шаг 5. Если ошибка сети существенна, перейти на шаг 1. В противном случае – конец.

Нейронной сети на шаге 1 попеременно в случайном порядке предъявляются все образы, чтобы сеть не забывала одни по мере запоминания других.

Сложности алгоритма обратного распространения ошибки

Блокировка сети. Алгоритм не эффективен, если производные по различным весам сильно отличаются. В этом случае для придания процессу коррекции весов инерционности, сглаживающей резкие скачки при перемещении по поверхности целевой функции, выражение для изменения весов дополняется значением изменения веса на предыдущей итерации:

$$\Delta w_{ij}^{(q)}(t) = -\eta \;\; (\mu \;\; \Delta w_{ij}^{(q)}(t-1) + (1-\mu) \;\; \delta_j^{(q)} \;\; y_i^{(q-1)}),$$

где μ – коэффициент инерционности; t – номер текущей итерации.

Медленная сходимость процесса обучения. Сходимость алгоритма доказана для дифференциальных уравнений, т.е. для бесконечно малых шагов в пространстве весов. Но это означают бесконечно большое время обучения.

Переобучение. Высокая точность, получаемая при обучении, может привести к ухудшению способности сети к обобщению и экстраполяции. Средство борьбы с этим – использование, помимо обучающей и тестовой, еще и подтверждающей выборки.

Попадание в локальные минимумы. Алгоритм не всегда обнаруживает глобальный минимум или не может выйти из него. Способом обхода «локальных ловушек» является расширение размерности пространства весов за счет увеличения скрытых слоев и числа нейронов скрытого слоя. Другой способ – использование эвристических алгоритмов оптимизации, например, генетических алгоритмов.

На рисунке 11.26 проиллюстрирована процедура обучения многослойного персептрона на основе алгоритма обратного распространения ошибки.

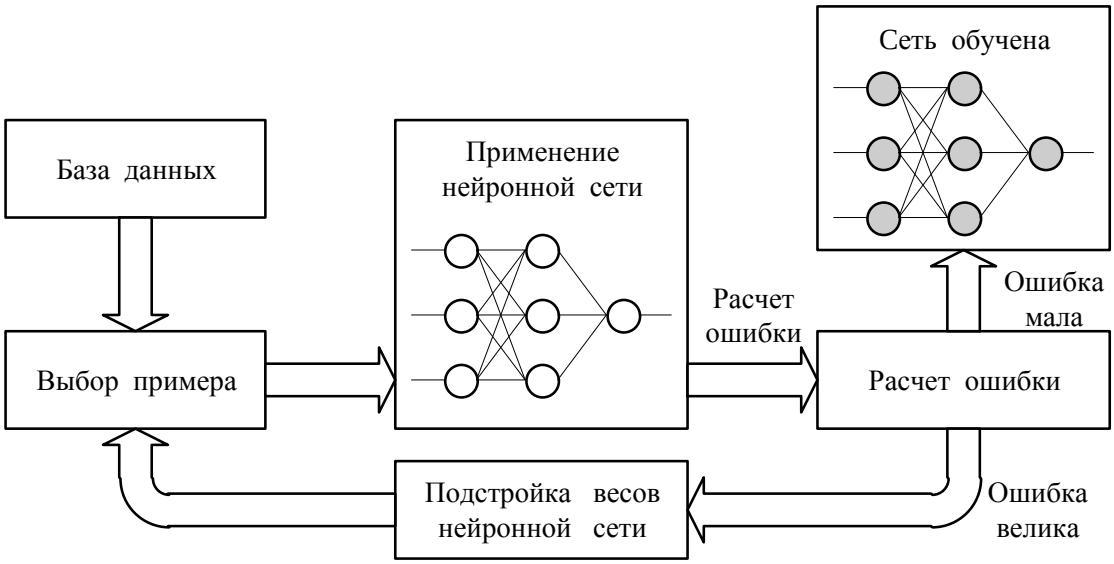


Рисунок 11.26 – Иллюстрация алгоритма обратного распространения ошибки

Существуют варианты данного алгоритма, позволяющие увеличить скорость обучения ИНС. Например, к такому варианту относится алгоритм линейного поиска, действующий следующим образом. Выбирается какое-либо разумное направление движения по многомерной поверхности. В этом направлении проводится линия, и на ней ищется точка минимума, затем все повторяется сначала. Очевидно, разумным направлением является направление скорейшего спуска.

На самом деле этот вроде бы очевидный выбор может быть не слишком удачным. После того, как был найден минимум по некоторой прямой, следующая линия, выбранная для кратчайшего спуска, может «испортить» результаты минимизации по предыдущему. Более разумно было бы выбирать «не мешающие друг другу» направления спуска. Так приходим к *алгоритму сопряженных градиентов* (рисунок 11.27), идея которого состоит в следующем. Поскольку нашли точку минимума вдоль некоторой прямой, производная по этому направлению равна нулю. Сопряженное направление выбирается та-

ким образом, чтобы эта производная и дальше оставалась нулевой. Тогда для достижения точки минимума достаточно будет N эпох.

На реальных поверхностях ошибки по мере реализации алгоритма условие сопряженности «портится», и, тем не менее, такой алгоритм, как правило, требует меньшего числа шагов, чем алгоритм обратного распространения.

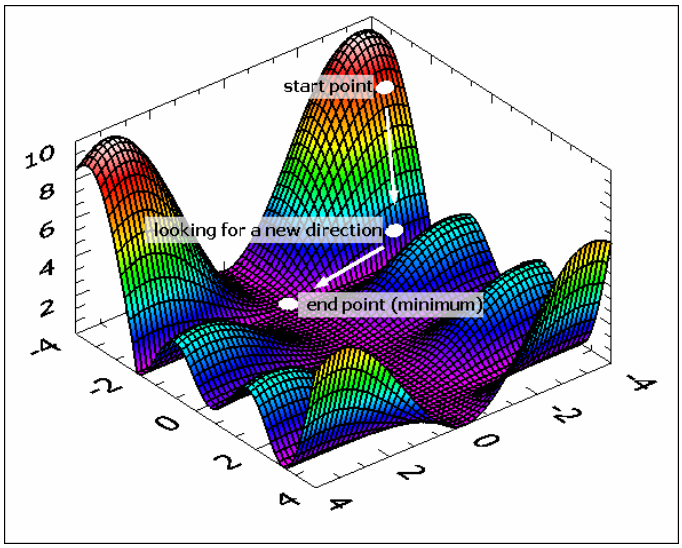


Рисунок 11.27 – Поверхность ошибки. Алгоритм сопряжённых градиентов

При всех преимуществах, данный метод имеет серьёзный недостаток – неустойчивость к избыточным либо повторяющимся данным. Данная проблема решается путем ужесточения требований к наборам данных для обучения ИНС, но в любом случае стоит отметить, что алгоритм обратного распространения ошибки лишён данного недостатка.

11.4.6 Обучение без учителя, самоорганизующиеся карты Кохонена

Альтернативная парадигма обучения без учителя самим названием подчеркивает отсутствие руководителя, контролирующего процесс настройки весовых коэффициентов. При использовании такого подхода не существует маркированных примеров, по которым проводится обучение сети.

Обучение без учителя (или *обучение на основе самоорганизации*) осуществляется без вмешательства внешнего учителя, или корректора, контролирующего процесс обучения.

Для обучения без учителя можно воспользоваться правилом конкурентного обучения. Например, можно использовать ИНС, состоящую из двух слоев – входного и выходного. Входной слой получает доступные данные. Выходной слой состоит из нейронов, конкурирующих друг с другом за право отклика на признаки, содержащиеся во входных данных. В простейшем случае ИНС действует по принципу «победитель получает все». Как было показано, при такой стратегии нейрон с наибольшим суммарным входным сигна-

лом «побеждает» в соревновании и переходит в активное состояние. При этом все остальные нейроны отключаются.

В следующем подразделе будут проиллюстрированы основные алгоритмы обучения без учителя на примерах различных разновидностей ИНС.

Самоорганизующиеся карты Кохонена (Self-Organization Maps) основаны на воспроизведении топологических свойств мозга, т.е. при обработке многомерного входного образа осуществляется его проецирование на нейронную сеть, представляющую собой решетку из нейронов меньшей размерности, с сохранением топологии многомерного входного образа;

Топологически сеть Кохонена представляет собой решетку из нейронов, расположенных в вершинах многогранников Вороного в 2d- или 3d-пространстве (рисунок 11.28). При этом смежные нейроны влияют друг на друга гораздо сильнее, чем удаленные. Для задания окрестностей нейронов используются гауссовы функции (рисунок 11.29).

В процессе самообучения на входы сети подаются данные, но сеть при этом подстраивается не под эталонное значение выхода, а под закономерности во входных данных.

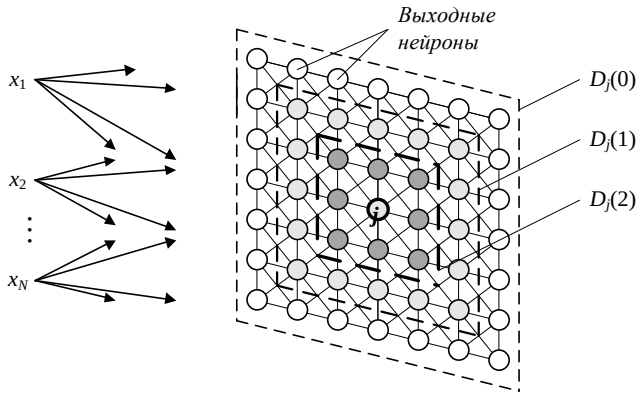


Рисунок 11.28 – Пример структуры сети Кохонена

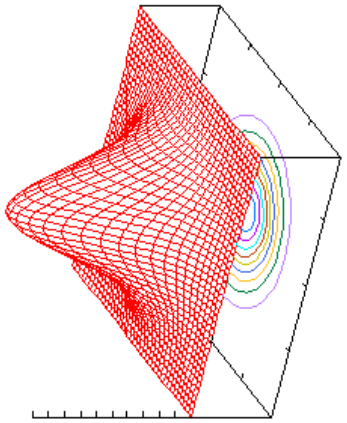


Рисунок 11.9 – Функция Гаусса для задания окрестности нейрона–победителя

Суть алгоритма обучения без учителя нейронной сети Кохонена заключается в размещении центров кластеров для совокупностей наблюдаемых объектов (образов) в нейронах решетки сети с последующей корректировкой положений этих кластеров путем подстройки весовых коэффициентов нейронов, соответствующих центрам этих кластеров, а также соседних с ними нейронов, в зависимости от вновь поступающих на вход сети данных.

Алгоритм самообучения сети Кохонена включает в себя следующие шаги.

Шаг 1. Инициализация сети: весовым коэффициентам сети, общее число которых равно MN, (M – число нейронов; N – размерность векторов) присваиваются малые случайные значения; $D_j(0)$ – начальная зона смежности нейронов.

Шаг 2. Предъявление сети нового входного сигнала $\mathbf{X} = \{x_1, \ldots, x_N\}$.

Шаг 3. Вычисление расстояния d_j от входного сигнала до каждого нейрона j:

$$d_j = \sqrt{\sum_{i=1}^{N} \left(x_i(t) - w_{ij}(t) \right)^2},$$

где $x_i(t)$ – i-й элемент входного сигнала в момент времени t; $w_{ij}(t)$ – вес связи от i-го элемента входного сигнала к нейрону j.

Шаг 4. Выбор нейрон j^*, для которого расстояние d_j является наименьшим.

Шаг 5. Настройка весов для нейрона j^* и всех нейронов из его зоны соседства $D_{j*}(t)$:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \left(x_i(t) - w_{ij}(t) \right).$$

где $\eta \in [0, 1]$ – шаг обучения, уменьшающийся с течением времени до нуля.

Шаг 6. Возвращение к шагу 2. Повторение алгоритма до стабилизации результата кластеризации.

На рисунках 11.30–11.32 приведены иллюстрации ИНС Кохонена.

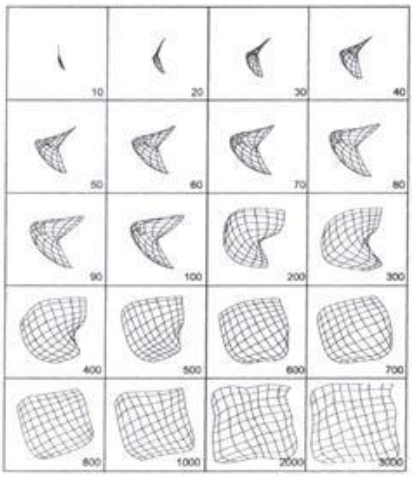


Рисунок 11.30 – Пример итеративного построения ИНС Кохонена

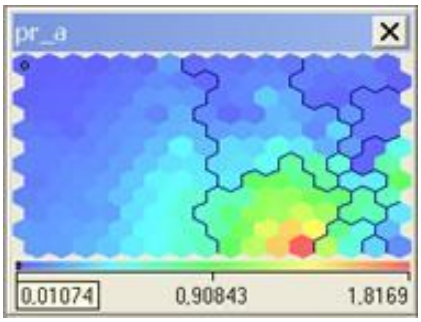


Рисунок 11.31 – Пример «раскраски» ИНС Кохонена в 2d-пространстве относительно отдельного признака

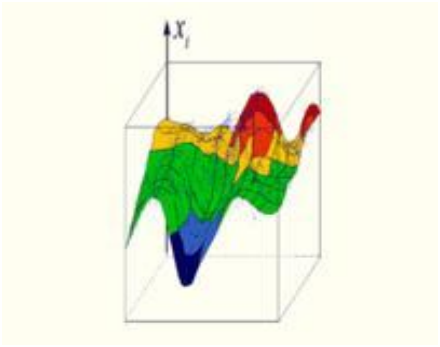


Рисунок 11.32 – Пример «раскраски» ИНС Кохонена в 3d-пространстве относительно отдельного признака

Проблемы ИНС Кохонена, пути их решения:

- так как часто побеждают (получают аккредитацию) одни и те же нейроны с наименьшими расстояниями, то они принудительно исключаются из рассмотрения, чтобы уравнять права всех нейронов;
- для сокращения длительности обучения на этапе инициализации нормализуются входные образы и начальные значения весовых коэффициентов нейронов:

$$x_i' = x_i \Big/ \sqrt{\sum_{j=1}^{N} x_j^2}, \quad w_0 = \frac{1}{\sqrt{N}},$$

- инициализация весовых коэффициентов случайными значениями может привести к тому, что классы, которым соответствуют плотно распределенные входные образы, сольются или, наоборот, раздробятся на отдельные подклассы в случае близких образов одного и того же класса. Поэтому используют метод выпуклой комбинации для преобразования входные нормализованных образов:

$$x_i = \alpha(t)\, x_i + (1 - \alpha(t))\, \frac{1}{\sqrt{n}},$$

где $\alpha(t)$ – коэффициент, изменяющийся при обучении от –1 до 1, в результате чего вначале на входы сети подаются практически одинаковые образы, а с течением времени они все больше сходятся к исходным.

11.4.7 Обучение с подкреплением

В *обучении с подкреплением* формирование отображения входных сигналов в выходные выполняется в процессе взаимодействия с внешней средой с целью минимизации индекса эффективности (рисунок 11.33).

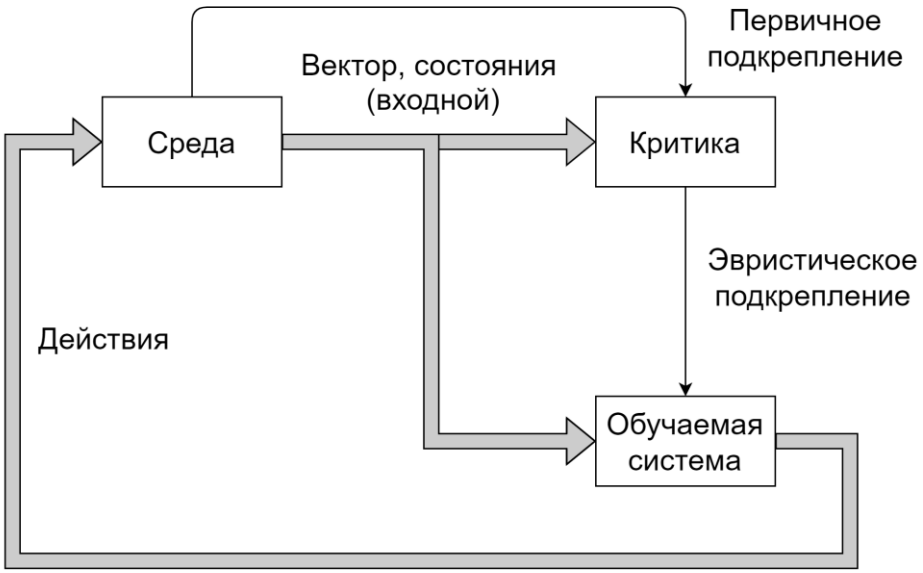


Рисунок 11.33 – Обучение с подкреплением

На рисунке 11.14 показана диаграмма одного из способов обучения с подкреплением, включающая блок «критики», который преобразовывает *первичный сигнал подкрепления*, полученный из внешней среды, в сигнал более высокого уровня, называемый *эвристическим сигналом подкрепления*.

Такой способ предполагает *обучение с отложенным подкреплением*. Это значит, что ИНС получает из внешней среды последовательность сигналов возбуждения (т.е. векторов состояния), которые приводят к генерации эвристического сигнала подкрепления.

Целью данного обучения является минимизация *функции стоимости* перехода, определенной как математическое ожидание кумулятивной стоимости *действий*, предпринятых в течение нескольких шагов, а не просто текущей стоимости.

Практическая реализация обучения с отложенным подкреплением осложнена по двум причинам:

- не существует учителя, формирующего желаемый отклик на каждом шаге процесса обучения;
- наличие задержки при формировании первичного сигнала подкрепления требует решения временной задачи присваивания коэффициентов доверия, т.е. что обучаемая ИНС должна быть способна присваивать коэффициенты доверия и недоверия действиям, выполненным на всех шагах, приводящих к конечному результату, в то время как первичный сигнал подкрепления формируется только на основе конечного результата.

Несмотря на эти сложности, обучение с отложенным подкреплением является очень привлекательным вследствие развития способности самостоятельного решения возникающих задач на основе лишь собственных результатов взаимодействия со средой.

Обучение с подкреплением тесно связано с динамическим программированием. «Погружая» обучение с подкреплением ИНС в предметную область динамического программирования, можно взять на вооружение все результаты последнего.

11.5 Основные концепции искусственных нейронных сетей

В предыдущем подразделе рассмотрены три концепции ИНС (однослойный персептрон, многослойный персептрон, самоорганизующиеся карты Кохонена). В данном подразделе рассмотрены другие основные концепции ИНС, получившие достаточно широкое распространение в практике нейросетевого анализа и моделирования.

11.5.1 Искусственные нейронные сети встречного распространения

На рисунке 11.34 показана структура нейронной сети встречного распространения.

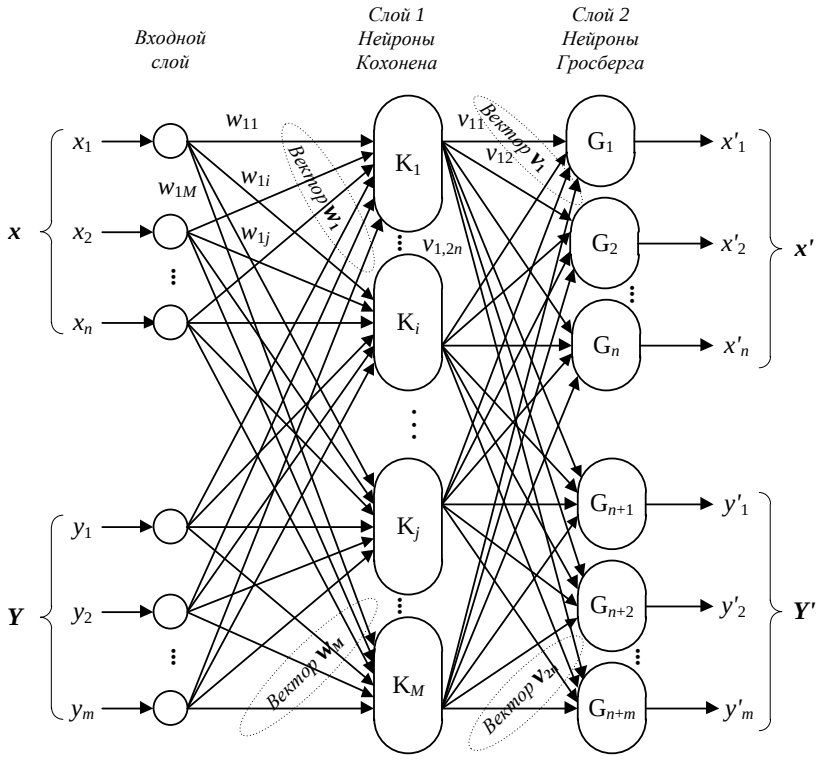


Рисунок 11.34 – Структура ИНС встречного распространения

Особенности ИНС встречного распространения:

- состоит из слоев нейронов Кохонена и Гроссберга;
- каждый элемент входного сигнала подается на все нейроны слоя Кохонена. Каждый нейрон слоя Кохонена соединен со всеми нейронами слоя Гроссберга;
- хорошие способности сети к обобщению, позволяющие получать правильный выход даже при неполном или зашумленном входном векторе;
- в режиме обучения на входы сети подаются как входные, так и соответствующие им выходные сигналы;
- целью обучения является настройка весовых коэффициентов для установления совпадения сигналов на входах (*X* и *Y*) с сигналами на выходах (*X'* и *Y'*) сети.

Нейроны слоя Кохонена:

- реализуют функцию порогового суммирования взвешенных входов;
- функционирует по правилу «победитель получает все» (уровень логической «1»); на выходе выигравшего нейрона формируется сигнал:

$$s_j = \sum_{i=1}^{n} w_{ij} x_i + \sum_{l=1}^{m} w_{ij} y_l,$$

где s_j – выход j-го нейрона Кохонена, $W_j = (w_{1j}, w_{2j}, ..., w_{nj}, w_{n+1,\,j}, ..., w_{mj},)$ – вектор весов j-го нейрона Кохонена;

- нейроны слоя Кохонена кластеризуют входные векторы (X и Y) в группы схожих с помощью такой подстройки весов слоя Кохонена, что близкие входные векторы активируют один и тот же нейрон данного слоя.

Нейроны слоя Гроссберга:

- назначением нейронов слоя Гроссберга является формирование требуемых выходных векторов после того, как нейроны слоя Кохонена разделили входные векторы на классы;
- на их выходах формируются взвешенные суммы сигналов выходов нейронов слоя Кохонена;
- каждый нейрон слоя Гроссберга выдает значение веса, связывающий его с единственным нейроном-победителем Кохонена, чей выход отличен от нуля.

Особенности обучения ИНС встречного распространения:

- на вход сети подаются нормализованные единичные векторы X и Y, а на выходе формируются их нормализованные аппроксимации X' и Y';
- слой Кохонена обучается по алгоритму обучения без учителя. При этом процесс обучения после выбора нейрона-победителя с весовым вектором, наиболее близким к входному вектору, состоит в дальнейшей подстройке (приближении) компонентов весового вектора выбранного нейрона к предъявленному входному вектору в соответствии с выражением

$$w_i(t+1) = w_i(t) + \eta(x^k - w_i(t)),$$

где $w_i(t+1)$, $w_i(t)$ – соответственно новое и предыдущее значения вектора весов выигравшего i-го нейрона-победителя для предъявленного входного вектора x^k ($k = 1, \ldots, N$), η – коэффициент скорости обучения (уменьшается по мере обучения).

- в результате обучения нейрон-победитель будет активизироваться для совокупности ассоциированных с ним «парных» векторов X и Y, соответствующих вектору весов этого нейрона;
- в отличие от слоя Кохонена, слой Гроссберга обучается с учителем. Подстройке подвергаются только те веса нейронов слоя Гроссберга, которые соединены с ненулевым нейроном Кохонена, в соответствии с правилом:

$$v_{ij}(t+1) = v_{ij}(t) + \eta \left(y_j - v_{ij}(t)\right) K_i,$$

где K_i – выход i-го нейрона Кохонена, y_j – j-й компонент требуемого выходного вектора. Первоначально $\eta = 0{,}1$ и уменьшается в процессе обучения до нуля.

- после обучения в нейронной сети встречного распространения реализуется свойство ассоциативной памяти, заключающееся в том, что предъявление на вход только вектора X (или Y) при отсутствии другого приводит к порождению на выходе как вектора X' так и Y'.

Области применения ИНС встречного распространения:

- распознавание образов,
- восстановление образов (ассоциативная память),
- сжатие данных (с потерями).

11.5.2 Искусственные нейронные сети радиальных базисных функций

На рисунке 11.35 представлена структура *ИНС радиальных базисных функций (Radial Basis Function Network, RBFN)*, а на рисунке 11.36 – график радиальной базисной функции.

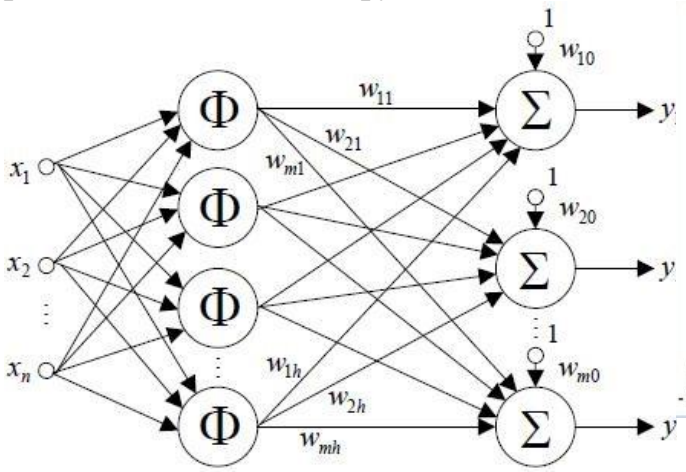


Рисунок 11.35 – Структура RBF-сети

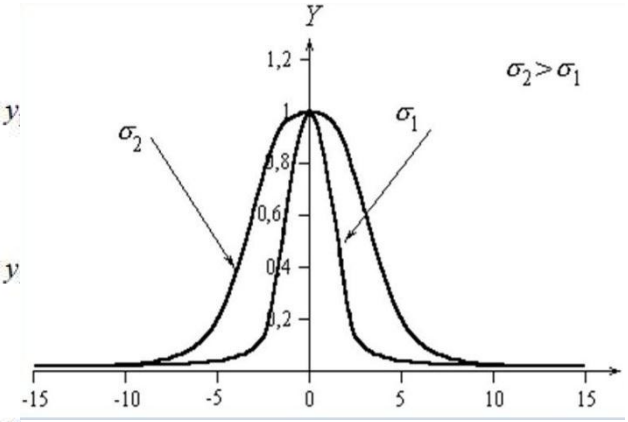


Рисунок 11.36 – Радиальная базисная функция

RBF-*сеть* – это ИНС, содержащая слой скрытых нейронов с радиально симметричной (гауссовой) активационной функцией, каждый из которых предназначен для хранения отдельного эталонного вектора (в виде вектора весов) и дают значимый отклик лишь в небольшой окрестности «центрального пика» в пространстве входных параметров.

Эти ИНС позволяют находить соответствия между входными и выходными векторами, используя локальные аппроксимации. Обычно обучение с учителем вычисляет только линейные комбинации аппроксиматоров. Так как количество этих комбинаций ограничено, процесс обучения проходит исключительно быстро и требует лишь небольшого числа обучающих примеров.

Условия построения RBF-сети:

- наличие эталонов, представленных в виде весовых векторов нейронов скрытого слоя: $C_i = (c_{i1}, \ldots, c_{iN})$ – весовой вектор i-го эталонного нейрона скрытого слоя, $i = 1, \ldots, h$;
- способ измерения расстояния входного вектора от эталона (обычно это евклидово расстояние):

$$\rho_i = \sqrt{\sum_{j=1}^{N} \left(x_j - c_{ij} \right)^2}, \quad i = 1, \ldots, h,$$

где $X = (x_1, \ldots, x_N)$ – входной вектор;

- специальная функция активации нейронов скрытого слоя, задающая выбранный способ измерения расстояния. Обычно используется функция Гаусса, существенно усиливающая малую разницу между входным и эталонным векторами

$$\phi_i = \exp\left(-\left(\frac{\rho_i - R}{\sigma_i}\right)^2\right), \qquad i = 1, \ldots, h.$$

Особенности обучения и функционирования RBF-сетей:

- обучение скрытого слоя сети подразумевает предварительное проведение кластеризации для нахождения эталонных векторов и значений σ_i;
- нейроны скрытого слоя соединены по полносвязной схеме с нейронами выходного слоя, которые осуществляют взвешенное суммирование. Для нахождения значения весов от нейронов скрытого к выходному слою используется линейная регрессия;
- в общем случае активационные функции нейронов скрытого слоя могут отражать законы распределения случайных величин (вероятностные нейронные сети) либо характеризовать различные аналитические зависимости между переменными (регрессионные нейронные сети).

Области применения RBF-сетей:

- распознавание образов;
- классификация.

Достоинства RBF-сетей:

- моделируют произвольную нелинейную функцию с помощью всего одного промежуточного слоя, тем самым избавляя от необходимости решать вопрос о числе слоев;
- параметры линейной комбинации в выходном слое можно полностью оптимизировать с помощью методов линейной оптимизации, которые работают быстро и не испытывают трудностей с локальными минимумами, мешающими при обучении с использованием алгоритма обратного распространения ошибки;
- обучается очень быстро (на порядок быстрее, чем с использованием алгоритма обратного распространения).

Недостатки RBF-сетей:

- обладают плохими экстраполирующими свойствами;
- весьма громоздки при большой размерности входных векторов.

11.5.3 Искусственные нейронные сети с анализом главных компонентов

Искусственные нейронные сети с анализом главных компонентов (*Principal Component Analysis*) сочетают два подхода к обучению – с учителем и без учителя. Сам метод анализа главных компонентов предназначен

для сокращения размерности пространства признаков с минимальной потерей полезной информации.

Проекции выявленных компонентов соответствуют собственным значениям ковариационной матрицы входного вектора.

Самообучающаяся часть PCA-сети выполняет выделение значимых признаков (компонентов), а часть, обучаемая с учителем (в виде многослойного персептрона), проводит классификацию входных данных (образов) по этим признакам (рисунок 11.37).

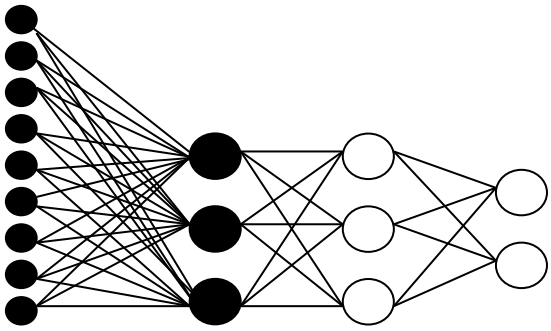


Рисунок 11.37 – Упрощенное представление *PCA*-сети

При построении PCA-сети сначала проводится выделение главных компонентов, а затем проводится выполняется процедура ее обучения. При этом при анализе главных компонентов нет необходимости тратить время на обучение, так как это стандартная статистическая процедура, основные же ресурсы тратятся на обучение многослойного персептрона посте того, как стабилизируются собственные значения ковариационной матрицы исходных данных.

11.5.4 Каскадные искусственные нейронные сети

На рисунке 11.38 представлена каскадно-корреляционная ИНС С. Фальмана и К. Лебьера.

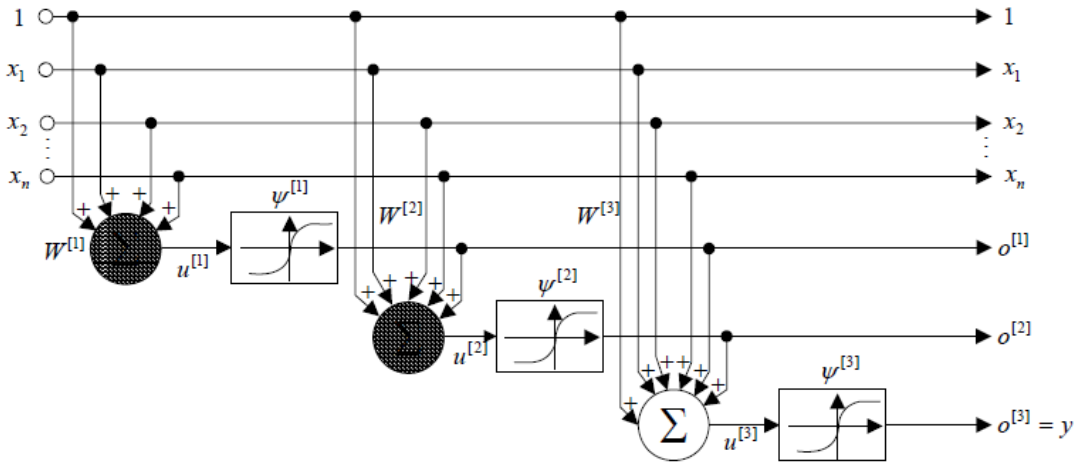


Рисунок 11.38 – Каскадно-корреляционная ИНС

Процедура построения-обучения каскадно-корреляционных сетей включает в себя следующие шаги.

Шаг 1. Формируется стандартная однослойная структура с n входами и m выходами (для m =1 это единственный нейрон), которая обучается с помощью любого из нелинейных алгоритмов обучения;

Шаг 2. После предъявления всей обучающей выборки $x(1)$, $x(2)$,…, $x(N)$ оценивается точность аппроксимации и, если ошибка велика, формируется каскад из n_2 нейронов-кандидатов (5–10 нейронов), параллельно подключенных ко входам сети 1, x_1 , x_2 ,…, x_n и выходу первого каскада $o^{[1]}$.

Нейроны-кандидаты, как правило, отличаются друг от друга начальными значениями синаптических весов $W^{[2]}(0)$, видом функций активации и алгоритмами обучения.

Шаг 3. Далее проводится обучение нейронов 2-го каскада при «замороженных» синаптических весах $W^{[1]}(N)$ 1-го каскада. На рисунке «замороженные» веса показаны в виде заштрихованных сумматоров.

Шаг 4. Среди n_2 нейронов-кандидатов выбирается один нейрон-победитель, у которого параметр корреляции является максимальным:

$$r_q^{[2]} = \left|\sum_{k=1}^{N}\left(o_q^{[2]}(k) - o_q^{-[2]}\right)\left(e_q^{[2]}(k) - e_q^{-[2]}\right)\right|,\ q=1,2,\ \ldots,\ n_2,$$

где $o_q^{-[2]}$ и $e_q^{-[2]}$ – средние значения выходного сигнала и ошибки) является максимальным.

Именно этот нейрон с «замороженными» весами $W^{[2]}(N)$ образует 2-й каскад, в то время как «проигравшие» нейроны изымаются из сети.

Шаг 5. Далее оценивается точность аппроксимации, обеспечиваемая вторым каскадом, и в случае необходимости формируется команда из n_3 кандидатов 3-го каскада, среди которых выбирается победитель с максимальным значением

$$r_q^{[3]} = \left|\sum_{k=1}^{N}\left(o_q^{[3]}(k) - o_q^{-[3]}\right)\left(e_q^{[3]}(k) - e_q^{-[3]}\right)\right|,\ q=1,2,\ \ldots,\ n_3$$

В случае достижения требуемой точности процесс наращивания каскадов завершается и выходной сигнал последнего каскада (на рисунке 11.38 – $o^{[3]}$) принимается в качестве выходного сигнала сети в целом.

Достоинства каскадно-корреляционных ИНС:

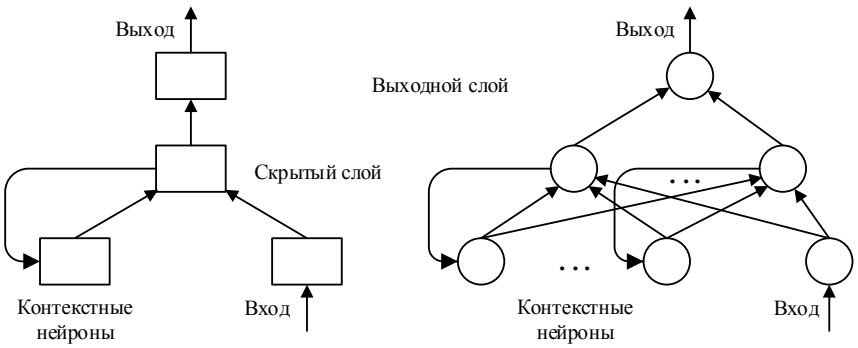
- эти сети не требуют предварительного задания ни архитектуры, ни количества нейронов в каскадах;
- нейроны в сеть добавляются в процессе обучения по мере необходимости, образуя не скрытые слои, а каскады, каждый из которых в качестве входных сигналов использует входы сети и выходы предыдущего каскада;
- для обучения можно использовать не только алгоритм обратного распространения ошибок, что позволяет существенно сократить время настройки;

- за счет «замораживания» синаптических весов сформированных ранее каскадов сокращаются вычислительные затраты на обучение;
- процедуру обучения сети можно существенно упростить, добавляя к предыдущему каскаду не группу нейронов-кандидатов, а единственный обобщенный формальный нейрон, настраивая не только его синаптические веса, но и характеристики активационной функции;

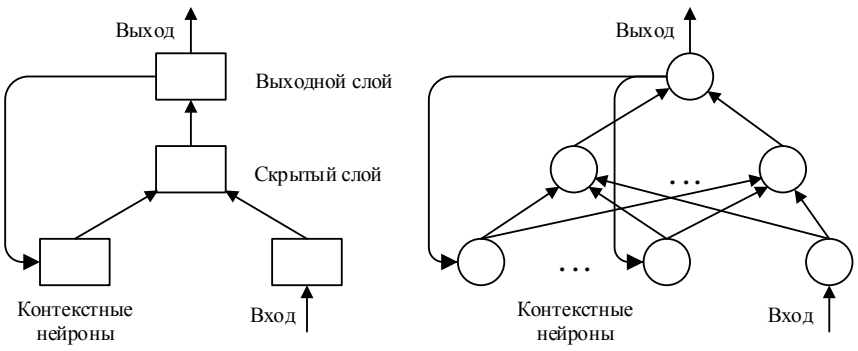
Недостаток каскадно-корреляционных ИНС – возможность обучения сети только в пакетном режиме при наличии заранее заданной обучающей выборки.

11.5.5 Искусственные нейронные сети Жордана и Элмана

Искусственные нейронные сети Жордана и Элмана являются разновидностями рекуррентных ИНС, получаются из многослойных персептронов введением контекстных нейронов, а также обратных связей, идущих от выходов нейронов последующих на контекстные нейроны (рисунок 11.39).



а)



б)

Рисунок 11.39 – Сети Элмана (*а*) и Жордана (*б*)

Основной особенностью таких ИНС является запоминание и извлечение последовательностей данных, а также учет предыстории анализируемых процессов. Использование контекстных нейронов (позволяет в ряде случаев повысить адаптивность ИНС.

11.5.6 Искусственные нейронные сети Хопфилда

На рисунке 11.40 показан пример структуры ИНС Хопфилда, а на рисунке 11.41 приведена иллюстрация состояний этой сети.

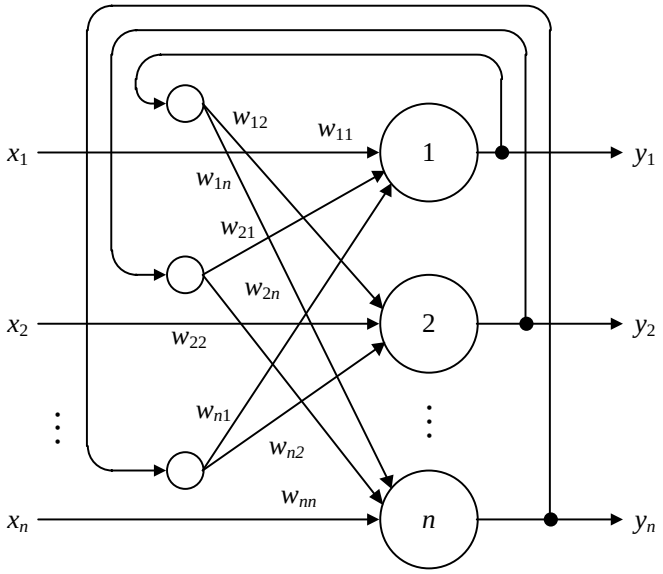


Рисунок 11.40 – Структура сети Хопфилда

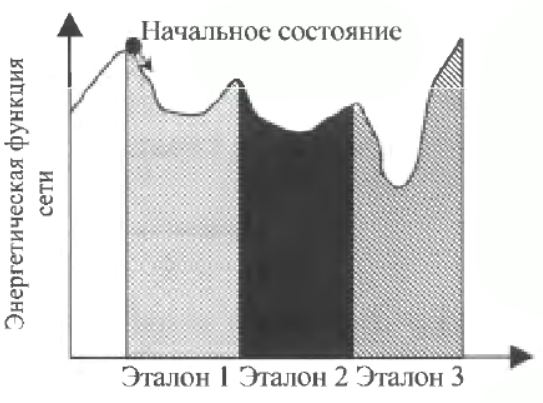


Рисунок 11.41 – Иллюстрация состояний сети Хопфилда

Особенности нейронных сетей Хопфилда:

- сеть Хопфилда реализует свойство автоассоциативной памяти – восстановление по искаженному (зашумленному) образу ближайшего к нему эталонного;
- сеть состоит из одного слоя нейронов, число которых определяет число входов и выходов сети, т.е. равно *n*;
- выход каждого нейрона соединен с входами всех остальных нейронов;
- входные сигналы задаются в виде некоторого набора бинарных сигналов $X^k = (x_1^k, x_1^k, ..., x_n^k), k = 1, ..., K$, каждый из которых x_i^k принимает значение либо «–1», либо «+1»;
- на выходе каждого нейрона формируется сигнал:

$$y_i = \begin{cases} 1, \ \sum_{l=1}^{n} w_{lj} x_j \geq \theta, \\ -1, \ \sum_{l=1}^{n} w_{lj} x_j < \theta, \end{cases}$$

где θ – пороговое значение;

- расчет и изменение весовых коэффициентов нейронов проводится на стадии инициализации сети лишь перед началом функционирования

сети, и все обучение сети сводится именно к этому расчету без обучающих итераций:

$$w_{ij} = \begin{cases} \sum_{k=1}^{m} x_i^k x_j^k, i \neq j, \\ 0, i \neq j, \end{cases}$$

где 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.

- при предъявлении входного вектора, сеть «сходится» к одному из запомненных в сети эталонов, представляющих множество равновесных точек, которые являются локальными минимумами функции энергии, содержащей в себе всю структуру взаимосвязей в сети;
- проблема устойчивости сети Хопфилда решена, так как выполняется достаточное условие устойчивости сетей с обратными связями, т.е. матрица весов симметрична ($w_{ij} = w_{ji}$) и имеет нули на главной диагонали ($w_{ii} = 0$);
- нейроны меняют свои состояния асинхронно, связь между ними осуществляется мгновенно;
- все возможные состояния (запомненные образы) сети образуют некое подобие холмистой поверхности, а текущее состояние сети аналогично поведению тяжелого шарика, пущенного на эту поверхность: он движется вниз по склону в ближайший локальный минимум.
- одна и та же сеть с одними и теми же весами связей может хранить и воспроизводить несколько различных эталонов. Каждый эталон является аттрактором, вокруг которого существует область притяжения. Любая система с несколькими аттракторами, к которым она тяготеет, может рассматриваться как автоассоциативная память;
- если на вход задается начальное состояние, отличное от эталонного, то это равносильно заданию частичной информации об эталоне. Если начальное состояние близко к эталону и попадает в область его притяжения, то сеть начинает двигаться к этому эталону – «вспоминает» его;
- число запоминаемых образов не должно превышать $0{,}15n$;
- динамическое изменение состояний сети может быть выполнено синхронно (все элементы модифицируются одновременно на каждом временном шаге) и асинхронно (в каждый момент времени выбирается и подвергается обработке один элемент).

Каждая точка поверхности соответствует некоторому сочетанию активностей нейронов в сети, а высота подъема поверхности в данной точке характеризует «энергию» этого сочетания (называемую функцией Ляпунова), которую можно представить следующим образом:

$$E = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} y_i y_i - \sum_{j=1}^{n} x_j y_j + \sum_{j=1}^{n} \theta_j y_j, \; i \neq j,$$

где E – искусственная энергия сети; w_{ij} – вес от выхода i-го ко входу j-го нейрона; x_j, y_j – вход и выход j-го нейрона; θ_j – порог j-го нейрона.

Изменение энергии ΔE, вызванное изменением состояния j-го нейрона:

$$\Delta E = \left(\sum_{i \neq j} w_{ij} y_i + I_j - \theta_j \right) \Delta y_j ,$$

где I_j – внешний выход j-го нейрона.

Если связь между какими-то нейронами имеет большой положительный вес, то сочетания, в которых эти нейроны активны, характеризуются низким уровнем энергии – именно к таким сочетаниям стремится вся сеть. И наоборот, нейроны с отрицательной связью при активизации добавляют к энергии сети большую величину, поэтому сеть стремится избегать подобных состояний.

Чтобы обучить сеть, необходимо сформировать соответствующий профиль энергетической поверхности, т.е. выбрать веса таким образом, чтобы при фиксировании каждого входного вектора сеть приходила к энергетическому минимуму, соответствующему нужному выходному вектору.

Описание функционирования нейронной сети Хопфилда:

Во-первых, после подачи входного образа на входы сети она генерирует выходные сигналы.

Во-вторых, затем входной образ снимается со входов сети.

В-третьих, после этого сеть переходит из состояния в состояние до тех пор, пока не стабилизируется. Если входной вектор частично искажен или неполон, сеть стабилизируется в состоянии, ближайшем к одному из эталонов.

11.5.7 Искусственные нейронные сети Хэмминга

На рисунке 11.42 показана структура ИНС Хэмминга.

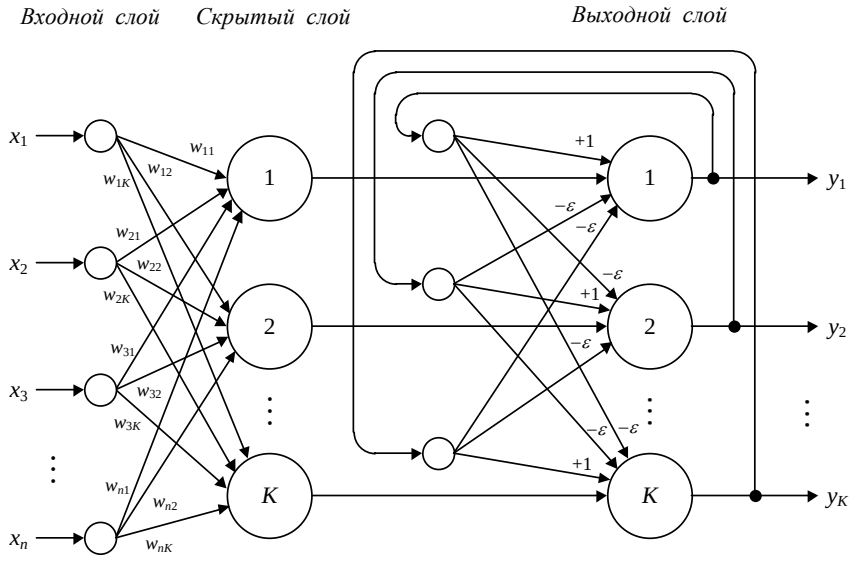


Рисунок 11.42 – Структура ИНС Хэмминга

Особенности нейронных сетей Хэмминга:

- позволяют решать задачу автоассоциативной памяти – воссоздания образов по неполной и искаженной информации;
- в отличие от сети Хопфилда, нейронная сеть Хэмминга выдает не сам эталонный образ, а его номер;
- состоит из двух слоев, каждый из которых имеет по K нейронов, соответствующих числу запоминаемых эталонов;
- значения элементов входных векторов $X^k = (x_1^k, ..., x_i^k, ..., x_n^k), k = 1, ..., K$ принимают бинарные значения «–1» или «1»;
- нейроны 2-го слоя связаны между собой ингибиторными (отрицательными обратными) синаптическими связями; единственный синапс с положительной обратной связью для каждого нейрона соединен с его же аксоном;
- постановка задачи заключается в следующем: имеется набор эталонных образов в виде бинарных векторов. Требуется сопоставить входной образ со всеми эталонными образами и отнести его к наиболее близкому (по числу совпавших элементов) эталону либо сделать заключение о несоответствии ни одному из эталонных образов. Сеть должна выбрать эталонный образ с минимальным расстоянием Хэмминга (минимальным числом отличающихся битов в двух бинарных векторах) до входного сигнала, в результате чего будет активизирован только один выход сети, соответствующий этому эталону.

Алгоритм функционирования нейронной сети Хэмминга заключается в следующем. Предварительно на стадии инициализации весовым коэффициентам 1-го слоя и порогу активационной функции присваиваются следующие значения:

$$w_{ik} = \frac{x_i^k}{2}, \theta_k = \frac{n}{2}, \ i = 1, 2, ..., n, \ k = 1, 2, ..., K,$$

где x_i^k – i-й элемент k-го эталона.

Весовые коэффициенты тормозящих синапсов во 2-м слое берут равными некоторому значению $0 < \varepsilon < 1/K$. Синапс нейрона, связанный с его же аксоном, имеет вес «+1».

Шаг 1. На нейроны входного слоя подается вектор $X = \{x_i\}$, $i = 1 \ldots n$. После чего он со входов снимается.

На выходах нейронов 1-го слоя формируются следующие значения (верхний индекс указывает номер слоя):

$$y_k^{(1)} = s_k^{(1)} = \sum_{i=1}^{n} w_{ik} x_i + \theta_j, k = 1, \ldots, K.$$

В соответствии с этим устанавливаются значения на выходах нейронов выходного слоя:

$$y_k^{(2)} = y_k^{(1)}, k = 1, \ldots, K.$$

Шаг 2. В результате новой (t+1)-й итерации определяются новые состояния нейронов выходного слоя:

$$s_k^{(2)}(t+1) = y_k^{(2)}(t) - \varepsilon \sum_{\substack{j=1 \\ j \neq k}}^{n} y_k^{(2)}(t), \;\; k = 1, ..., K,$$

$$y_k^{(2)}(t+1) = f\left(s_k^{(2)}(t+1)\right), \quad k = 1, ..., K.$$

Активационная функция f представляет собой пороговую функцию, причем значение порога должно быть достаточно большим, чтобы возможные значения $s_k^{(2)}$ не приводили к насыщению.

Шаг 3. Проверка изменения состояний нейронов выходного слоя за последнюю итерацию. И переход к шагу 2 в случае, если наблюдались изменения. Иначе – окончание процедуры.

В идеальном случае после стабилизации сети должен получиться выходной вектор со всеми нулевыми элементами кроме одного положительного, индекс которого и укажет на распознаваемый входной образ.

11.5.8 Искусственные нейронные сети адаптивной резонансной теории

Искусственные нейронные сети адаптивной резонансной теории (Adaptive Resonance Theory, ART-сети) предназначены для преодоления существенного недостатка ИНС, обучаемых с учителем, заключающегося в том, что запоминание новых входных сигналов приводит к перенастройке всей сети в целом, в результате чего качество их работы по ранее запомненным классам образов в общем случае ухудшается.

Кроме того, традиционные ИНС не способны самостоятельно определить появление нового класса и для его выделения требуется достоверная информация от учителя.

Сети адаптивного резонанса реализуют *свойство пластичности*. Для каждого нового входного вектора первоначально предпринимается попытка отнести его к одному из имеющихся классов. В случае его достаточной близости к одному из классов в параметрическом пространстве в заданной метрике целевой функции, веса, соотнесенные с выбранным классом, модифицируются. В противном случае, если новый вектор значительно отличается от всех ранее запомненных, принимается решение о формировании нового класса, основные параметры которого задаются новым предъявленным образом.

Известны различные модификации ART-сетей. Рассмотрим этапы функционирования ИНС АРТ-1 (рисунок 11.43).

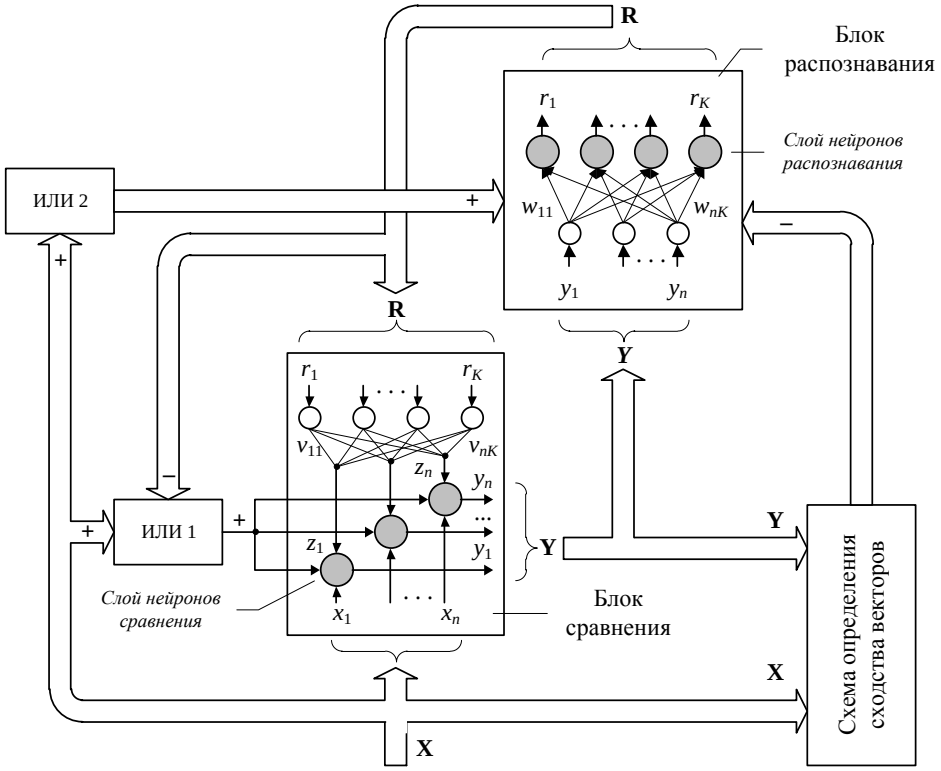


Рисунок 11.43 – Структура ИНС АРТ-1

Шаг 1. Инициализация

Устанавливаются значения весов нейронов слоя распознавания:

$$w_{ik} < \frac{H}{H-1+n},\ i=1,...,n,\ k=1,...,K,,\quad i=1\ldots n,\ k=1\ldots K,$$

где n – число компонентов входного вектора **X** (число нейронов слоя сравнения); K – число нейронов слоя распознавания; H – константа в диапазоне от 1 до 2.

Устанавливаются значения весов нейронов слоя сравнения, равными «1».

Устанавливаются параметр сходства ρ в диапазоне [0, 1].

Шаг 2. Обучение

На входы сети последовательно подаются входные бинарные векторы $\mathbf{X}_k$.

И весовые векторы $\mathbf{W}_k$ изменяются так, чтобы сходные векторы $\mathbf{X}_k$ активизировали соответствующие нейроны слоя распознавания. Веса вычисляются по правилу:

$$w_{ik} = \frac{Hy_i}{H-1+\sum_{j=1}^{n} y_j},$$

где y_i – i-й компонент вектора **Y**; k – номер активного нейрона в слое распознавания.

Компоненты вектора весов $\mathbf{V}_k$, связанные с новым запоминаемым образом, изменяются таким образом, что $v_{ik} = y_i$ для всех i.

Шаг 3. Распознавание

Входной ненулевой вектор X проходит без изменения через блок сравнения на вход нейронов слоя распознавания.

Далее для каждого *k*-го нейрона слоя распознавания вычисляется свертка его весового вектора $\mathbf{W}_k$ с вектором **Y**.

Выход нейрона слоя распознавания с максимальным значением свертки, т.е. наиболее «близкого» к входному вектору, переходит в активное (единичное) состояние, т.е. «резонирует», тормозя остальные нейроны этого слоя, которые установятся в «0».

Шаг 4. Сравнение

Единица с выхода *k*-го возбужденного нейрона слоя распознавания подается на каждый *i*-й нейрон в слое сравнения со своим весом, устанавливая на входах z_i нейронов слоя сравнения уровень либо «0», либо «1».

Выход схемы ИЛИ 1 инициирует сравнение. И теперь в слое сравнения могут возбудиться лишь те нейроны, на входах которых соответствующие компоненты x_i и z_i одновременно равны «1»

Схема определения сходства векторов определяет логическое произведение входного вектора **X** и вектора **Z**, который равен весовому вектору $\mathbf{W}_k$ выигравшего нейрона.

При существенном отличии векторов **X** и **Z** возбужденный нейрон в слое распознавания должен быть заторможен.

Шаг 5. Поиск

В случае совпадения или удовлетворения условий близости векторов **X** и **Y** процесс классификации завершается.

В противном случае осуществляется поиск среди других запомненных образов для определения наиболее близкого к входному.

Если определено, что ни один из запомненных векторов не соответствует входному, то вводится новый (*K*+1)-й нейрон в распознающем слое и его весовые векторы $\mathbf{W}_k$ и $\mathbf{V}_k$ устанавливаются в соответствии с новым входным вектором.

Особенности ИНС АРТ-1:

- быстрый и устойчивый поиск образов;
- конечность процесса обучения, обусловленная стабильным набором весов; повторяющиеся последовательности обучающих векторов не приводят к циклическому изменению весов;
- на подобии с биологическими прототипами решается» проблема стабильности–пластичности»;
- недостатк АРТ-сетей – недостаточная надежность сохранения информации. Так, в случае «потери» одного образа разрушается вся память.

Особенности ИНС АРТ-2:

- используются для распознавания движущихся изображений;
- вектора и сигналы представляются не только в бинарном, но и в аналоговом виде;

- асинхронная работа блоков сети, что является важным для аппаратной реализации;
- использование многослойной сети для реализации блока распознавания образов.

Особенности ИНС АРТ-3:

- сеть представляет собой многослойную архитектуру;
- при переходе от слоя к слою происходит контрастирование входных образов и запоминание их в виде все более общих категорий;
- основной задачей каждого отдельного слоя является сжатие входящей информации; образ входит в адаптирующийся резонанс между некоторой парой слоев, в дальнейшем этот резонанс распространяется на следующие слои иерархии;
- используется механизм зависимости активности синапсов обратных связей от времени, аналогичный рефрактерному торможению биологического нейрона после передачи возбуждения;
- гибридизация за счет использование в многослойной иерархии слоев, которые не являются слоями АРТ, а относятся к другим типам ИНС.

11.5.9 Двунаправленная ассоциативная память

Двунаправленная ассоциативная память (Bi-Directional Associative Memories), предложенная Б. Коско – это ИНС с обратными связями, базирующаяся на адаптивной резонансной теории С. Гросберга и автоассоциативной памяти Хопфилда. На рисунке 11.44 показана структура BDAM-сети.

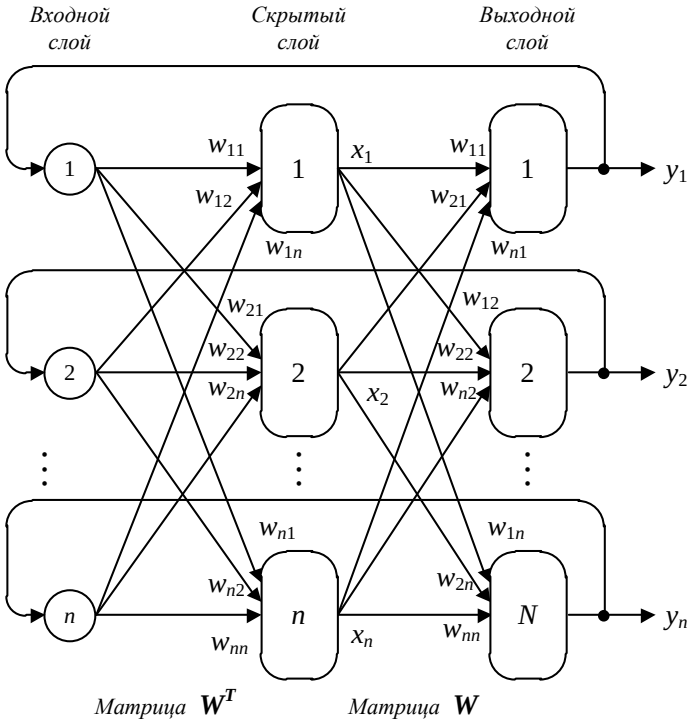


Рисунок 11.44 – Структура BDAM-сети

Особенности BDAM-сети:

- как и сети Хопфилда и Хэмминга, BDAM-сеть способна к обобщению, вырабатывая правильные реакции, несмотря на искаженные входы. Однако эта сеть решает задачи гетероассоциативной памяти, т.е. входной образ может быть ассоциирован с не коррелирующим с ним образом;
- проводя аналогию с биологическими системами, отметим, что значения весов **W** и $\mathbf{W}^{\mathrm{T}}$ образуют долговременную память, а состояние нейронов – кратковременную память;
- нейроны скрытого и выходного слоев выполняют функцию взвешенного суммирования входов с сигмоидальной (логистической) активационной функцией активации *F*;
- устойчивость двунаправленной ассоциативной памяти гарантируется транспонированием матрицы весовых коэффициентов;
- двунаправленная ассоциативная память сводится к сети Хопфилда, если матрица весов **W** является квадратной и симметричной, т. е. $\mathbf{W} = \mathbf{W}^{\mathrm{T}}$;
- для безошибочной работы число запоминаемых векторов *N* не должно превышать $\dfrac{n}{2\log_2 n}$.

Обучение и функционирование BDAM-сети

До начала функционирования нейронная сеть обучается с использованием набора пар векторов **X** и **Y** путем вычисления значений весовых коэффициентов **W** и $\mathbf{W}^{\mathrm{T}}$, реализующих отношения ассоциаций. При этом весовая матрица **W** вычисляется как сумма произведений всех пар векторов *N* из обучающей выборки:

$$\mathbf{W} = \sum_{i=1}^{N} \mathbf{X}_i^{\mathrm{T}} \mathbf{Y}_i, \quad i = 1 \ldots N.$$

При восстановлении запомненных ассоциаций вектор **X** или его часть кратковременно устанавливается на выходах нейронов скрытого слоя. Вектор **X** обрабатывается матрицей весов **W** нейронов выходного слоя. Затем вектор **X** удаляется, и сеть вырабатывает ассоциированный вектор **Y** на выходе нейронов выходного слоя, поступающий на входы нейронов входного слоя и обрабатывающийся транспонированной матрицей $\mathbf{W}^{\mathrm{T}}$ весов нейронов скрытого слоя.

$$\mathbf{Y} = F(\mathbf{XW}),$$
$$\mathbf{X} = F(\mathbf{YW}^{\mathbf{T}}).$$

В каждом цикле происходит уточнение выходного вектора. Процесс повторяется до достижения устойчивого состояния сети, при котором вектор **X** и **Y** не изменяются.

BDAM-сеть функционирует в направлении минимизации энергии сети (функции Ляпунова) в соответствии со значениями весов.

11.5.10 Когнитрон

Когнитрон (К. Фукушима, 1975 г.) – ИНС, представляющая собой гипотетическую модель биологической системы восприятия и распознавания, инвариантную к поворотам, перемещениям, изменениям масштабов образов. Структурно когнитрон организован подобно зрительной коре мозга человека, состоящей нескольких слоев нейронов. Слои когнитрона организованы однотипно, и каждый из них, подобно отдельным слоям зрительной коры, реализует различные уровни обобщения. Например, если входной слой нейронов распознает лишь простые образы (линии) и их ориентацию, то последующие слои способны к все более сложному обобщению, качество которого не зависит от положений распознаваемых образов.

На рисунке 11.45 в упрощенном виде показана структура когнитрона, на рисунке 11.46 – пример решения задачи распознавания, а на рисунке 11.47 – взаимосвязь слоев когнитрона.

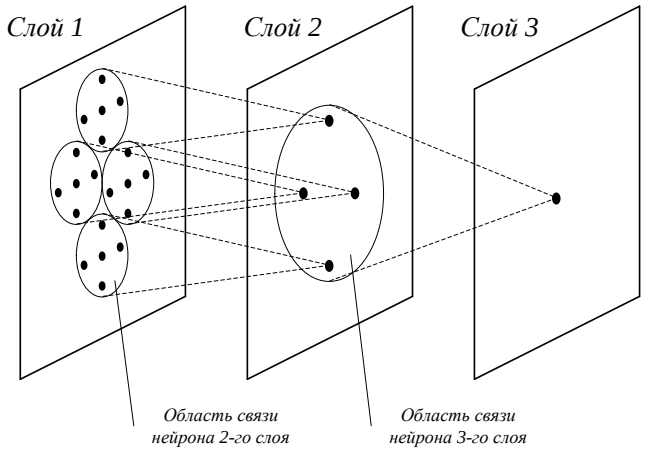


Рисунок 11.45 – Структура когнитрона

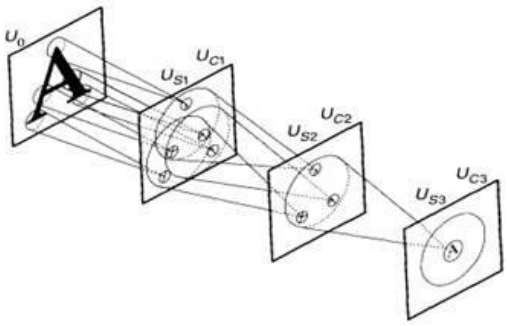


Рисунок 11.46 – Пример решения задачи распознавания когнитроном

Особенности когнитрона:

- когнитрон – модель системы восприятия и распознавания, инвариантной к поворотам, перемещениям, изменениям масштабов образов;
- слои организованы однотипно, каждый из них реализует различные уровни обобщения;
- нейрон из последующего слоя связан с ограниченным набором нейронов предыдущего слоя;
- каждый нейрон выходного слоя реагирует на полное входное поле при ограниченном количестве слоев;
- состоит из иерархически связанных слоев нейронов двух типов – тормозящих и возбуждающих;

- каждый слой когнитрона содержит два типа нейронов: возбуждающие и тормозящие. Возбуждающие нейроны одного слоя стремятся вызвать активизацию соединенного с ними нейрона следующего слоя. Тормозящие нейроны нейтрализуют это возбуждение. Возбуждение нейрона определяется значением нелинейной функции активации от взвешенной суммы его возбуждающих и тормозящих входов;
- на каждый нейрон 2-го слоя оказывают латеральное торможение нейроны из области его конкуренции;
- обучение когнитрона представляет собой обучение без учителя.

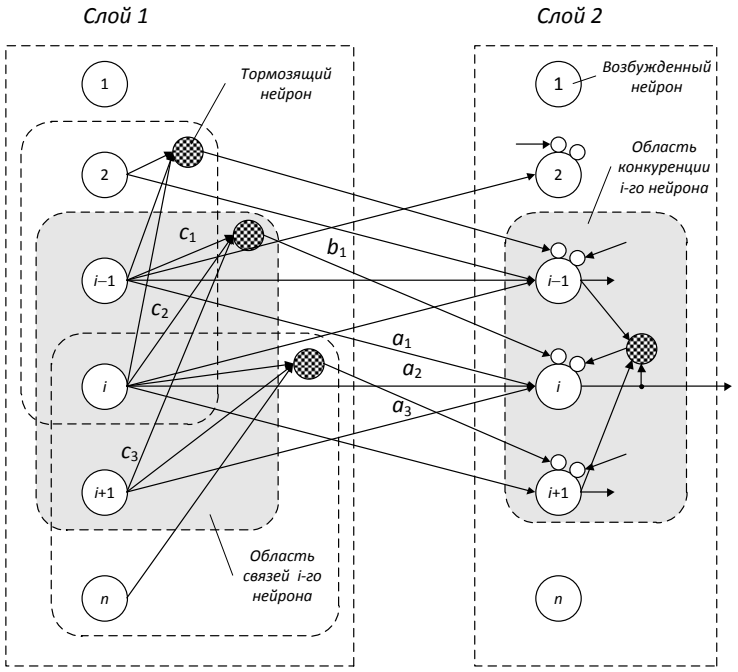


Рисунок 11.47 – Взаимосвязь слоев когнитрона

Значение выхода возбуждающего нейрона

$$y^{в} = F\left(\frac{1+\sum_i a_i u_i}{1+\sum_j b_j v_j} - 1\right) = F\left(\frac{\sum_i a_i u_i - \sum_j b_j v_j}{1+\sum_j b_j v_j}\right),$$

$$F(\varphi) = \begin{cases} \varphi, & \text{при } \varphi \geq 0, \\ 0, & \text{при } \varphi < 0, \end{cases}$$

где a_i – вес i-го возбуждающего входа; u_i – выход i-го возбуждающего нейрона предыдущего слоя; b_j – вес j-го тормозящего входа; v_j – выход j-го тормозящего нейрона предыдущего слоя; веса – только положительные.

Значение на выходе тормозящего нейрона:

$$y^{m} = \sum_i c_i y_i^{в}, \ \sum_i c_i = 1, \ c_i \text{ – вес } i\text{-го возбуждающего входа.}$$

Рассмотрим кратко процессы обучения и функционирование когнитрона.

При инициализации значения на всех выходах нейронов идентичны.

Соседние нейроны слоя (из области конкуренции) конкурируют между собой за счет перекрытия областей связей соседних нейронов. В результате обучения в заданной области слоя возбуждается только один нейрон, который будет оказывать латерально-тормозящее воздействие на соседние нейроны из области его конкуренции. В результате обучения синапсы возбужденного нейрона будут усиливаться, а синапсы соседних нейронов останутся неизменными.

Возбуждающие веса данного нейрона изменяются следующим образом:

$$\delta a_i = \eta\, c_j u_j,$$

где c_j – вес тормозящей связи j-го нейрона с тормозящим нейроном i; u_j – выход j-го нейрона; a_i – вес i-го возбуждающего входа; q –коэффициент скорости обучения.

Изменение значений *тормозящих весов* этого нейрона вычисляется по формуле:

$$\delta b_i = \frac{\eta \sum_j a_j u_j}{2 y_i^m}.$$

Если отсутствуют возбужденные нейроны в области конкуренции, то изменения весов вычисляются следующим образом:

$$\delta a_i = \eta' c_j u_j,\ \delta b_i = \eta' y_i^m,$$

где η' – положительный коэффициент скорости обучения ($\eta' < \eta$).

Благодаря обучению у активизированных нейронов возбуждающие веса увеличиваются сильнее, чем тормозящие. И наоборот, у нейронов, которые проиграли конкуренцию, возбуждающие веса возрастают незначительно, а тормозящие – сильнее.

При обучении веса нейронов 2-го (и последующих) слоев настраиваются так, чтобы активные сигналы на их выходах соответствовали векторам, которые предъявлялись в процессе обучения.

11.5.10 Неокогнитрон

Развитием когнитрона является *неокогнитрон*, представляющий собой многоуровневую иерархическую нейронную сеть, организация и принципы функционирования которой наиболее соответствуют модели зрительной коры головного мозга.

На рисунке 11.48 в упрощенном виде показана схема взаимодействия смежных слоев неокогнитрона, на рисунке 11.49 – взаимосвязь простого нейрона со сложными нейронами из предыдущего слоя.

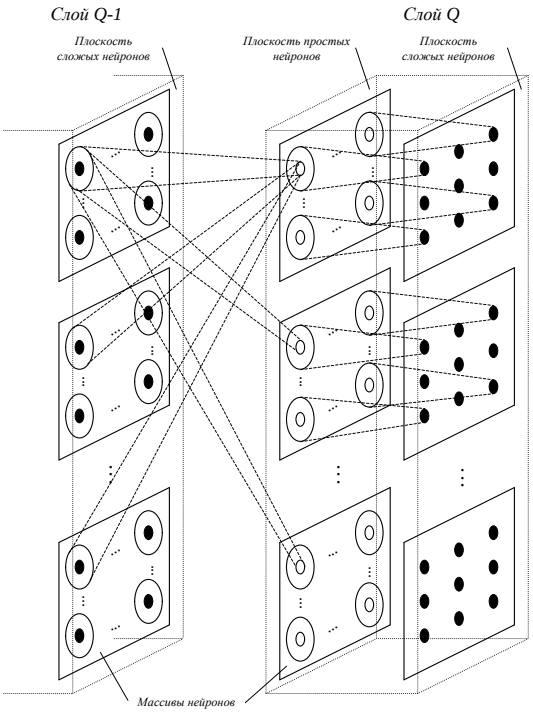


Рисунок 11.48 – Схема взаимодействия смежных слоев неокогнитрона

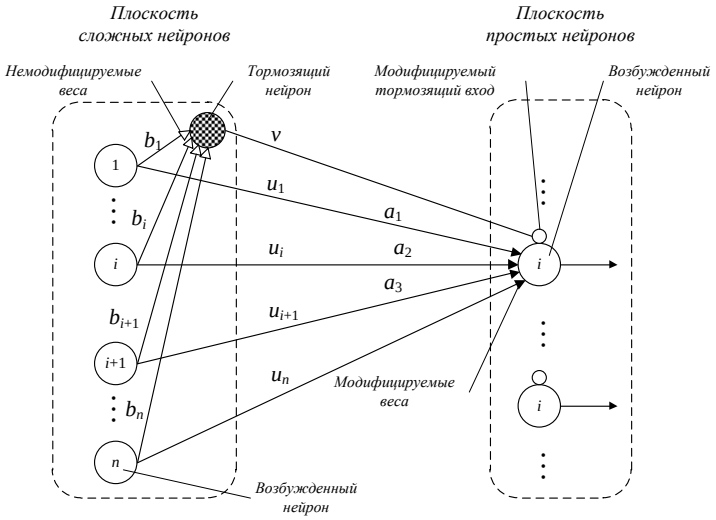


Рисунок 11.49 – Взаимосвязь простого нейрона со сложными нейронами из предыдущего слоя

Особенности когнитрона:

- *неокогнитрон* представляет собой многоуровневую иерархическую нейронную сеть, организация и принципы функционирования которой соответствуют модели зрительной коры головного мозга;
- каждый слой неокогнитрона состоит из двух плоскостей, разделенных на двумерные массивы нейронов. Первая плоскость, содержащая простые нейроны, получает сигналы с выходов сложных нейронов предыдущего слоя, выделяет определенные образы и затем передает их во вторую плоскость данного слоя, содержащую сложные нейроны, где образы обрабатываются таким образом, чтобы обеспечить их меньшую позиционную зависимость;
- внутри слоя массивы простых и сложных нейронов соответствуют друг другу;
- рецептивное поле каждого нейрона от слоя к слою возрастает, количество же нейронов в слое при этом уменьшается. Наконец, в каждом массиве выходного слоя имеется только один сложный нейрон, который реагирует на определенный входной образ;
- при распознавании входной образ подается на вход, а вычисления осуществляются слой за слоем. Так как лишь небольшая часть входного образа подается на вход каждого простого нейрона, некоторые простые нейро-

ны реагируют на наличие характеристик, которым они обучены, и возбуждаются. В следующих слоях выделяются более сложные характеристики как определенные комбинации выходов сложных нейронов, и уменьшается позиционная зависимость.

Особенности плоскости простых нейронов:

- отдельный массив плоскости простых нейронов настраивается на один специфический входной образ. Каждый простой нейрон массива реагирует на ограниченную рецептивную область входного образа, если часть образа, на которую он настроен, встречается во входном образе и обнаружена в его рецептивной области. Другие массивы простых нейронов этой плоскости в этом слое могут быть настроены, например, на повороты образов. Причем для выделения каждого дополнительного образа (или его версии) требуется дополнительный массив;
- рецептивные области плоскости простых нейронов в каждом массиве перекрываются для покрытия всего входного поля этого слоя. Каждый такой простой нейрон получает сигналы от соответствующих рецептивных областей всех массивов плоскости сложных нейронов из предыдущего слоя. Следовательно, простой нейрон реагирует на появление своего образа в любой сложной плоскости предыдущего слоя, если этот образ окажется внутри его рецептивной области;
- простые нейроны в отличие от сложных имеют настраиваемые веса связей, настраиваемые таким образом, чтобы выработать максимальную реакцию на определенные образы.

Особенности плоскости сложных нейронов:

- сложные нейроны решают задачу уменьшения позиционной зависимости реакции неокогнитрона на образы. Для этого на входы каждого сложного нейрона подаются выходные сигналы с набора простых нейронов из соответствующего множества первой плоскости того же слоя;
- активизация любого простого нейрона из рецептивной области сложного нейрона является достаточным условием для возбуждения данного сложного нейрона. Таким образом, сложный нейрон реагирует на тот же образ, что и простые нейроны в соответствующем ему массиве, но он менее чувствителен к позиции образа, чем любой из них;
- каждый слой сложных нейронов реагирует на все большую область входного образа, по сравнению с предшествующими слоями, что приводит к требуемому уменьшению позиционной зависимости реакции неокогнитрона на образы в целом;
- помимо активизирующих, в плоскости сложных нейронов присутствуют тормозящие нейроны, которые вырабатывают выходные сигналы, пропорциональные квадратному корню из взвешенной суммы квадратов их входных сигналов: $v = \sqrt{\sum_i (b_i u_i)^2}$.

Обучение неокогнитрона:

- обучения неокогнитрона – обучение без учителя. На вход неокогнитрона подается распознаваемый образ, и веса синапсов настраиваются слой за слоем. Значение веса от каждого сложного нейрона к заданному простому увеличивается, когда удовлетворяются следующие два условия:
  - активизируется сложный нейрон;
  - реакция одного из простых нейронов больше, чем у его соседей из любой из области конкуренции.
- в результате простой нейрон обучается реагировать более сильно на образы, появляющиеся наиболее часто в его рецептивной области. Если распознаваемый образ отсутствует на входе, тормозящий нейрон предохраняет от случайной активизации соответствующий простой нейрон;
- процедура обучения и латерального торможения когнитрона и неокогнитрона аналогичны. При этом выходы простых и сложных нейронов являются непрерывными, неотрицательными и изменяются по линейному закону;
- при срабатывании на входной образ простого нейрона его веса должны быть увеличены. Также увеличиваются веса всех простых нейронов из данного массива для этого самого образа. Таким образом, все нейроны в массиве обучаются распознавать одни и те же свойства образа, и после обучения будут делать это независимо от позиции образа в поле сложных нейронов из предшествующего слоя. Это определяет способность неокогнитрона к самовосстановлению. Так, если активизируемый нейрон выйдет из строя, то среди других выбирается другой, реагирующий наиболее сильно, который и будет обучен распознаванию входного образа, заменяя отказавший нейрон;
- имеется возможность обучения неокогнитрона с учителем, когда заранее определяется его реакция на каждый входной образ.

11.5.10 Сверточные нейронные сети

Сверточные нейронные сети (Convolutional Neural Network, CNN) – ИНС, (Я. Лекун, 1998 г.), изначально предназначенные для эффективного распознавания изображений (рисунок 11.50).

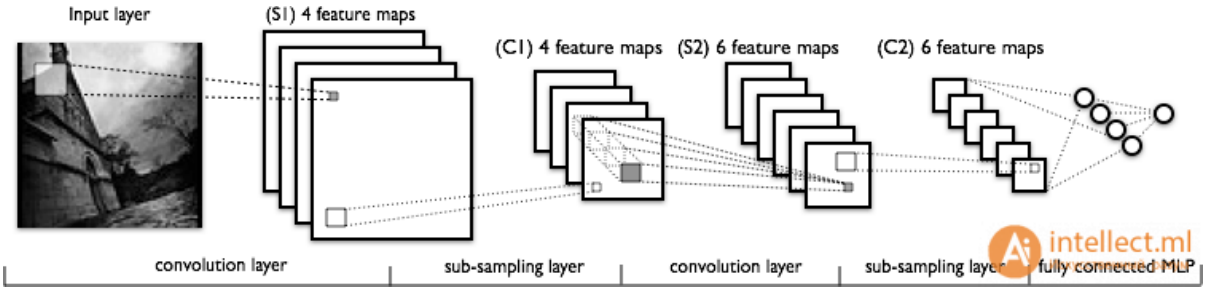


Рисунок 11.50 – Схема сверточной нейронной сети (с сайта http://deeplearning.net)

Особенности структуры сверточных нейронных сетей – первые два типа слоев: *сверточный* – convolutional (рисунок 11.51) и *субдискретизирующий* – subsampling, чередуясь между собой, формируют входной вектор признаков для последнего *полносвязного слоя* – многослойного персептрона.

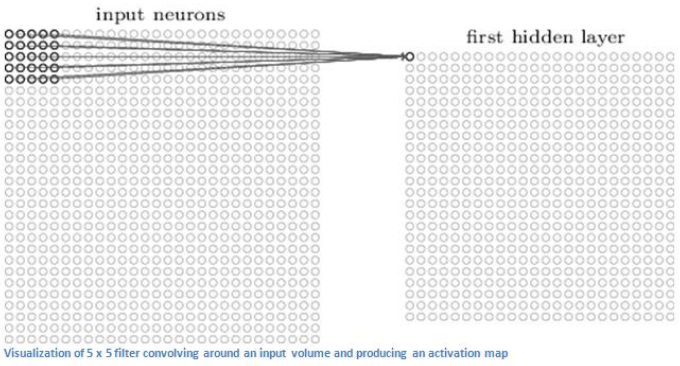


Рисунок 11.51 – Пример свертки

Свертка – операция над парой матриц **A** (размером $n_x \times n_y$) и **B** (размером $m_x \times m_y$), в результате выполнения которой формируется матрица $\mathbf{C} = \mathbf{A} * \mathbf{B}$ размером $(n_x - m_x + 1) \times (n_y - m_y + 1)$. Каждый элемент результата вычисляется как скалярное произведение матрицы **B** и некоторой подматрицы **A** такого же размера (подматрица определяется положением элемента в результате). То есть,

$$C_{i,j} = \sum_{u=0}^{m_x-1} \sum_{v=0}^{m_y-1} A_{i+u,j+v} B_{u,v}.$$

Матрица **B** «перемещается» по матрице **A**, и в каждом положении вычисляется скалярное произведение матрицы **B** и той части матрицы **A**, на которую она в данный момент «наложена». Получившееся значение фиксируется в соответствующем элементе результата. На рисунке 11.52 приведен пример свертки двух матриц размером 5×5 и 3×3.

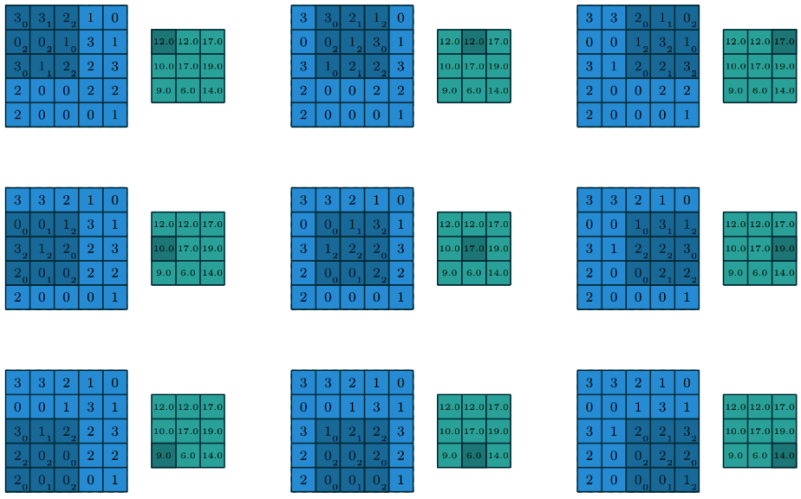


Рисунок 11.52 – Пример свертки двух матриц размером 5×5 и 3×3

Субдискретизирующий (пулинговый) слой (рисунок 11.53) предназначен для снижения размерности изображения. Исходное изображение делится на блоки размером $w \times h$ и для каждого блока вычисляется некоторая функция. Чаще всего используется функция максимума (max pooling) или взвешенного среднего (weighted average pooling). Обучаемых параметров у этого слоя нет.

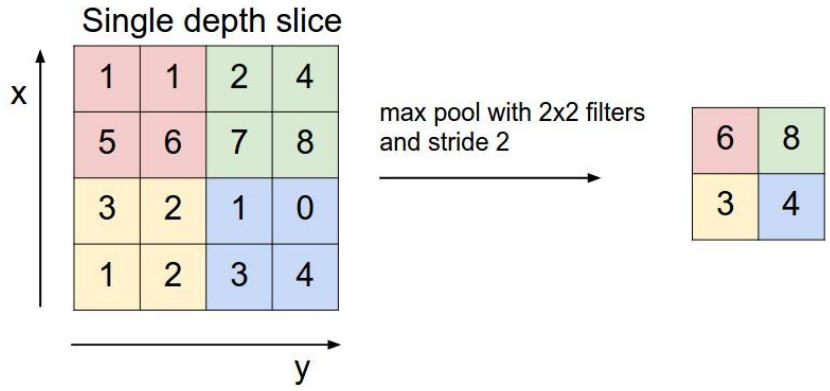


Рисунок 11.53 – Пример субдискретизации (пуллинга)

Основными задачими субдискретизирующего слоя являются: уменьшение изображения; увеличение инвариантности выходных сигналов к незначительным изменениям входных сигналов; распараллеливание вычислений.

Сверточные нейронные сети позволяют более точно, по сравнению с многослойным персептроном, решать задачи распознавания и классификации изображений, так как они учитывают 2d-топологию изображений. При этом сверточные нейронные сети устойчивы к небольшим смещениям, изменениям масштаба и поворотам объектов на входных изображениях.

11.5.10 Рекуррентные нейронные сети

Рекуррентные нейронные сети (Recurrent Neural Network, RNN) – ИНС, в которых связи между элементами образуют направленную последовательность (рисунок 11.54).

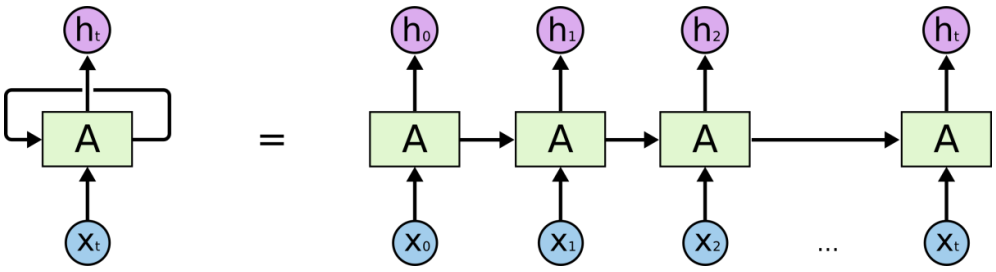


Рисунок 11.54 – Представление рекуррентной нейронной сети

Такие ИНС эффективно используются для решения задач:

- обработки текста на естественном языке и автоматического перевода;
- обработки аудиоинформации (автоматическое распознавание речи);
- обработки видеоинформации;
- распознавание эмоций;
- генерации описания изображений.

На рисунке 11.55 показаны различные разновидности структур рекуррентных нейронных сетей.

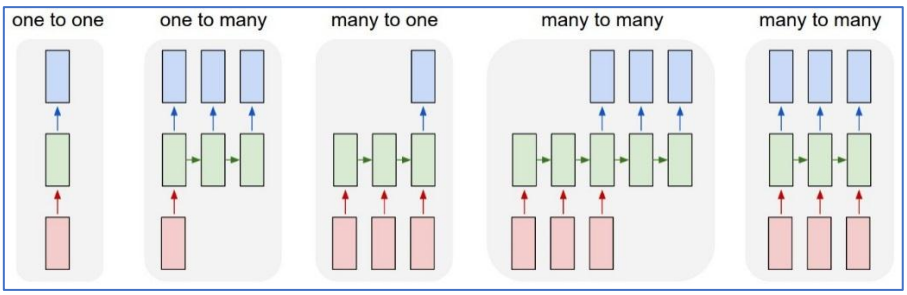


Рисунок 11.55 – Варианты структур рекуррентных нейронных сетей

Примерами рекуррентных нейронных сетей являются:

- ИНС Жордана и Элмана (см. пункт 11.5.5);
- ИНС Хопфилда и Хэмминга (см. пункт 11.5.6 и 11.5.7);
- двунаправленная ассоциативная память Б. Коско (см. пункт 11.5.9);
- ИНС долго-краткосрочной памяти (Long Short-Term Memory, LSTM).

Наиболее известной в настоящее время разновидностью рекуррентных нейронных сетей являются LSTM-сети, способные к запоминанию долговременных зависимостей и основанных на использовании предыдущих состояний для определения текущего.

На рисунке 11.56 приведена структура LSTM-сети.

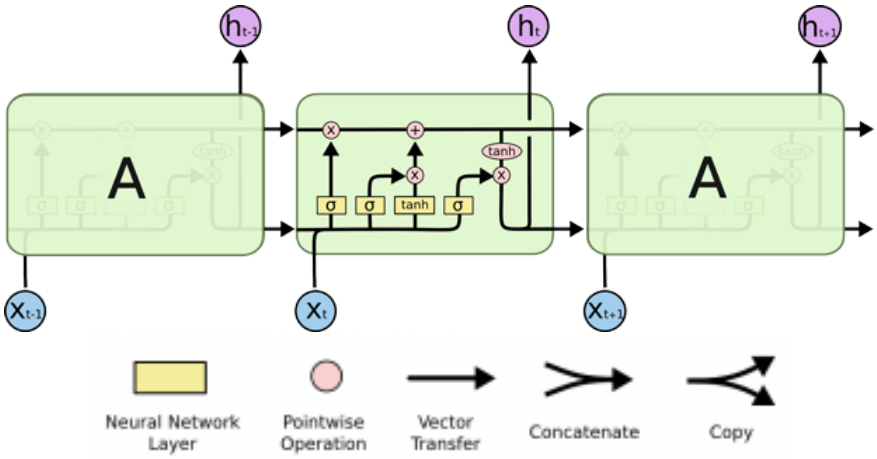


Рисунок 11.56 – Структура LSTM-сети

На рисунке 11.57 и показано состояние ячейки LSTM-сети, а на рисунке 11.58 – механизм разрешения передачи данных.

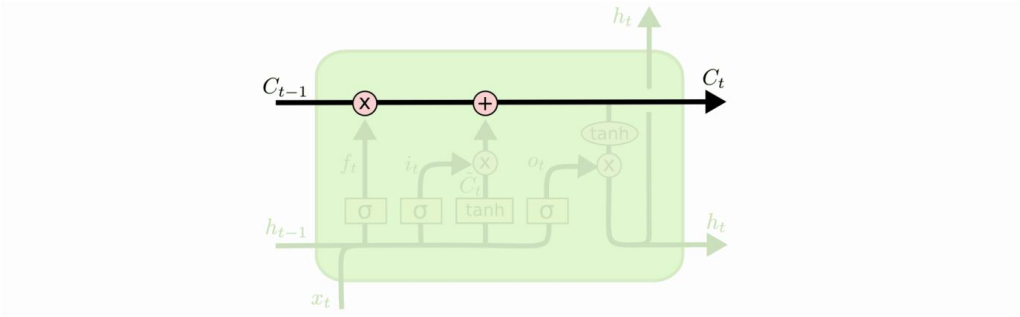


Рисунок 11.57 – Состояние ячейки LSTM-сети

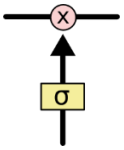
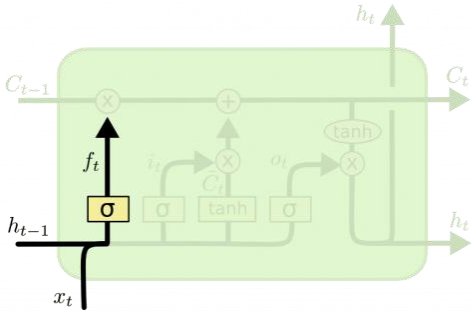


Рисунок 11.58 – Gate-механизм избирательной передачи данных в LSTM-сети

Проиллюстрируем алгоритм работы LSTM-сети на примере одного ее элемента.

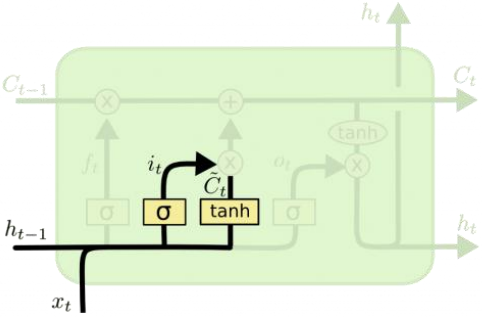
Шаг 1. Механизм забывания (forget gate – гейт забывания), рисунок 11.59.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Рисунок 11.59 – «Гейт забывания» (forget gate):
$\sigma \in [0, 1]$– *сигмоидальная функция,*
«0» – *полностью удалить,* «1» – *полностью сохранить*

Шаг 2. Обновление предыдущего состояния до текущего состояния (рисунок 11.60).

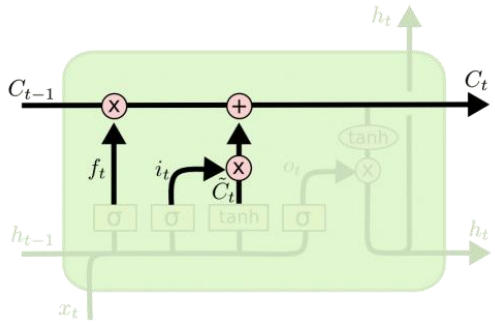


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Рисунок 11.60 – Обновление до текущего состояния C_t

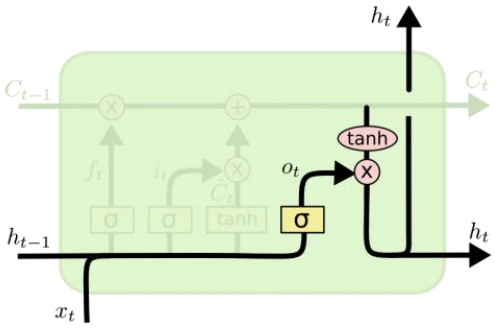
Шаг 3. Фильтрация состояния ячейки (рисунок 11.61).



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Рисунок 11.61 – Формирование отфильтрованного состояния ячейки C_t

Шаг 4. Формирование результата на выходе ячейки (рисунок 11.62).



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Рисунок 11.62 – Вывод результата

Области применения и задачи LSTM-сетей;

- распознавание и предсказание текстов при нефиксированной последовательности слов в текстовых документах;
- моделирование языка;
- машинный перевод;
- обработка аудио, видео и изображений;
- анализ тональности и классификация текстов.

Вопросы для самопроверки

1. Из каких элементов состоит формальный нейрон?
2. Назовите функции активации нейрона.
3. Какие типы нейронов в искусственной нейронной сети можно выделить в зависимости от выполняемых ими функций?
4. В чем заключается необходимость нормализация данных?
5. В чем состоит обучение на основе правила Хебба?
6. Что такое конкурентное обучение ИНС?
7. Какова цель обучения с учителем и в чем заключается алгоритм обратного распространения ошибки?

8. Каковы особенности алгоритма сопряженных градиентов?
9. Какова цель обучения без учителя ИНС?
10. В чем заключается суть алгоритма обучения без учителя самоорганизующихся карт Кохонена?
11. Что из себя представляет обучение с подкреплением?
12. Когда прекращается процесс обучения ИНС?
13. Какими свойствами обладают искусственные нейронные сети?
14. Когда использование искусственной нейронной сети является целесообразным?
15. Назначение, особенности обучения и функционирования ИНС встречного распространения.
16. Назначение, особенности обучения и функционирования ИНС радиальных базисных функций.
17. Назначение, особенности построения, обучения и функционирования ИНС с анализом главных компонентов.
18. Назначение, особенности построения, обучения и функционирования каскадных ИНС.
19. Назначение, особенности построения, обучения и функционирования ИНС Жордана и Элмана.
20. Назначение, особенности построения, обучения и функционирования ИНС Хопфилда.
21. Назначение, особенности построения, обучения и функционирования ИНС Хэмминга.
22. Назначение, особенности построения, обучения и функционирования ИНС адаптивной резонансной теории.
23. Назначение, особенности построения, обучения и функционирования ИНС двунаправленной ассоциативной памяти.
24. Назначение, особенности построения, обучения и функционирования когнитрона.
25. Назначение, особенности построения, обучения и функционирования неокогнитрона.
26. Назначение, особенности построения, обучения и функционирования сверточных нейронных сетей.
27. Назначение, особенности построения, обучения и функционирования рекуррентных нейронных сетей.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Андрейчиков А.В., Андрейчикова О.Н. Интеллектуальные информационные системы. – М. Финансы и статистика, 2006. – 424 с.
2. Борисов В.В., Круглов В.В., Федулов А.С. Нечеткие модели и сети. 2-е изд. стереотип. – М.: Горячая линия – Телеком, 2018. – 284 с.
3. Дорогов В.Г., Теплова Я.О. Введение в методы и алгоритмы принятия решений: учебное пособие. – М.: ИНФРА-М, 2012. - 240 с.
4. Кулик С.Д. Элементы теории принятия решений (критерии и задачи): учебное пособие. – М.: МИФИ, 2010. – 188 с.
5. Мастяева И.Н., Горемыкина Г.И., Семенихина О.Н. Методы оптимальных решений. – М.: ИНФРА-М, 2016. – 384 с.
6. Осовский С. Нейронные сети для обработки информации. – М.: Финансы и статистика, 2004. – 344 с.
7. Рассел С., Норвиг П. Искусственный интеллект: современный подход. 2-е издание. – М.: Вильямс, 2008. – 1424 с.
8. Рутковская Д., Пилинський М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечёткие системы. 2-е изд. – М.: Горячая линия – Телеком, 2012. – 452 с.
9. Сидоркина И. Г. Системы искусственного интеллекта. – М.: Кнорус, 2011. – 248с.
10. Системы поддержки принятия решений: учебник и практикум для бакалавриата и магистратуры / В. Г. Халин [и др.]; под ред. В.Г. Халина, Г.В. Черновой. – М.: Юрайт, 2015. – 494 с.
11. Тихомирова А.Н., Матросова Е.В. Теория принятия решений. – М.: ИНФРА-М, 2017. – 68 с.
12. Уоссерман Ф. Нейрокомпьютерная техника: Теория и практика: Пер. с англ. М.: Мир, 1992. – 240 с.
13. Хайкин С. Нейронные сети: полный курс, 2-е изд., испр. – М.: Вильямс, 2006. – 1104 с.
14. Черноруцкий И.Г. Методы принятия решений: Учеб. пособие. – СПб.: БХВ-Петербург, 2005. – 416 с.

Учебное издание

Борисов Вадим Владимирович,
Бобряков Александр Владимирович,
Мисник Антон Евгеньевич

Экспертные системы
Курс лекций направлению «Информатика и вычислительная техника»
(уровень подготовки – бакалавриат)

Технический редактор М.А. Андреев
Корректор Л.И. Чурлина

Темплан издания филиала НИУ МЭИ в г. Смоленске, 2021 г., учеб. изд.
Подписано в печать 12.02.2021 г.
Формат бумаги 60×84 $^{1}/_{16}$. Тираж 300 экз. Печ. л. 7,00. Усл. печ. л. 6,72.

Издательство «Универсум»
214014, г. Смоленск, ул. Герцена, д.2
ISBN 978-5-91412-464-6