А. В. Бахтин
И. В. Ремизова

ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМИ ПРОЦЕССАМИ

Учебное пособие

Санкт-Петербург
2024

Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Санкт-Петербургский государственный университет промышленных технологий и дизайна»
Высшая школа технологии и энергетики

А. В. Бахтин
И. В. Ремизова

ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМИ ПРОЦЕССАМИ

Учебное пособие

2-е издание, стереотипное

Утверждено Редакционно-издательским советом ВШТЭ СПбГУПТД

Санкт-Петербург
2024

УДК 62-5(075)
ББК 32.965я7
Б 307

Рецензенты:
кандидат технических наук, доцент кафедры АПХП Санкт-Петербургского государственного технологического института (Технического университета)
И. В. Рудакова;
кандидат технических наук, заведующий кафедрой АТПиП Высшей школы технологии и энергетики Санкт-Петербургского государственного университета промышленных технологий и дизайна
Д. А. Ковалев

Бахтин, А. В.
Б 307 Интеллектуальные системы управления технологическими процессами: учебное пособие. — 2-е изд., стереотип. / А. В. Бахтин, И. В. Ремизова. — СПб.: ВШТЭ СПбГУПТД, 2024. — 48 с.

ISBN 978-5-91646-359-0

Учебное пособие соответствует программам и учебным планам следующих дисциплин: «Системы искусственного интеллекта», «Интеллектуальные системы управления технологическими процессами», «Интеллектуальные технологии в автоматизации», «Нейросетевые технологии в автоматизации».

В учебном пособии раскрываются вопросы понятия искусственного интеллекта. Приведены основные виды топологий нейронных сетей и алгоритмов их обучения. Показаны примеры структур систем управления с использованием нейросетевых технологий и особенности их функционирования.

Учебное пособие может быть использовано при самостоятельном изучении курса и подготовке к занятиям студентами всех форм обучения и направлений.

УДК 62-5(075)
ББК 32.965я7

ISBN 978-5-91646-359-0

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Попытки понять сущность сознания предпринимались с древних времен. С тех пор человечество не оставляет попыток создать машины, наделенные интеллектом, т. е. создать искусственный интеллект. Причем человек надеется, что такая машина будет сама решать, какие действия предпринимать в зависимости от ситуации.

Но пока все созданные человеком «думающие» машины принимают решения на основе базовых правил, которые явно или неявно были заложены в них при создании или программировании. Если о машине говорят, что она интеллектуальна, то имеют в виду, что ее способы принятия решений похожи, по крайней мере, до некоторой степени, на те, которые применяет человек.

Классическое понятие интеллекта было предложено А. Тьюрингом более полувека назад, в 1950 г. [1]. Проблема была сформулирована на основе имитационной игры, в которой человек и интеллектуальная машина помещаются в различные комнаты. Исследователь, задавая вопросы, должен определить, кто ему отвечает – человек или машина. Если он не может распознать, кто есть кто, то считается, что машина «интеллектуальна».

Развитие компьютерной техники позволяет использовать в системах управления нейронные сети – одно из новейших направлений реализации искусственного интеллекта. Нейронные сети находят свое применение в системах распознавания образов, обработки сигналов, предсказания и диагностики, в робототехнических и других сложных системах.

Система управления, разработанная на основе нейронных технологий, обладает рядом преимуществ:

- во-первых, нейронная технология управления позволяет строить модели сложных объектов управления по принципу «черного ящика»;
- во-вторых, нейронные модели легко адаптируются при изменении параметров моделируемого объекта;
- в-третьих, они позволяют реализовать модели для многомерных объектов.

ГЛАВА 1. ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

1.1. Процесс мышления

На протяжении последних нескольких десятилетий различными учеными, медиками и физиологами, с одной стороны, и кибернетиками с другой, делались заявления о том, что они понимают, как работает мозг человека. Однако вскоре выяснялось, что они, попросту говоря, «погорячились».

Процесс мышления, протекающий в человеческом сознании, невероятно сложен и пока не поддается расшифровке [2, 3].

Процессы обработки информации в мозгу человека не совпадают с аналогичными процессами в компьютере. Человек получает информацию из внешнего мира от своих пяти органов чувств. Эта информация помещается в буфер кратковременной памяти для анализа (рис. 1). В другой области памяти (долговременной) хранятся символы и смысловые связи между ними, которые используются для объяснения новой информации, поступающей из кратковременной памяти. Важно подчеркнуть, что **в долговременной памяти хранятся не столько факты и данные, сколько объекты и связи между ними,** т. е. символьные образы. При этом доступ к информации в долговременной памяти осуществляется очень эффективно: практически любой элемент данных может быть извлечен в течение цикла обращения и затем преобразован. Например, человек успевает отдернуть руку от горячей печки до того, как получит ожог, и резко вывернуть руль автомобиля при возникновении неожиданного препятствия на дороге. Такой автоматизм действий объясняется использованием образов, ранее запомненных в долговременной памяти.

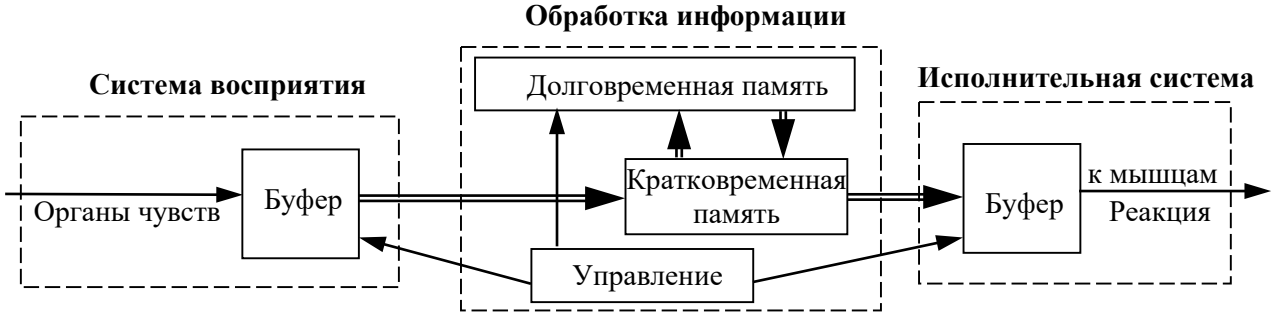


Рисунок 1 – Схема процесса обработки информации в мозге человека

В связи с этим интересен эксперимент, проведенный академиком Н. М. Амосовым, ведущим киевским кардиохирургом и заведующим лабораторией биокибернетики Института кибернетики АН УССР, в 70-е гг. XX в. На самой быстродействующей на тот момент времени машине БЭСМ-6 была проведена попытка смоделировать мозг анаконды, который содержит всего 300 нейронов на обработку и еще столько же – на прием информации. В

результате ответ на внешнее воздействие компьютер считал более часа. При проведении подобного эксперимента на современных компьютерах природная скорость реакции анаконды также не будет достигнута, на основании чего можно сделать вывод, что мозг обрабатывает информацию не так, как машина.

Однако процесс занесения информации в долговременную память достаточно длителен и занимает порядка 7 с для одного образа (за это время устанавливаются все связи, необходимые для его извлечения из памяти в будущем). Обработка информации в кратковременной памяти и перемещение отобранной информации в долговременную занимает 15-20 мин. Например, если после мозговой травмы в результате какой-либо катастрофы долговременная память может полностью восстановиться, то информация, поступившая в последние 15-20 мин. до катастрофы, будет полностью утрачена.

В этом плане можно провести аналогию между кратковременной памятью человека и оперативной памятью компьютера, для которой даже кратковременное отключение электропитания означает полную потерю информации. Долговременная память человека скорее похожа на дисковую память компьютера.

Способ хранения символьных образов в долговременной памяти во многом напоминает способ хранения числовой информации в базе данных сетевого типа. Элементы данных принадлежат поднаборам, наборам и моделям, поднаборы, в свою очередь, принадлежат наборам и моделям и т. д.

Человеческая память хранит не числовые данные, а образы и символы. Символьные образы объединены в мозге в так называемые **чанки** – наборы фактов и связей между ними, запомненные и извлекаемые как единое целое. Чанки хранятся совместно с взаимосвязями между ними. В каждый момент времени человек может обрабатывать и интерпретировать не более 4-7 чанков.

В этом можно убедиться, если попытаться по памяти записать текст:

Система, способная находить эффективные решения, часто неожиданные для пользователя и даже для разработчика.

Совершенно другой результат будет, если в этой фразе переставить слова, тем самым нарушив связи, с помощью которых укрупняются чанки:

пользователя решения и для даже Система, эффективные разработчика неожиданные способная для находить часто.

В последнем случае каждое слово воспринимается как отдельный чанк, содержащий определенное и известное понятие русского языка, и в этой фразе их 13, что превышает объем кратковременной памяти. Тогда как в первой фразе эти же слова были объединены известными связями в более крупные чанки, их там было от 2 до 4. Но если в предыдущем случае во второй фразе были известны хотя бы значения слов, но не ясны связи между ними, то в случае, если и слова не ясны, воспроизвести удастся только несколько символов:

экпырнтесе ытмсеис.

В этой записи содержатся те же буквы, что и в словосочетании «экспертные системы», но слова записаны с переставленными буквами. Этого достаточно, чтобы каждая буква воспринималась как самостоятельный чанк, т. е. в этом случае необходимо запомнить 17 чанков, что значительно превышает

объем кратковременной памяти человека. В то же время в нормальной фразе все объекты (символы) возможно объединить всего в 2 чанка, и запоминание не составит проблем.

Средний специалист в конкретной предметной области помнит 50-100 тысяч чанков, которые и использует в своей профессиональной деятельности. Накопление такого объема информации в долговременной памяти и создание указателей для него требует 10-20 лет.

1.2. Понятие искусственного интеллекта

Одним из аргументов в пользу возможности создания интеллектуальных систем является следующее размышление [3]:

- люди – интеллектуальные системы;
- люди – системы, подчиняющиеся законам физики;
- законы физики могут быть аппроксимированы с любой степенью точности и представлены как программа компьютера. Из этого делается вывод, что интеллектуальное поведение может быть смоделировано машинными программами, т. е. создан искусственный интеллект.

Разработчики первых систем, основанных на знаниях, подчеркивали, что цель искусственного интеллекта – не создание имитатора естественного интеллекта, а разработка машин с интеллектом в буквальном и полном смысле [4]. Свою уверенность они основывали на возможности уже тех компьютеров решать многие формальные проблемы лучше, чем человек. Их оппоненты впадали в другую крайность, требуя создания тогда таких интеллектуальных систем, которые обладали бы всеми чертами **естественного интеллекта, т. е. наличием сознания, преднамеренностью действий, способностью к адаптации, гибкостью решений и т. п.** В результате искусственный интеллект попадал в сложную ситуацию, так как реально было невозможно создать что-либо, одновременно являющееся «интеллектом» и «машиной», и ставилась под сомнение даже попытка создать хотя бы первое приближение к искусственному интеллекту, т. е. такая позиция по отношению к искусственному интеллекту явно непродуктивна. На данный момент недостаточно данных для построения мыслящей искусственной системы, так же, как и данных о работе естественного разума. Отсюда определять интеллект искусственных систем возможно только в **поведенческом** смысле.

Интеллектуальной системой будет считаться система, обладающая хотя бы одной или двумя из перечисленных выше черт естественного интеллекта за исключением наличия сознания.

Проведем сравнительный анализ человеческой когнитивной системы (т. е. системы, основанной на знаниях) и компьютерной системы, которые могут быть описаны на различных уровнях абстрагирования.

Таблица 1 – Уровни описания систем, основанных на знаниях

Уровни описания	Компьютерные системы	Когнитивные системы
Физический	Компьютер представляется множеством элементов, работающих по физическим законам. На этом уровне нет «символов» и «операций», есть только сигналы.	Уровень нейроанатомии, описывает материальные и структурные аспекты нервной системы человека. На этом уровне действуют слабые электрические импульсы – сигналы.
Логический	Компьютер представляет собой сеть логических вентилей (И, ИЛИ и др.) и может быть описана двоичным языком (булевой логики).	Уровень нейрофизиологии, показывает нервную систему как сеть функционально связанных нейронов.
Уровень представления	Уровень символьного машинного языка – ассемблера. Связи между логическими сигналами интерпретируются символами и операторами.	Следуя компьютерной модели, следует предположить существование некоего «ассемблера» в человеческом мозгу. Наиболее известна модель «языка мысли», выдвинутая Д. Фодором в 1975 г.
Коммуникационный	Уровень языков программирования более высокого уровня и интерактивного общения с компьютером.	Уровень коммуникаций и выводов на естественном языке, поэтому этот уровень часто называют лингвистическим.
Ситуационный	Работа компьютера оценивается с позиций решения той или иной проблемы или задания.	Уровень понимания и целенаправленной деятельности.

Эти уровни описания введены для декомпозиции вычислительного и мыслительного процессов и выполнены в функциональных терминах, поэтому система может моделироваться и изучаться на различных уровнях независимо от других. Данная модель не накладывает никаких ограничений на способы ее реализации. Поэтому, если считать ее адекватной, она может быть реализована с помощью искусственных средствах [5].

Основная проблема при реализации такой модели состоит в описании уровня представления. В пользу существования **«языка мысли»** говорит тот факт, что наши мысли не зависят от конкретного языка, на котором они выражаются, и наоборот, лингвистические возможности различных естественных языков имеют те же структурные свойства. Эти два фактора дают основание предполагать, что естественная когнитивная система имеет внутренний язык низкого уровня (врожденный и общий для всего человечества), в терминах которого и протекают когнитивные процессы, в то время, как естественные языки – лишь коммуникационная оболочка системы.

Существуют следующие подходы к построению систем искусственного интеллекта:

1. **На основе обработки символьной информации.** Этот подход предполагает сначала установление законов работы системы естественного интеллекта, которую пытаются воспроизвести, а затем представить их на формальном языке, чтобы реализовать на компьютере [6].

2. **На базе нейронных сетей.** Это альтернативный подход, который позволяет воспроизвести интеллектуальное поведение без наличия его точного формального описания.

Для искусственного интеллекта нет четкого определения, поэтому определим несколько характерных признаков искусственного интеллекта:

- искусственный интеллект – это техника программирования (software techniques), использующая представление информации символами и проводящая символьную ее обработку с учетом отношений между ними. Символьная обработка не лучше и не хуже числовой – она просто другая и, возможно, не более интеллектуальна, чем числовая. Ее можно осуществить с помощью обычных языков программирования, но лучше – с помощью специально разработанных для этой цели (ПРОЛОГ, ЛИСП);

- искусственный интеллект (как наука) является подобластью информатики (computer science) и его основной проблемой является воспроизведение на компьютере человеческих способов сознательного поведения, а конкретнее, способов рассуждения и решения задач и т. п.;

- программы искусственного интеллекта работают с символами, а не с числами, хотя символы могут принимать формы чисел, знаков, слов (могут быть существительными или прилагательными). При этом, в отличие от обычных программ, программы искусственного интеллекта учитывают связи (отношения) между символами. В такой программе информация внутренне взаимосвязана. Эта информация и отношения между символами образуют то, что возможно интерпретировать как смысл или знание.

И здесь необходимо разделить такие понятия, как информация и знание.

Информация – множество отдельных слов и знаков.

Знание – слова и связи между словами, структурирование и отражение отношений.

Например, программа искусственного интеллекта понимает отношение между понятиями «автомобиль», «старт», «бензин». Поэтому она может сделать из этих знаний вывод: если автомобиль не стартует, то потому, что нет бензина. Обычные (числовые) программы тоже могут оперировать с символами, например, базы данных могут использовать термины в текстуальной форме, но они не являются программами искусственного интеллекта, хотя бы потому, что не учитывают отношения между этими терминами.

ГЛАВА 2. ОБЛАСТЬ ПРИМЕНЕНИЯ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

2.1. Основные области исследований по искусственному интеллекту

– Представление знаний – наиболее важная область исследований по искусственному интеллекту, которая является основой для практически всех остальных направлений по искусственному интеллекту, так как любая система искусственного интеллекта основана на использовании знаний о конкретной предметной области. Носителем знаний является человек. Знания также накапливаются в различных литературных источниках, руководящих материалах и т. д. Эти знания должны быть формализованы, чтобы обеспечить их использование в компьютере.

– Решение задач сводится к поиску путей из некоторой исходной точки в целевую точку. Человек делает это с помощью дедуктивного логического вывода, рассуждений, процедурного анализа, аналогии и индукции на базе знаний и своего опыта. Компьютеры пока могут решать задачи только на основе логического вывода и процедурного анализа.

– Экспертные системы представляют собой класс компьютерных программ, которые выдают советы, проводят анализ, дают консультации и ставят диагноз, выполняют классификацию и осуществляют управление. Экспертные системы решают задачи в узкой предметной области (конкретной области экспертизы) на основе дедуктивных рассуждений и способны находить решение неструктурированных и плохо определенных задач [2, 6].

– Средства общения с компьютером на естественном языке. Цель исследований установить принципы взаимодействия между людьми и на их основе создать машины, с которыми можно было бы общаться так же, как с людьми. Здесь можно выделить 4 ключевые проблемы:

– Машинный перевод – использование компьютеров для перевода текстов с одного языка на другой. В 60-х гг., на раннем периоде исследований по искусственному интеллекту, когда казалось, что возможности искусственного интеллекта не ограничены, начали активно использоваться программы машинного перевода. Однако мечта построить хороший переводчик не удалась. Критики часто приводят курьезные примеры таких переводов. Например, при двойном переводе с английского на русский и обратно получилось следующее:

– *The spirit is willing, but the flesh is weak* – Дух силен, а человек – слаб.

– *The wine is good, but the meat is rotten* – Вино – хорошее, а мясо – гнилое.

– Дело в том, что почти все слова имеют несколько значений, определяемых по контексту: spirit – душа, алкоголь; willing – волевой, готовый; flesh – живой человек, мясо; weak – слабый, плохой.

– Информационный поиск – обеспечение с помощью компьютера доступа к информации по конкретной тематике, хранящейся в большой базе данных. В настоящее время поиск ведется по ключевым словам или по контексту, поиск по аналогиям пока невозможен.

– Генерация документов – переработка документов, имеющих определенную форму или заданных на специализированном языке, в эквивалентный документ в другой форме или на другом языке. Например, автоматическое преобразование руководства для инженера по работе с компьютером в такое же руководство, но для врача или студента, не имеющего специальной подготовки, т. е. для «чайника».

– Взаимодействие с компьютером – организация диалога между компьютером и неподготовленным пользователем, что особенно важно для обеспечения взаимодействия между пользователем и, например, экспертной системой или роботом.

– Обучение. Вместо того, чтобы заставлять пользователя преодолевать сложности программирования, проще обучить компьютер сложностям выполнения конкретной задачи, стоящей перед пользователем. В результате появляются пакеты, обладающие зачатками искусственного интеллекта. Например, тестовые редакторы типа Microsoft Word и т. п., способные обучаться и облегчать формирование типовых документов, проверять грамматику и др.

– Когнитивное моделирование. Целью когнитивного моделирования является разработка теории, концепций и моделей человеческого мышления и его функций. Оно позволяет не только диагностировать и лечить психические заболевания, но и выявлять процессы, протекающие в сознании человека при решении задач.

– Робототехника. Первые автоматы в России связывают с именем И. П. Кулибина. Современная робототехника возникла с появлением микропроцессорного управления. В результате на современных предприятиях роботы широко применяются при создании различных машин, в том числе и самих компьютеров. Исследования в области робототехники входят как составная часть в исследования по искусственному интеллекту и ставят целью оснастить компьютеры средствами приема и обработки визуальной информации, средствами манипулирования объектами в некоторой среде. Эти исследования ведутся в следующих трех направлениях:

1) разработка сенсоров, в частности, для визуальной информации, и распознавание информации, поступившей от систем восприятия;
2) создание манипуляторов и систем управления ими;
3) выявление эвристик для решения задач перемещения в пространстве и манипулирования объектами (планирование деятельности робота).

2.2. Понятие нейронной сети

Из различных определений нейронных сетевых моделей здесь приводится наиболее пространное, но более понятное.

Нейронная сеть – параллельная распределенная структура обработки информации, состоящая из обрабатывающих информацию элементов (нейронов), соединенных между собой сигнальными каналами (связями).

Каждый нейрон имеет одну выходную связь, которая может разветвляться до любой желаемой степени и соединять его с необходимым числом других элементов сети. Выходной сигнал элемента может быть любой математической формы, а информационный процесс внутри элемента, его **функция преобразования**, задается произвольно, и его результат зависит от текущих значений сигналов, поступающих на вход элемента, и значений, хранящихся в его локальной памяти.

Первые исследования по нейронным сетям принято относить к началу 1940-х гг., хотя они представляли собой, фактически, описание работы биологических нейронных систем.

Способность нейронной сети к обучению впервые исследована У. МакКаллоком и У. Питтсом. В 1943 г. вышла их работа «Логическое исчисление идей, относящихся к нервной деятельности», в которой была построена модель нейрона и сформулированы принципы построения искусственных нейронных сетей [1].

И лишь в конце 60-х гг. появились сообщения об использовании нейронных сетей для распознавания образов, подавления отражений (эхо) в телефонных сетях и т. п.

Крупный толчок развитию нейрокибернетики дал американский нейрофизиолог Френк Розенблатт, предложивший в 1962 г. свою модель нейронной сети – персептрон. Воспринятый первоначально с большим энтузиазмом, он вскоре подвергся интенсивным нападкам со стороны крупных научных авторитетов. И хотя подробный анализ их аргументов показывает, что они оспаривали не совсем тот персептрон, который предлагал Розенблатт, крупные исследования по нейронным сетям были свернуты. К этому времени относится также критика нейронных сетей М. Минским, разработчиком фреймового метода представления знаний, которая затормозила дальнейшее развитие работ по нейронным сетям на несколько десятилетий.

Началом широкого использования нейронных сетей в технике искусственного интеллекта принято считать 80-е гг.

В 1982 г. американский биофизик Дж. Хопфилд предложил оригинальную модель нейронной сети, названную его именем. В последующие несколько лет было найдено множество эффективных алгоритмов: сеть встречного потока, двунаправленная ассоциативная память и др.

В 1986 г. была предложена конфигурация (архитектура) наиболее часто сейчас используемой многослойной сети с прямым соединением нейронов и обратным распространением ошибок «multilayer feed forward back propagation neural network – BPNN». С тех пор области внедрения нейронных сетей все время неуклонно расширяются.

Первые сообщения о применении нейронных сетей в диагностике и управлении технологическим процессом относятся к началу 90-х гг.

Фактически нейронные сети – это **эмпирические модели,** которые аппроксимируют представление о работе нейронов человеческого мозга. Нейросетевая технология не предназначена для клонирования мозга и создания

его компьютерных копий, но способна моделировать природу для реализации некоторых возможностей естественного интеллекта [3].

Результатом такого моделирования будет являться решение тех проблем, связанных с автоматизацией, которые направлены на компенсацию незнания или непонимания механизма протеканий технологического процесса или трудностей точного его описания, вызванных их сложностью, нелинейностью и тому подобными факторами. Все это затрудняет создание моделей **первого** класса, или **феноменологических моделей,** которые, будучи построенными на основе описания физических и химических явлений в технологическом процессе, обладают высокой робастностью и прогнозирующей способностью. Однако из-за своей сложности они часто не могут быть использованы в системах управления, работающих в реальном времени, требуют больших вычислительных мощностей и сложны в идентификации.

Второй класс моделей – **регрессионные,** используют подход к объекту моделирования как к черному ящику, т. е. требуют не знания процессов, происходящих в нем, а лишь значений входов и соответствующих им выходов. Это фактически эмпирические модели, которые просты, удобны для применения в системах управления. Однако, будучи идентифицированы при определенном состоянии технологического процесса, могут оказаться неадекватными при использовании на том же процессе, но при других его состояниях. **Нейронная сеть является** также **эмпирической моделью**, однако **с улучшенными характеристиками.**

ГЛАВА 3. ПОНЯТИЯ, ВИДЫ И СТРУКТУРЫ НЕЙРОНОВ И ИХ ПРЕОБРАЗУЮЩИХ ФУНКЦИЙ

3.1. Нейроны и связи между ними

Внешне структура нейронной сети напоминает структуру биологической нейронной сети, и большая часть терминологии в этой области появилась из нейронауки, которая исследует мозг и память [1].

Мозг состоит из нейронов, которые являются малыми единицами обработки информации. Природный нейрон (рис. 2) состоит из тела клетки с ядром и протоплазмой, одного или нескольких дендритов, проводящих импульсы к нейрону, и аксона, выводящего импульс из нейрона. Между окончанием аксона и началом дендритов других нейронов находится пространство (щель) – синапс, через которое импульсы с аксона передаются на дендрит другого нейрона, поэтому такая связь называется синаптической.

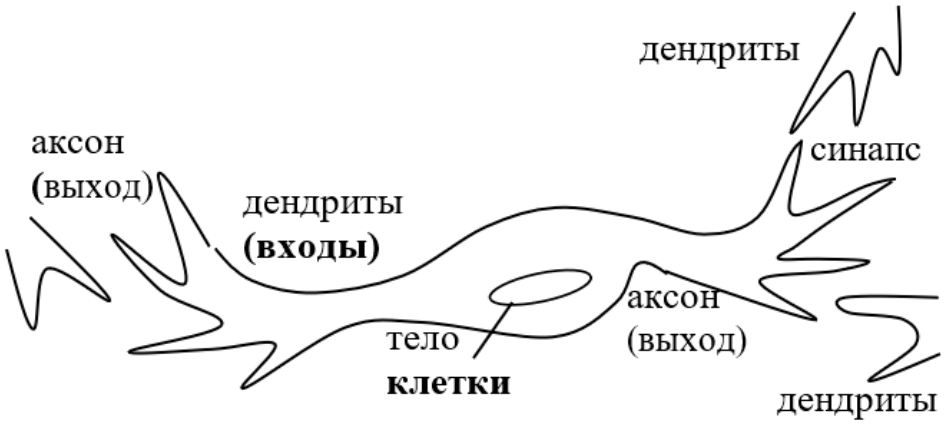


Рисунок 2 – Биологический нейрон

Синапс можно рассматривать как точку соединения, в которой принимают сигналы дендриты. Уникальной способностью нейрона является прием, обработка и передача электрохимических сигналов по нейронной сети.

Импульсы через синапс проходят только в одном направлении. При получении импульса нейрон оценивает его силу; некоторые импульсы игнорируются, некоторые пытаются возбудить нейрон, некоторые – воспрепятствовать этому. Эффект действия всех полученных импульсов суммируется, и, если суммарный эффект превышает некоторый порог, то нейрон возбуждается, выдавая импульс на выход, т. е. посылая по аксону сигнал другим нейронам.

Интенсивность сигнала, получаемого нейроном, а следовательно и возможность его активации, сильно зависит от активности синапсов. Каждый синапс имеет протяженность, и специальные химические вещества передают сигнал вдоль него. Один из самых авторитетных исследователей нейросистем,

Дональд Хебб, высказал идею о том, что обучение заключается в первую очередь в изменениях «силы» синаптических связей.

В человеческом мозге находится около сотни биллионов (10^{11}) нейронов; обычно нейрон контактирует или взаимодействует с 10^4 других, что дает порядка 10^{15} связей. Именно такое большое количество нейронов и многочисленность их связей обеспечивают мощь человеческого интеллекта. Естественно, что компьютеру до этого очень далеко, но работы по созданию машины, моделирующей процессы, подобные мозговой деятельности человека, позволяют достичь ряда полезных результатов.

Поэтому **искусственный нейрон** для успешного моделирования работы естественного должен действовать точно так же (рис. 3).

В нейрон поступает некоторое множество входных сигналов из внешней среды или из других нейронов: обозначим $X = \{x_{ij}\}$, где i – номер входного параметра; j – номер нейрона, т.е. на входы x_{ij} j-го нейрона k-го слоя (уровня) поступает информация $u_{ij}(k\text{-}1)$ с нейронов $(k\text{-}1)$-го уровня. Эта информация умножается в каждом входном канале на свой весовой коэффициент w_{ij} (обозначим $W = \{w_{ij}\}$), характеризующий ее влияние на результат. Собственно, эти веса соответствуют «силе» одной биологической синаптической связи и являются настраиваемыми параметрами сети. В некоторых сетях абсолютное значение весов ограничивается интервалом [0, 1], так как такая нормализация позволяет избежать больших значений на входе нейрона.

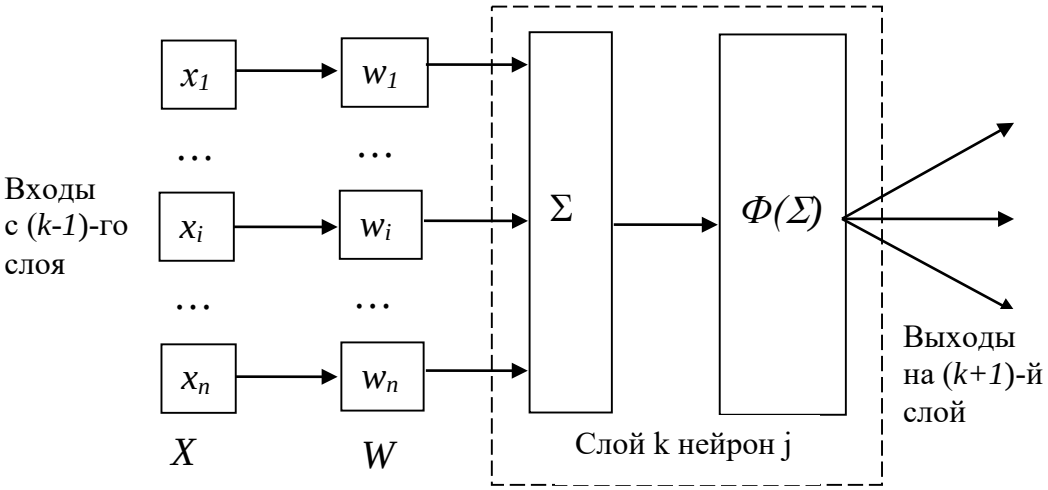


Рисунок 3 – Структура искусственного нейрона

Связи нейронов в сети могут быть как однонаправленными, когда выход одного нейрона является входом для другого, так и двунаправленными, как показано на рис. 4.

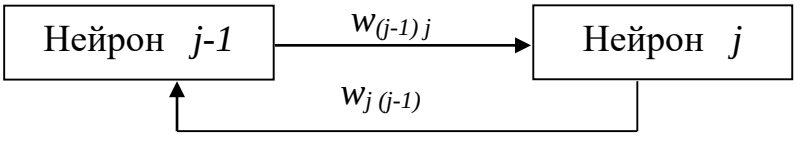


Рисунок 4 – Схема двунаправленной связи между нейронами

Веса двунаправленных связей, как правило, не совпадают: $w_{j(j-1)} \neq w_{(j-1)j}$. Обычно область знаний такой когнитивной системы формируется и хранится как множество значений этих весов, и система обучения новым знаниям строится на поиске оптимальных весов связей.

Аналогично действию тела биологического элемента, взвешенные входы суммируются, а полученная сумма сравнивается с некоторым пороговым значением $\Theta_j(k)$, как в естественном нейроне:

$$Zj(k) = \sum_{i=1}^{n} w_{ij} x_{ij}(k) - \Theta_j(k) = \sum_{i=1}^{n} w_{ij} u_{ij}(k-1) - \Theta_j(k).$$

Если $Z_j(k)>0$, то нейрон генерирует возбуждающий (положительный) сигнал, если $Z_j(k)<0$ – ингибирующий (тормозящий) сигнал (или отсутствие сигнала). Выходной сигнал образуется преобразованием Z_j согласно принятой функции преобразования (активации) нейрона $\Phi(Z_j)$. Наиболее распространенными функциями являются: **пороговая, сигнум, сигмоидальная, гиперболический тангенс, линейная, линейно-пороговая** [1].

1. В **пороговой функции** нейрон остается неактивным до тех пор, пока его вход не достигает порогового значения $\Theta_j(k)$. Когда этот порог достигнут, нейрон возбуждается и посылает выходное дискретное значение:

$$\Phi(\Sigma_j) = \begin{cases} 0, & \text{если } \Sigma_j \le \Theta_j \\ 1, & \text{если } \Sigma_j > \Theta_j \end{cases}.$$

2. Если порог $\Theta_j(k) = 0$, то пороговая функция называется **сигнум-функцией**:

$$\Phi(\Sigma_j) = \begin{cases} 1, & \text{если } \Sigma_j > 0 \\ 0, & \text{если } \Sigma_j = 0 \\ -1, & \text{если } \Sigma_j < 0 \end{cases}.$$

Пороговая функция наиболее точно моделирует нелинейную передаточную характеристику биологического нейрона и дает нейронным сетям большие возможности.

3. С помощью функции преобразования можно выполнять сужение диапазона выходной величины – это так называемые сжимающие функции (рис.5), например: **сигмоидальные,** или S–образные функции, и **гиперболические** функции (тангенс):

Сигмоидальная функция:

$$\Phi(Z_j) = \frac{1}{(1 - e^{-z_j})}.$$

Гиперболический тангенс:

$$\Phi(Z_j) = \frac{(1 - e^{-z_j})}{(1 + e^{-z_j})}.$$

Диапазон изменения $\Phi(Z_j)$ находится в пределах [0,1] для сигмоидальной функци или [-1,1] для функции гиперболического тангенса и может быть при необходимости изменен введением коэффициента масштабирования $g{:}Z_j \rightarrow g \cdot Z_j$.

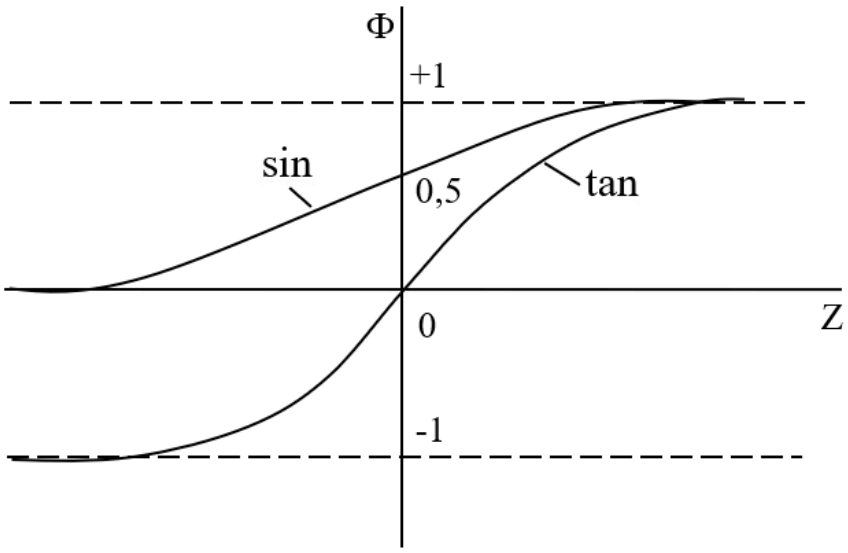


Рисунок 5 – Графики сжимающих функций

Фактически функция преобразования – это нелинейная усилительная характеристика искусственного нейрона. Коэффициент усиления, например, для сигмоидальной функции, выражается наклоном кривой в зависимости от уровня возбуждения и изменяется от малых значений при больших отрицательных возбуждения (приближение к оси Z) до максимального значения при нулевом возбуждении и снова уменьшается при больших положительных возбуждения. Сеть с такой функцией может успешно обрабатывать как слабые сигналы, которые она усиливает, так и сильные, которые нет необходимости усиливать.

4. **Линейная функция** (рис.6) имеет простую форму:

$$\Phi(\Sigma_j) = g\ \Sigma_j,$$

где g – системный параметр, и в большинстве случаев равен 1.

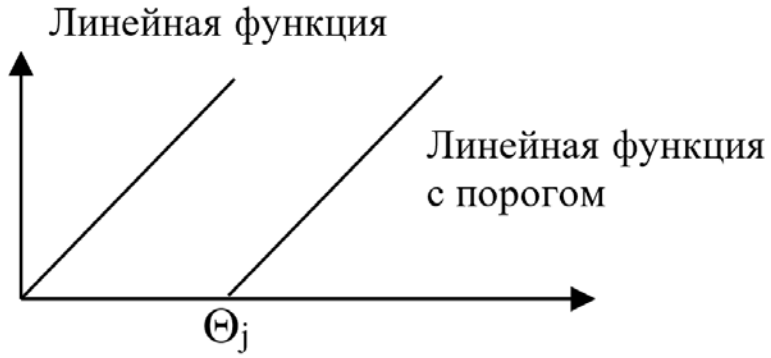


Рисунок 6 – График линейной функции

5. **Линейно-пороговая** функция является формой линейной функции:

$$\Phi(\Sigma_j) = \begin{cases} 0, & если\ \Sigma_j \le \Theta_j \\ \Sigma_j - \Theta_j, & если\ \Sigma_j > \Theta_j \end{cases}.$$

Эта функция допускает возбуждение нейрона только в случае достижения на суммарном входе порогового значения. Для линейных функций выход нейрона не ограничивается верхней полкой диапазона.

Обычно функция $\Phi(Z_j)$ – нелинейная, так как, при линейной $\Phi(Z_j)$ нейронная сеть аналогична обычной регрессии и нет смысла ее применять.

Выход $U_j(k)=\Phi(Z_j)$ искусственного j-го нейрона k-го слоя используется в качестве входа для одного или нескольких вышележащих нейронов.

Выбор функции обычно связан с проектом конкретной нейронной сети. В некоторых инструментальных средах разработчик сети свободен в выборе функции. Однако, даже если вид функции заранее определен, то разработчик должен определить параметры функции (порог Θ_j, системную переменную g) путем постановки экспериментов с конкретной нейронной сетью.

В рассмотренной простой модели искусственного нейрона отсутствуют многие свойства его биологического двойника. Например, в модели не учитываются задержки, которые воздействуют на динамику системы. Входные сигналы сразу же порождают выходной сигнал. Не принимаются во внимание также воздействия функции синхронизации процессов, происходящих в биологическом нейроне, которые ряд исследователей считают решающими. Но несмотря на эти ограничения, сети, построенные из этих нейронов, имеют свойства, очень напоминающие биологическую систему.

3.2. Структура простейшей нейронной сети

Нейроны в сетях группируются слоями. Входной слой состоит из нейронов, которые получают вход из внешней среды. Выходной слой состоит из нейронов, которые связывают выход системы с пользователем или внешней

средой. Обработка знаний в нейронной сети состоит из взаимодействия между слоями нейронов. Проектирование связей между нейронами эквивалентно программированию системы для обработки входа и создания желаемого выхода.

Проектирование нейронной сети состоит из следующих этапов:

1) упорядочение нейронов по слоям;
2) определение связей между нейронами различных слоев, так же как и между нейронами внутри слоя;
3) решение о том, каким образом нейрон получает вход и как создает выход;
4) определение силы связей внутри сети, чтобы узнать соответствующие значение весов путем использования контрольного набора данных.

Существует много способов объединения искусственных нейронов в сеть, для чего используются различные типы связей:

- **полная связь** – каждый нейрон первого слоя связан с каждым нейроном второго слоя;
- **частичная связь** – каждый нейрон первого слоя не обязательно должен быть связан со всеми нейронами второго слоя;
- **однонаправленная связь** – нейроны первого слоя посылают выходы нейронам второго слоя, но не наоборот (может быть полной и частичной);
- **двунаправленная связь** – нейроны первого слоя связаны с нейронами второго слоя, которые, в свою очередь, имеют связь с нейронами первого слоя (может быть полной и частичной);
- **иерархическая связь** – нейроны одного слоя связаны только с нейронами следующего слоя. Если связь не иерархическая, то нейроны одного слоя могут посылать свои выходы не только нейронам следующего слоя, но и другим слоям.

Внутри одного слоя группируются нейроны одного типа, которые могут иметь или не иметь связи между собой. Обычно во многих вариантах топологий нейроны одного слоя не связаны между собой.

Сигмоидальная функция приближается к ступенчатой с порогом $a = 0$ при $a \to +\infty$. Очевидно, пороговая функция более удобна при аппаратной реализации нейрона, тогда как сигмоидальная функция предпочтительна в аналитических исследованиях, поскольку она монотонна, всюду дифференцируема и имеет непрерывные производные любого порядка.

Свойства симметрии гиперболического тангенса относительно точки $Y=0$, а также тот факт, что $F(0)=0$, являются важными для ряда сетей.

Таким образом, каждый нейрон характеризуется вектором весовых множителей и параметрами преобразующей функции. Нейрон способен получать сигналы и в зависимости от их интенсивности и собственных характеристик выдавать выходной сигнал. При этом, если выходной сигнал нейрона близок к единице, то говорят, что нейрон возбужден.

ГЛАВА 4. ТОПОЛОГИИ НЕЙРОННЫХ СЕТЕЙ И ИХ ПРИМЕНЕНИЕ ДЛЯ КОНКРЕТНЫХ ЗАДАЧ

4.1. Объединение нейронов в нейронную сеть

Прежде всего, необходимо сделать несколько замечаний об организации синаптической связи между двумя отдельными нейронами. В простейшем случае синапс не осуществляет преобразования сигнала при передаче. Можно рассматривать усложненные модели синапса. Например, можно предположить какое-либо нелинейное преобразование в нем сигнала или же, предположив наличие у синапса памяти, осуществлять преобразование сигнала с учетом состояния этой памяти. Введением в сеть дополнительных нейронов, выполняющих эти преобразования сигнала, можно свести сеть с усложненными синапсами к сети с простыми синапсами и дополнительными нейронами. Поэтому в дальнейшем следует ограничиться рассмотрением сетей с простейшими связями.

К настоящему времени предложено большое количество способов для объединения нейронов в нейросеть, при этом говорят о «топологии» нейросети. Рассмотрим наиболее важные из них [1]. Без ограничения общности допустим, что нейроны в сети расположены слоями. Обычно выделяют входной слой, на который подается возбуждающий сигнал, выходной слой, с которого снимают переработанный сетью сигнал, а все остальные слои называют скрытыми, поскольку они не видны пользователю.

4.2. Сети прямого распространения – персептроны

Сети этого типа состоят из нескольких слоев нейронов: входного слоя, выходного и нескольких «скрытых» слоев [1]. Нейроны каждого слоя не связаны между собой. Выходной сигнал с каждого нейрона поступает на входы всех нейронов следующего слоя.

Нейроны входного слоя не осуществляют преобразования входных сигналов, их функция заключается в распределении этих сигналов между нейронами первого скрытого слоя (рис. 7).

Функционирование сети прямого распространения чрезвычайно просто. Входной сигнал, подаваемый на сеть, поступает в нейроны входного слоя, проходит по очереди через все слои и выделяется с выходом нейронов выходного слоя. По мере распространения сигнала по сети он претерпевает ряд преобразований, которые зависят от его начального значения, от преобразующей функции и величин весов связей. Пусть сеть состоит из K слоев: одного входного, одного выходного и $(K-2)$ скрытых слоев. Набор выходных сигналов нейронов k-го слоя $(k=1, 2,..., K)$ обозначим U^k. Далее обозначим W^* набор весов синаптических связей, соединяющих нейроны k-го слоя с нейронами $(k+1)$ – го слоя; w_{ij}^k соединяет нейроны u_i^k и u_i^{k+1}(рис. 7) ($u_i^k \in U^k$, $u_i^{k+1} \in U^{k+1}$).

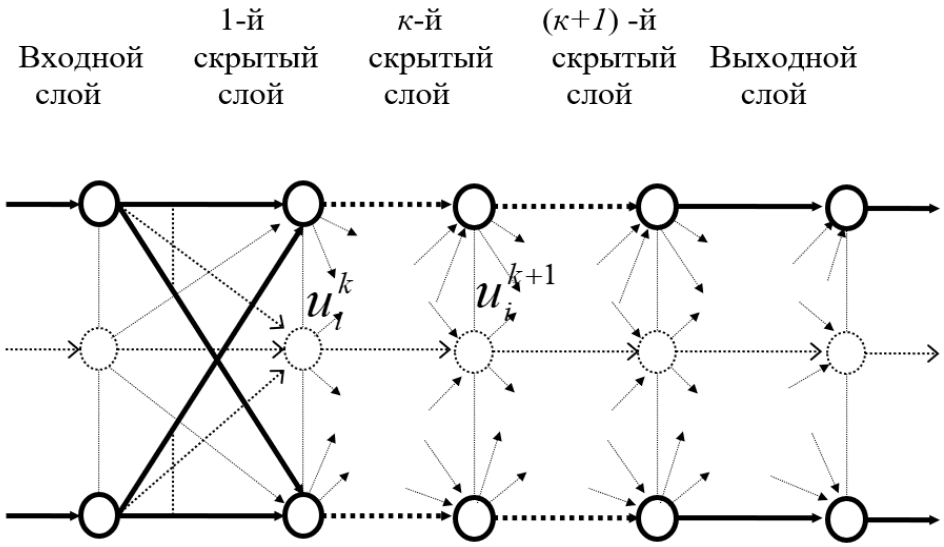


Рисунок 7 – Сеть прямого распространения

Нелинейную преобразующую функцию обозначим как *f(·)*. Тогда прямое функционирование сети прямого распространения описывается следующим соотношением:

$$u_i^{k+1} = f(\sum_{i=1}^{n(k)} w_{ij}^k \cdot u_i^k),$$

где $j = 1,2,.....,n(k+1)$.

4.3. Самоорганизующиеся карты Кохонена

Сеть, топологию которой предложил Тéуво Кохонен [1], выдающийся финский ученый, состоит из двух слоев (рис. 8). Первый из них выполняет функцию распределения входного сигнала между нейронами второго слоя. Нейроны второго слоя, называемого иногда слоем Кохонена, расположены на плоскости и связаны с нейронами своего слоя связями, величина которых зависит от расстояния между нейронами и обычно имеет вид «мексиканской шляпы» (рис. 9).

Такой вид связей обеспечивает взаимное усиление сигнала близкими нейронами и ослабление влияния далеких нейронов, что делает более контрастной границу раздела возбужденных нейронов от остальных, ложное возбуждение которых этим подавляется.

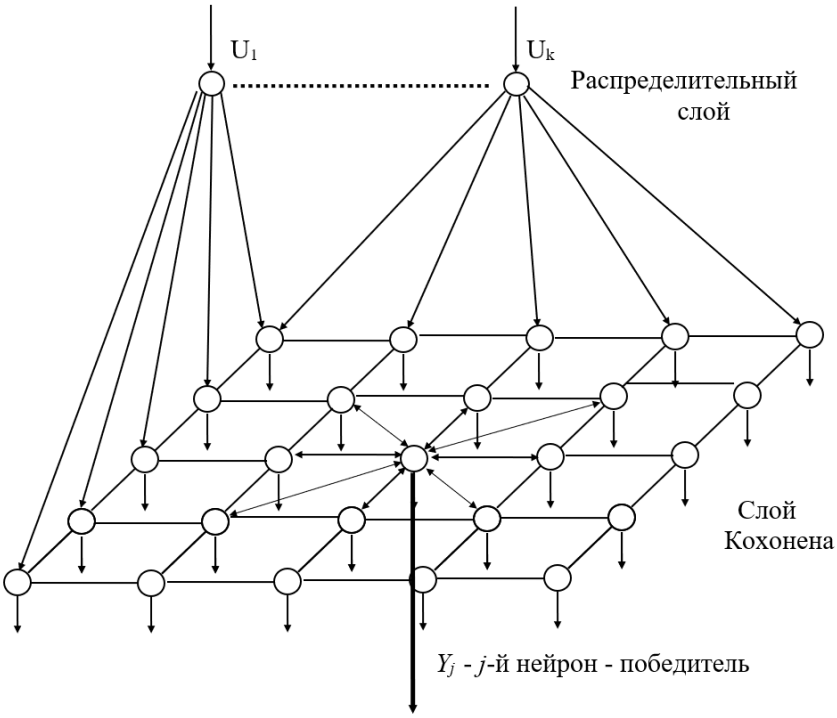


Рисунок 8 – Сеть Кохонена «Kohonen map» (веса связей между нейронами слоев, а также некоторые связи между нейронами слоя Кохонена не показаны)

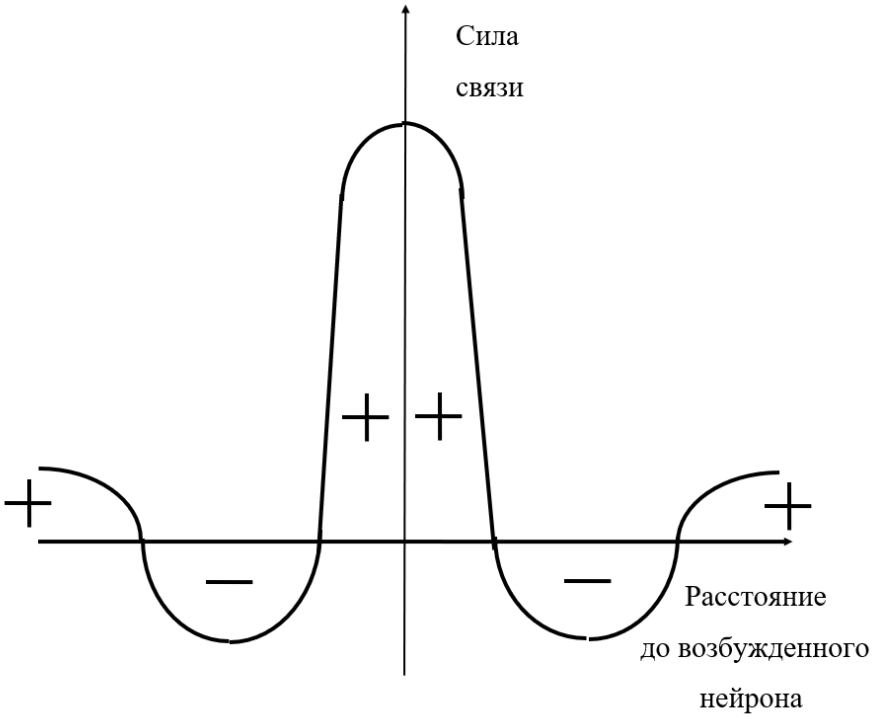


Рисунок 9 – Вид функции взаимного влияния нейронов в слое Кохонена

Результатом работы сети Кохонена при подаче на входной слой некоторого вектора является определение нейрона, который возбужден более других – нейрон-победитель. Этот нейрон более других близок к предъявленному образу, поскольку выход каждого нейрона второго слоя определяется как сумма взвешенных входов сети.

В своей простейшей форме сеть Кохонена функционирует по принципу: «Победитель берет все». Это означает, что для данного входного вектора только один нейрон второго слоя сети выдает на выходе логическую единицу, все остальные выдают ноль. Однако после выделения победителя происходит скрытая от пользователя операция коррекции весов между первым и вторым слоями. Дело в том, что сеть Кохонена обучается без учителя, поэтому каждый новый образ, предъявленный сети, может изменить силы связей.

После предъявления сети достаточного количества образов все нейроны как бы разбиваются на подмножества, каждое из которых «откликается» на образы соответствующего класса, т. е. сеть способна осуществлять классификацию предъявляемых образов, причем переход от одного подмножества к другому происходит непрерывно. В этом заключается свойство сети к обобщению – достаточно правильно распознавать объекты, которые ранее сети не предъявлялись, но в какой-то мере обладающие свойствами известных классов.

4.4. Сети Хопфилда

Сеть Хопфилда – однослойная сеть [1]. Все нейроны связаны друг с другом связями W_{ij} (рис.10), причем сигнал с выхода нейрона может подаваться на его же вход, и необязательно $W_{ij} = W_{ji}$.

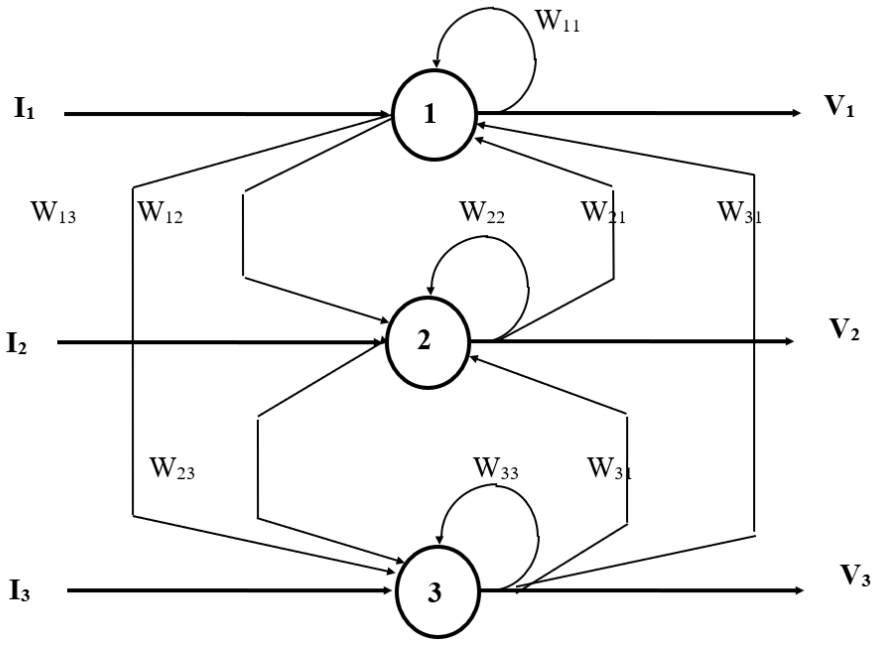


Рисунок 10 – Сеть Хопфилда «Hopfield net» с тремя нейронами

Каждая компонента входного вектора I_i подается на соответствующий i-й нейрон. Поскольку сигнал с выхода каждого нейрона подается на входы всех остальных, входной вектор начинает циркулировать, преобразуясь, по сети до тех пор, пока сеть не придет в устойчивое состояние, т. е. когда все нейроны на каждом последующем цикле будут вырабатывать тот же сигнал, что и на предыдущем. Очевидно, возможны случаи бесконечной циркуляции входного вектора без достижения устойчивого состояния.

Функционирование сети Хопфилда формально можно описать следующими уравнениями:

$$\frac{dUi}{dt} = -U_i + \sum_{j=1}^{n} W_{ji} \cdot V_j + I_i \; i = 1,2,\ldots..,n,$$

$$V_j = F(U_j) \qquad j = 1, 2 \ldots, n,$$

где U_i – значение выхода суммирующего блока i-го нейрона.

Сети Хопфилда имеют многочисленные применения. Ряд из них связан со способностью этих сетей запоминать, а затем восстанавливать даже по неполной входной информации различные образы. Другие применения связаны с возможностью использования сетей Хопфилда для решения оптимизационных задач.

Таким образом, выше было описано три основных способа организации слоистых структур нейронов в сети:

1. Сеть прямого распространения – персептрон: нейроны каждого слоя не связаны между собой. Сигналы передаются только на нейроны следующего слоя.
2. Карта Кохонена: нейроны внутри слоя Кохонена могут иметь неизменяемые тормозящие связи с соседями. Связи с нейронами входного слоя настраиваются в процессе обучения с учетом топологической близости нейронов в слое Кохонена.
3. Сеть Хопфилда: все нейроны связаны друг с другом.

Используя различные сочетания элементов этих структур, можно построить сеть с практически любой из известных к настоящему моменту топологией.

4.5. Другие архитектуры нейросетей

В этом разделе показаны некоторые популярные типы искусственных нейронных сетей. Все они состоят из элементов сетей трех основных типов, рассмотренных ранее.

Когнитрон «Gognitron».

Когнитрон был предложен как модель процесса восприятия человека. Он продемонстрировал значительные способности к распознаванию образов. Когнитрон имеет топологию сети прямого распространения, т. е. он состоит из нескольких слоев, причем на каждый нейрон передается возбуждение только от нейронов из определенной области предыдущего слоя. При обучении

когнитрона настройка связей проводится только у наиболее возбужденного нейрона каждой области. Это соответствует идее самоорганизации Кохонена. Можно сказать, что по сути когнитрон является сетью прямого распространения, каждый слой которой представляет собой самоорганизующуюся карту Кохонена.

Неокогнитрон «Neocognitron».

Неокогнитрон является дальнейшим развитием когнитрона. Неокогнитрон моделирует зрительную систему человека. Он состоит из чередующихся слоев двух типов: S и C. В задачу S-слоев входит распознавание объектов, а C-слой обеспечивает независимость результата распознания от изменения положения образа «инвариантность распознавания».

В отличие от когнитрона, неокогнитрон сочетает самоорганизацию и обучение с учителем, что связано с отличием в настройке S- и C-слоев.

Двунаправленная ассоциативная память «Bidirectional Associative Memory».

Двунаправленная ассоциативная память содержит два слоя Хопфилда, причем каждая связь двунаправлена. Предъявляемый сети образ подается на нейроны первого слоя и начинает циркулировать между слоями, пока сеть не придет в устойчивое положение. При этом на нейронах второго слоя выделяется запомненный образ, который наиболее близок к предъявленному. Обучение этой сети проводится на основе стандартного правила Хебба. По сути, двунаправленная ассоциативная память является прямым обобщением сети Хопфилда.

Сети встречного распространения «Counter propagation Networks».

Одним из наиболее перспективных приложений такой сети является сжатие данных. Сеть встречного распространения состоит из двух слоев: входной слой – самоорганизующаяся карта Кохонена, второй слой – стандартный слой сети прямого распространения. В настоящее время исследования по искусственным нейронным сетям находятся в стадии интенсивного роста. Поэтому любой обзор по искусственным нейронным сетям страдает неполнотой. Число областей, где искусственные нейронные сети успешно применяются, постоянно растет, в связи с чем появляются новые структуры нейронных сетей и методы их обучения.

ГЛАВА 5. МЕТОДЫ, ПРАВИЛА И АЛГОРИТМЫ, ПРИМЕНЯЕМЫЕ ПРИ ОБУЧЕНИИ РАЗЛИЧНЫХ ТОПОЛОГИЙ СЕТЕЙ

5.1. Методы обучения нейронных сетей

Решение задачи на нейрокомпьютере принципиально отличается от решения той же задачи на обычном компьютере с фон Неймановской архитектурой. Решение задачи на обычном компьютере заключается в обработке вводимых данных в соответствии с программой. Программу составляет человек. Для составления программы нужно продумать алгоритм, т.е. определенную последовательность математических и логических действий, необходимых для решения этой задачи. Алгоритмы, как и программы, разрабатываются людьми, а компьютер используется лишь для выполнения большого количества элементарных операций: сложения, умножения, проверки логических условий и т. п.

Нейрокомпьютер же используется как «черный ящик», который можно обучить решению задач из какого-нибудь класса. Нейрокомпьютеру «предъявляются» исходные данные задачи и ответ, который соответствует этим данным и который был получен каким-либо способом. Нейрокомпьютер должен сам построить внутри «черного ящика» алгоритм решения этой задачи, чтобы выдавать ответ, совпадающий с правильным. Кажется естественным ожидать, что чем больше различных пар «исходные данные – ответ» будет предъявлено нейрокомпьютеру, тем адекватнее решаемой задаче он сконструирует модель.

После обучения нейрокомпьютера и предъявления ему исходных данных, которых он раньше не встречал, он тем не менее выдает правильное решение – в этом заключается способность нейрокомпьютера к обобщению.

Поскольку в основе нейрокомпьютера лежит искусственная нейронная сеть, то процесс обучения состоит в настройке параметров этой сети. При этом, как правило, топология сети считается неизменной, а к подстраиваемым параметрам обычно относятся параметры нейронов и величины синаптических весов. К настоящему времени в литературе под обучением принято понимать процесс изменения весов связей между нейронами.

Рассмотрим два направления классификации методов обучения сетей [1].

Первое направление – **по способам использования учителя.**

С учителем: сети предъявляются примеры входных данных и выходных. Сеть преобразует входные данные и сравнивает свой выход с желаемым. После этого проводится коррекция весов с целью получить лучшую согласованность выходов.

Обучение с последовательным подкреплением знаний: в этом случае сети не дается желаемое значение выхода, а вместо этого сети ставится оценка, хорош выход или плох.

Обучение без учителя: сеть сама вырабатывает правила обучения путем выделения особенностей из набора входных данных.

Второе направление классификации методов обучения – **по использованию элементов случайности.**

Детерминистские методы: в них шаг за шагом осуществляется процедура коррекции весов сети, основанная на использовании их текущих значений, например, значений желаемых выходов сети (алгоритм обучения, основанный на обратном распространении ошибки, является примером детерминистского обучения).

Стохастические методы обучения: основываются на использовании случайных изменений весов в ходе обучения (алгоритм Больцмановского обучения является примером стохастического обучения).

5.2. Правила обучения нейросетей

Правила обучения определяют закон, по которому сеть должна изменить свои синаптические веса в процессе обучения.

Правило Д. Хебба

Большинство методов обучения основываются на общих принципах обучения нейросетей, развитых Дональдом Хеббом. Принцип Хебба можно сформулировать следующим образом: «Если два нейрона одновременно активны, увеличьте силу связи между ними», что можно записать как

$$dW_{ij} = gf(Y_i) \cdot f(Y_j),$$

где dW_{ij} – величина изменения синапса W_{ij} ;

Y_i – уровень возбуждения i-го нейрона;

Y_j – уровень возбуждения j-го нейрона;

$f(.)$ – преобразующая функция;

g – константа, определяющая скорость обучения.

Дельта – правило

Дельта – правило, известное, как правило снижения квадратичной ошибки, было предложено Видроу-Хоффом [1]. Дельта – правило используется при обучении с учителем.

$$dW_{ij} = g \cdot (D_j - Y_j) \cdot Y_i,$$

где D_j – желаемый выход j-го нейрона.

Таким образом, изменение силы связей происходит в соответствии с ошибкой выходного сигнала ($D_j - Y_j$) и уровнем активности входного элемента Y. Обобщение дельта – правила, называемое обратным распространением ошибки, используется в нейронных сетях с двумя и более слоями.

ART – правило

Теория адаптивного резонанса (ART) была развита в [1]. Теория адаптивного резонанса – это обучение без учителя, когда самоорганизация происходит в результате отклика на выбор входных образов. ART-сеть способна к классификации образов. Теория адаптивного резонанса использует концепцию долговременной и кратковременной памяти для обучения нейронной сети. В долговременной памяти хранятся реакции на образы, которым сеть была обучена, в виде векторов весов. В кратковременную память помещаются

текущий входной образ, ожидаемый образ, классификация входного образа. Ожидаемый образ выбирается из долговременной памяти всякий раз, когда на вход нейронной сети подается новый образ. Если они схожи в соответствии с определенным критерием, сеть классифицирует его как принадлежащий к существующему классу. Если они различны, формируется новый класс, в котором входной вектор будет первым членом класса.

Такое обучение называют состязательным обучением. Простейший тип состязательного обучения определяется правилом «победитель берет все», т. е. ансамбль с лучшим выходом активизируется, остальные – подавляются.

Элемент с наибольшим уровнем активации называют «победитель». Когда он выбран, нейронная сеть добавляет черты вводимого образа в состав долговременной памяти путем повторного прогона вперед – назад через веса долговременной памяти. Этот процесс С. Гроссберг назвал резонансом.

Правило Кохонена

Т. Кохонен использовал концепцию состязательного обучения для развития обучающего правила «без учителя» в нейронных сетях типа карты Кохонена (см. рис. 8).

Правило Кохонена заключается в следующем. Сначала выбирается победитель по стратегии «победитель берет все». Поскольку выход j-го нейрона определяется скалярным произведением (U, Wj) входного вектора U с вектором весов связей между входным слоем и j-м нейроном, то он зависит от угла между векторами U, Wj. Поэтому выбирается нейрон, вектор весов Wj которого наиболее близок ко входному вектору U. Другими словами, выбирается наиболее активный нейрон. Далее конструируется новый вектор Wj так, чтобы он был ближе к входному вектору U:

$$Wijnew = Wijold + g\,(U - Wijold),\ i = 1,2,\ldots,k,$$

где k – количество входов сети;

g – константа обучения.

Больцмановское обучение

Больцмановское обучение состоит в подкреплении знаний в соответствии с целевой функцией изменения выхода нейронной сети. Это обучение использует вероятностную функцию для изменения весов. Эта функция обычно имеет вид распределения Гаусса, хотя могут использоваться и другие распределения.

Больцмановское обучение выполняется в несколько этапов:

1. Коэффициенту T присваивают большое начальное значение.

2. Через сеть пропускают входной вектор и по выходу вычисляют целевую функцию.

3. Случайным образом изменяют вес в соответствии с распределением Гаусса:

$$P(x)=exp(-x^2/T^2),$$

где x – изменение веса.

4. Снова вычисляют выход и целевую функцию.

5. Если значение целевой функции уменьшилось, т.е. улучшилось, то сохраняют изменение веса. Если же нет и величина ухудшения целевой функции составляет ΔC, то вероятность сохранения изменения веса вычисляется следующим образом.

Величина $P(\Delta C)$ – вероятность изменения ΔC в целевой функции – определяется с использованием распределения Больцмана:

$$P(\Delta C) \sim exp(-\Delta C/kT),$$

где k – константа, аналогичная константе Больцмана, выбирается в зависимости от условий задачи.

Затем выбирают случайное число V, используя равномерное распределение от нуля до единицы. Если $P(\Delta C) > V$, то изменение веса сохраняется, иначе изменение веса равно нулю.

Шаги 3 – 5 повторяют для каждого из весов сети, при этом постепенно уменьшают T, пока не будет достигнуто приемлемо низкое значение целевой функции. После этого повторяют весь процесс обучения для другого входного вектора. Сеть обучается на всех векторах, пока целевая функция не станет допустимой для всех них. При этом для обеспечения сходимости изменение T должно быть пропорциональным логарифму времени t:

$$T(t) = T(0) / log(1+t).$$

Это означает, что скорость сходимости целевой функции невелика, следовательно, время обучения может быть очень большим.

5.3. Алгоритмы обучения нейросетей

Алгоритм обратного распространения – это итеративный градиентный алгоритм, который используется с целью минимизации среднеквадратичного отклонения текущего выхода многослойного персептрона и желаемого выхода. Он используется для обучения многослойных нейронных сетей с последовательными связями.

В нейронных сетях применяются несколько вариантов сигмоидальных передаточных функций.

Функция Ферми (экспоненциальная сигмоида):

$$f(s) = \frac{1}{1 + e^{-2\alpha s}},$$

где s – выход сумматора нейрона;

α – некоторый параметр.

Рациональная сигмоида:

$$f(s) = \frac{s}{|s| + \alpha}.$$

Гиперболический тангенс:

$$f(s) = th\frac{s}{\alpha} = \frac{e^{\frac{s}{\alpha}} - e^{-\frac{s}{\alpha}}}{e^{\frac{s}{\alpha}} + e^{-\frac{s}{\alpha}}}.$$

Сигмоидальные функции являются монотонно возрастающими и имеют отличные от нуля производные на всей области определения. Эти характеристики обеспечивают правильное функционирование и обучение сети. Наиболее эффективной передаточной функцией является рациональная сигмоида. Для вычисления гиперболического тангенса требуется больше всего тактов работы процессора.

Функционирование многослойной сети выполняется в соответствии с формулами:

$$s_{t_m} = \sum_{t_{m-L}=1}^{N_{m-L}} w_{t_m t_{m-L}} y_{t_{m-L}} - b_{t_m},\ i_m = 1,2,...,N_m,\ m = 1,2,...,L,$$

$$y_{t_m} = f(s_{t_m}),\ i_m = 1,2,...,N_m,\ m = 1,2,...,L,$$

где s – выход сумматора;

w – вес связи;

y – выход нейрона;

b – смещение;

i – номер нейрона;

N – число нейронов в слое;

m – номер слоя;

L – число слоев;

f – функция активации.

Согласно методу наименьших квадратов, минимизируемой целевой функцией ошибки нейронной сети является величина:

$$E(w) = \frac{1}{2}\sum_{j,p}(y_{j,p}^{(N)} - d_{j,p})^2,$$

где $y_{j,p}^{(N)}$ – реальное выходное состояние нейрона j выходного слоя N нейронной сети при подаче на ее входы p-го образа;

$d_{j,p}$ – идеальное (желаемое) выходное состояние этого нейрона.

Суммирование ведется по всем нейронам выходного слоя и по всем обрабатываемым сетью образам. Минимизация ведется методом градиентного спуска, что означает подстройку весовых коэффициентов следующим образом:

$$\Delta w_{ij}^{(n)} = -\eta \cdot \frac{\partial E}{\partial w_{ij}}, \qquad (1)$$

где w_{ij} – весовой коэффициент синаптической связи, соединяющей i-й нейрон слоя $n-1$ с j-м нейроном слоя n;

η – коэффициент скорости обучения, $0<\eta<1$.

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j} \cdot \frac{dy_j}{ds_j} \cdot \frac{\partial s_j}{\partial w_{ij}}. \qquad (2)$$

Здесь под y_j, как и раньше, подразумевается выход нейрона j, а под s_j – взвешенная сумма его входных сигналов, аргумент активационной функции. Так как множитель dy_j/ds_j является производной этой функции по ее аргументу, из этого следует, что производная активационной функции должна быть определена на всей оси абсцисс. В связи с этим функция единичного скачка и прочие активационные функции с неоднородностями не подходят для рассматриваемых нейронных сетей. В них применяются такие гладкие функции, как гиперболический тангенс или классический сигмоид с экспонентой. В случае гиперболического тангенса

$$\frac{dy}{ds} = 1 - s^2.$$

Третий множитель $\partial s_j/\partial w_{ij}$, очевидно, равен выходу нейрона предыдущего слоя $y_i^{(n-1)}$. Что касается первого множителя в (2), он легко раскладывается следующим образом:

$$\frac{\partial E}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{ds_k} \cdot \frac{\partial s_k}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{ds_k} \cdot w_{jk}^{(n+1)}.$$

Здесь суммирование по k выполняется среди нейронов слоя $n+1$.

Вводится новая переменная:

$$\delta_j^{(n)} = \frac{\partial E}{\partial y_j} \cdot \frac{dy_j}{ds_j}.$$

Далее получаем рекурсивную формулу для расчетов величин $\delta_j^{(n)}$ слоя n из величин $\delta_k^{(n+1)}$ более старшего слоя $n+1$.

$$\delta_j^{(n)} = \left[\sum_k \delta_k^{(n+1)} \cdot w_{jk}^{(n+1)} \right] \cdot \frac{dy_j}{ds_j}. \qquad (3)$$

Для выходного же слоя:

$$\delta_l^{(N)} = (y_l^{(N)} - d_l) \cdot \frac{dy_l}{ds_l}. \qquad (4)$$

Теперь можно записать (1) в раскрытом виде:

$$\Delta w_{ij}^{(n)} = -\eta \cdot \delta_j^{(n)} \cdot y_i^{(n-1)}. \qquad (5)$$

Иногда для придания процессу коррекции весов некоторой инерционности, сглаживающей резкие скачки при перемещении по поверхности

целевой функции, (5) дополняется значением изменения веса на предыдущей итерации:

$$\Delta w_{ij}^{(n)}(t) = -\eta \cdot (\mu \cdot \Delta w_{ij}^{(n)}(t-1) + (1-\mu) \cdot \delta_j^{(n)} \cdot y_i^{(n-1)}), \quad (6)$$

где μ – коэффициент инерционности;

t – номер текущей итерации.

Таким образом, полный алгоритм обучения нейронной сети с помощью процедуры обратного распространения строится так:

1. На входы сети подается один из возможных обучающих примеров и, в режиме обычного функционирования нейронной сети, когда сигналы распространяются от входов к выходам, следует рассчитать значения последних:

$$s_j^{(n)} = \sum_{i=0}^{M} y_i^{(n-1)} \cdot w_{ij}^{(n)},$$

где M – число нейронов в слое n-1 с учетом нейрона с постоянным выходным состоянием +1, задающего смещение; $y_i^{(n-1)}=x_{ij}^{(n)}$ – i-й вход нейрона j слоя n.

$y_j^{(n)} = f(s_j^{(n)})$, где $f()$ – сигмоидальная функция;

$y_q^{(0)}=I_q$, где I_q – q-я компонента вектора входного образа.

2. Рассчитываются $\delta^{(N)}$ для выходного слоя по формуле (4).

Рассчитываются по формуле (5) или (6) изменения весов $\Delta w^{(N)}$ слоя N.

3. По формулам (3) и (5) или (3) и (6) рассчитываются, соответственно, $\delta^{(n)}$ и $\Delta w^{(n)}$ для всех остальных слоев, $n=N-1,...,1$.

4. Корректируются веса связей сети:

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

5. Если ошибка сети существенна, перейти на шаг 1. Иначе – конец.

Сети на шаге 1 попеременно в случайном порядке предъявляются все тренировочные образы, чтобы сеть, образно говоря, не забывала одни по мере запоминания других (рис. 11).

Классический метод обратного распространения относится к алгоритмам с линейной сходимостью. Для увеличения скорости сходимости необходимо использовать матрицы вторых производных функции ошибки.

Алгоритм обучения нейронной сети с помощью процедуры обратного распространения – первый эффективный алгоритм обучения многослойных нейронных сетей, а также один из самых популярных алгоритмов обучения, с его помощью были решены и решаются многочисленные практические задачи.

Также существуют многочисленные модификации алгоритма обратного распространения, которые связаны с использованием различных функций ошибки, различных процедур определения направления и величины шага:

1) функции ошибки:

- интегральные функции ошибки по всей совокупности обучающих примеров;
- функции ошибки целых и дробных степеней;

2) процедуры определения величины шага на каждой итерации:

- инерционные соотношения;
- отжиг;

3) процедуры определения направления шага:

- с использованием матрицы производных второго порядка (метод Ньютона и др.);
- с использованием направлений на нескольких шагах (партан метод и др.).

5.4. Описание алгоритма «Delta Bar Delta»

Алгоритм «Delta Bar Delta» был создан Р. Джекобсом с целью ускорения обучения сети за счет использования эвристического подхода [1]. Алгоритм использует предыдущие значения градиента функции. Зная эту информацию, он совершает изменения в пространстве весов с помощью ряда эвристических правил.

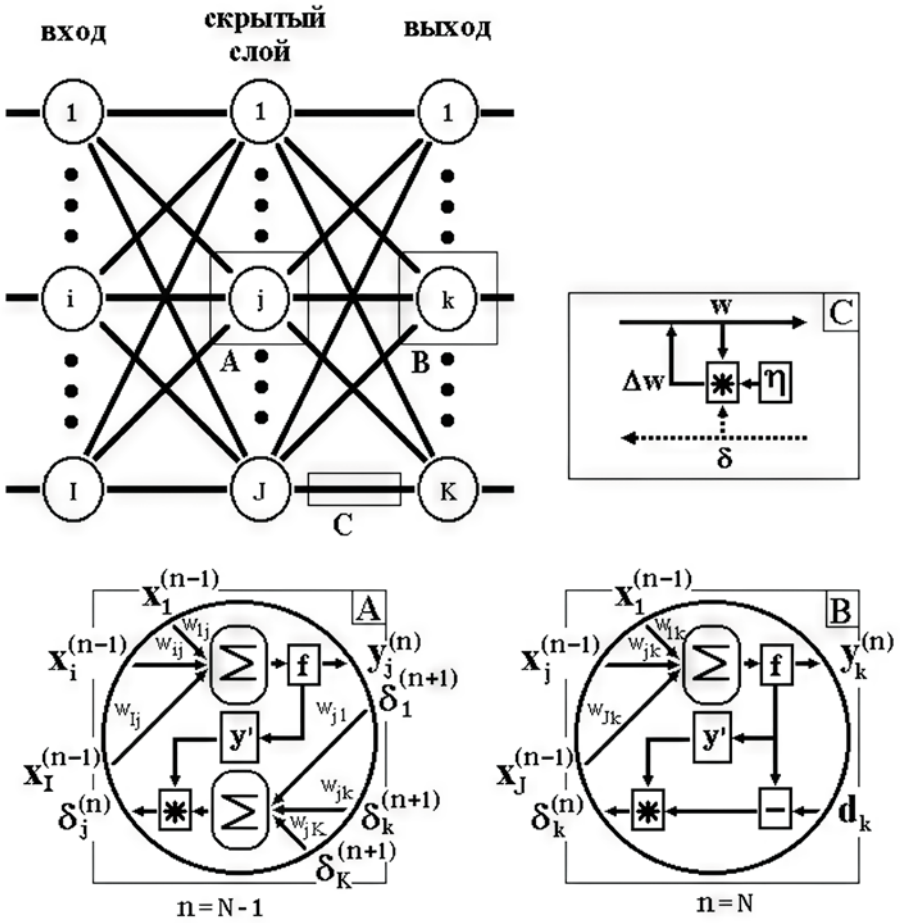


Рисунок 11 – Диаграмма сигналов в сети при обучении по алгоритму обратного распространения ошибки

Опыт показывает, что размерности пространства весов могут значительно различаться с точки зрения общей поверхности ошибки. Якобс предложил ряд эвристик, суть которых в том, что каждый вес должен изменяться в соответствии со своей индивидуальной скоростью обучения. Суть в том, что размер шага обучения для одного веса не всегда подходит в качестве единого шага обучения для всех весов. Более того, этот размер может со временем изменяться.

Первые эвристики по изменению индивидуальных шагов обучения были предложены Х. Кестеном. Он предложил, что если последовательные изменения веса имеют противоположные знаки, то значит, данный вес осциллирует, и, следовательно, скорость обучения должна быть уменьшена.

Позже Д. Садирис предложил следующую эвристику: если серия последовательных изменений веса имеет одинаковые знаки, то скорость обучения должна быть увеличена.

$$w[i+1] = w[i] + a[i]*g[i],$$

где $w[i]$ и $w[i+1]$ – значение веса на шаге i и $i+1$;
$a[i]$ – коэффициент скорости обучения на шаге i;
$g[i]$ – градиент изменения веса на шаге i.

Расчет среднего изменения градиента на шаге i:

$$g_{av}[i] = (1 - convex)*g[i] + convex*g[i-1],$$

где g_{av} – взвешенное среднее изменение градиента на шаге i;
$convex$ – фактор выпуклости весов.

Расчет изменения скорости обучения на шаге i:

$$da[i] = \begin{cases} k_1 & \text{если } g_{av}[i-1]*g[i] > 0 \\ -k_2*a[i] & \text{если } g_{av}[i-1]*g[i] < 0 \\ 0 & \text{если } g_{av}[i-1]*g[i] = 0, \end{cases}$$

где $da[i]$ – изменение скорости обучения на шаге i;
k_1 – константа увеличения скорости обучения;
k_2 – константа уменьшения скорости обучения.

Линейное увеличение изменения скорости позволяет избежать быстрого роста скорости. Геометрическое уменьшение предполагает, что скорость обучения всегда положительная. Более того, скорость может уменьшаться быстрее на участках с сильной нелинейностью.

Парадигма «Delta Bar Delta» является попыткой ускорить процесс сходимости алгоритма обратного распространения за счет использования дополнительной информации об изменении параметров и весов во время обучения.

Стандартный алгоритм «Delta Bar Delta» не использует эвристики, основанной на моменте.

Даже небольшое линейное увеличение коэффициента может привести к значительному росту скорости обучения, что вызовет скачки в пространстве весов.

Геометрическое уменьшение коэффициента иногда оказывается недостаточно быстрым.

5.5. Алгоритм «Extended Delta Bar Delta»

Алгоритм «Extended Delta Bar Delta» был создан Э. Минаем и Р. Вильямсом как естественное продолжение работы Якобса. Ими был использован параметр момент связи «momentum», представляющий собой некоторое число, пропорциональное предыдущему изменению веса. Авторы использовали значения момента для ускорения обучения с помощью ряда эвристических правил.

Изменение веса на последующем шаге:

$$dw[i+1] = a[i]*g[i] + m[i]*dw[i];$$
$$w[i+1] = w[i] + dw[i+1],$$

где $dw[i]$ – изменение веса на шаге i;
$w[i]$ – значение веса на шаге i;
$a[i]$ – коэффициент скорости обучения на шаге i;
$g[i]$ – градиент изменения веса на шаге i;
$m[i]$ – значения момента на шаге i.

Расчет среднего изменения градиента на шаге i:

$$g_{av}[i] = (1 - convex)*g[i] + convex*g[i-1],$$

где g_{av} – взвешенное среднее изменение градиента на шаге i;
$convex$ – фактор выпуклости весов.

Расчет изменения скорости обучения на шаге i:

$$da[i] = \begin{cases} k_{a1}*exp(-y_a*|g_{av}[i]|) & \text{если } g_{av}[i-1]*g[i] > 0 \\ -k_{a2}*a[i] & \text{если } g_{av}[i-1]*g[i] < 0 \\ 0 & \text{если } g_{av}[i-1]*g[i] = 0, \end{cases}$$

где $da[i]$ – изменение скорости обучения на шаге i;
k_{a1} – фактор масштабирования скорости обучения;
y_a – экспоненциальный фактор скорости обучения;
k_{a2} – фактор масштабирования скорости обучения.

Расчет изменения момента на шаге i:

$$dm[i] = \begin{cases} k_{m1}*exp(-y_m*|g_{av}[i]|) & \text{если } g_{av}[i-1]*g[i] > 0 \\ -k_{m2}*m[i] & \text{если } g_{av}[i-1]*g[i] < 0 \\ 0 & \text{если } g_{av}[i-1]*g[i] = 0, \end{cases}$$

где $dm[i]$ – изменение значения момента на шаге i;
k_{m1} – фактор масштабирования момента;
y_m – экспоненциальный фактор момента;
k_{m2} – фактор масштабирования момента.

Коэффициенты скорости обучения и скорости изменения момента имеют различные константы, контролирующие их увеличение и уменьшение.

Для всех связей принимаются следующие ограничения:

$$a[i] < a_{max},$$

где a_{max} – верхняя граница скорости обучения;

$$m[i] < m_{max},$$

где m_{max} – верхняя граница момента.

Если текущая ошибка превышает минимальную предыдущую ошибку с учетом максимального отклонения, то все связи восстанавливаются для наилучшего варианта и коэффициенты обучения и момента уменьшаются.

Обучение сетей Кохонена, построение карт признаков

Для построения карты Кохонена требуется достаточно представительная выборка обучающих векторов признаков (U). Пусть каждый вектор U множества (U) имеет размерность k: $U=(U_1 , U_2 , \ldots ,U_k)$.

Тогда первый «распределительный» слой сети Кохонена должен иметь k нейронов; n нейронов второго слоя «карты» располагаются на плоскости в какой-либо регулярной конфигурации, например, из квадратной прямоугольной сетки (см. рис. 8). Настраиваемым связям между нейронами первого и второго слоев W_{ij} присваиваются случайные значения.

Здесь индекс i обозначает номер нейрона первого слоя, индекс j – номер нейрона второго слоя. До начала обучения задают функцию влияния нейронов второго слоя друг на друга $g(r,t)$, где r – расстояние между нейронами; t – параметр, характеризующий время обучения.

Эта функция традиционно имеет вид «мексиканской шляпы» (см. рис. 9), которую в процессе обучения, по мере увеличения параметра t, делают более «узкой». Однако часто используют более простые функции, например:

$$g(r,t) = \begin{cases} 1, if r < D/t, & t = 1,2,3... \\ 0, if r > D/t, & t = 1,2,3... \end{cases},$$

где D – константа, характеризующая начальный радиус положительного пика «мексиканской шляпы».

Каждый цикл обучения заключается в поочередном предъявлении сети векторов обучающего множества с последующей корректировкой весов W_{ij}. Корректировка осуществляется следующим образом.

1. При появлении на входе сети очередного обучающего вектора U сеть вычисляет отклик нейронов второго слоя:

$$Y_j = \sum_{i=1}^{k} W_{ij} \cdot U_{i,} \quad j = 1,2,\ldots, n.$$

2. Выбирается нейрон – победитель, т. е. нейрон с наибольшим откликом. Его номер C определяется как:

$$C = argmax Y_j \ , \ j=1,2, \ldots, n.$$

3. Корректировка весов связей *W* осуществляется по следующей формуле:

$$W_{ij}^{new} = W_{ij}^{old} + \alpha \cdot g(r,t) \cdot (U_i - W_{ij}^{old}), \quad i=1,\ldots,k; \quad j=1,\ldots n,$$

где α – константа, характеризующая обучение.

Если после очередного цикла обучения процесс изменения весов замедлился, увеличивают параметр *t*.

Обучение сетей Хопфилда

Здесь следует выделить две возможности, связанные с последующим использованием сети: будет ли она использоваться как ассоциативная память либо будет использоваться для решения оптимизационной задачи [1].

Сеть используется как ассоциативная память, а именно: для хранения *m* двоичных векторов V_s , $s=1,2,\ldots m$: $V_s=(V_1,V_2 ,...,V_m)$.

Это означает, что при предъявлении сети любого из этих векторов она должна прийти в устойчивое состояние, соответствующее этому вектору, т. е. на выходе нейронов должен выделиться этот же вектор. Если же сети будет предъявлен неизвестный ей вектор *U*, то на выходе сети должен появиться один из запомненных векторов V_i, который наиболее близок к *U*.

Очевидно, количество нейронов в такой сети должно быть равно длине хранимых векторов *n*.

Простейший способ формирования весов такой сети достигается следующей процедурой:

$$W_{ij} = \sum_{s=1}^{m} V_{is} \bullet V_{js},$$

Однако емкость такой сети, т. е. количество хранимых векторов m, невелика, $m \sim log\ n$.

Сеть используется для решения оптимизационной задачи. Такая возможность обусловлена следующим замечательным свойством сетей Хопфилда: в процессе функционирования сети величина, которую в литературе принято называть «энергией» сети Хопфилда, не возрастает. Один из вариантов «энергии» сети Хопфилда:

$$E_h = \frac{A}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} \bullet V_i \bullet V_j + \frac{B}{2} \sum_{i=1}^{n} V_i \bullet I_i,$$

где *A,B* – константы, определяемые задачей.

Задача исследования состоит в формулировке исходной оптимизационной проблемы в терминах нейросети и записи минимизируемого функционала E_h. Полученное для W_{ij} выражение дает значение весовых множителей. В результате функционирования сеть приходит в равновесное состояние, которое соответствует локальному минимуму функционала E_h. Величины возбужденности нейронов при этом соответствуют значениям аргументов, на которых достигается минимум.

ГЛАВА 6. СОЗДАНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ УПРАВЛЕНИЯ. СТРУКТУРНЫЕ СХЕМЫ С НЕЙРОРЕГУЛЯТОРОМ

6.1. Структурные схемы нейронной системы управления

Существует несколько различных схем построения адаптивных нейронных систем управления [7, 8, 9, 10], однако все они делятся на три основные группы:

- с применением обучаемой нейронной модели объекта для последующего обучения нейронного регулятора;
- с опорной моделью эталонного переходного процесса;
- инверсное использование нейронной модели в качестве регулятора.

Для выбора наиболее подходящей структуры рассмотрим схемы, относящиеся к этим группам.

Структурная схема с применением обучаемой нейронной модели объекта для последующего обучения нейронного регулятора представлена на рис. 12. Нейронная модель процесса необходима для обучения регулятора динамическим особенностям объекта или если неизвестны управляющие воздействия.

На вход модели подаются управляющие сигналы U_k и задержанные на один такт выходы объекта ΔY_{k-1} (количество тактов задержки зависит от динамических характеристик объекта). На вход нейронного регулятора подаются сигналы с выхода объекта ΔY_k и их задержанные на один такт значения ΔY_{k-1}.

Нейронная модель обучается на объекте управления, при этом ошибка сети вычисляется как разность между выходным сигналом нейросетевой модели и реальным значением выходного сигнала с объекта. Затем веса нейронной сети модели «замораживаются» и происходит обучение нейронной сети регулятора. После обучения нейронный регулятор способен генерировать управляющие воздействия в соответствии со свойствами объекта.

Данная схема построена с учетом того, что для обучения нейронных сетей используется алгоритм обратного распространения ошибки «Back-Propagation». Только с помощью этого метода можно осуществить обратное распространение ошибки через «замороженные» слои нейронной модели на выход нейронного регулятора и далее через нейронный регулятор на его вход. Это достоинство является, в то же время, и ограничением данной схемы, так как в случае плохой точности обучения невозможно использовать другие структуры сетей или другие алгоритмы обучения.

Структурная схема с опорной моделью эталонного переходного процесса представлена на рис. 13. Ошибка нейронного регулятора определяется как разность между заданным значением параметра согласно эталонной модели переходного процесса и текущим значением параметра. Далее ошибка используется для подстройки весов нейронного регулятора. На вход регулятора поступает задание, текущее и предыдущее значения параметров. Как видно на рис. 13, для обучения нейронного регулятора необходимо иметь эталонный переходный процесс, который получают, используя обычный ПИ или ПИД регулятор. Обучающая выборка регулятора представляет собой значения

эталонного переходного процесса. В течение обучения нейронной сети предъявляется один и тот же переходный процесс, пока сеть не научится его повторять, т. е. после обучения нейронный регулятор должен генерировать управляющие воздействия так, чтобы переходный процесс был близок к эталонному.

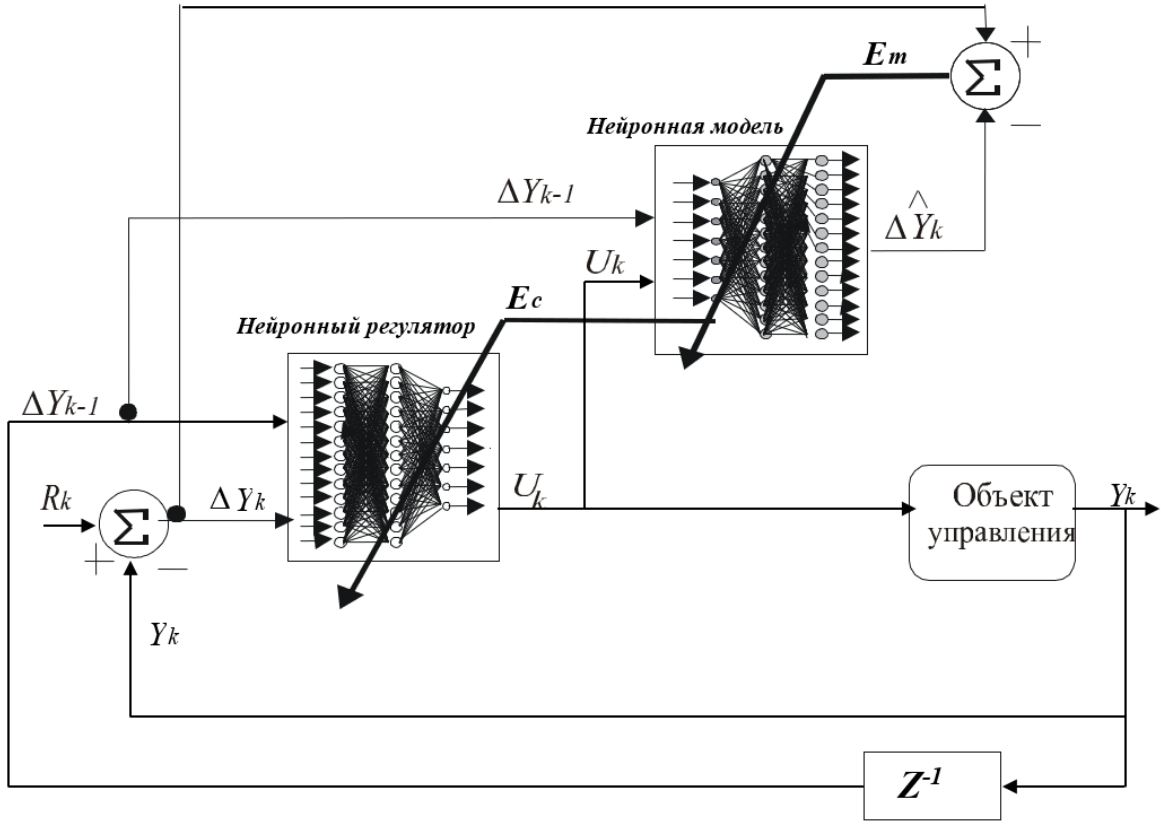


Рисунок 12 – Структурная схема с применением обучаемой нейронной модели объекта для обучения нейронного регулятора

Схемы такого рода применимы только тогда, когда хорошо известна динамика объекта, т.е. имеется модель желаемого переходного процесса. В нашем случае имеется объект с большим количеством взаимосвязанных параметров, динамика которых сложна и связана с конкретной ситуацией. Поскольку управление идет по множеству каналов одновременно, то просчитать влияние каналов друг на друга в переходных процессах весьма затруднительно.

Кроме того, в таком распределенном объекте существует далеко не один вариант управляющих воздействий, приводящих к необходимому снижению ошибки управления. Поэтому применение данной схемы затруднено и, при попытке реализации, можно ожидать значительную ошибку управления.

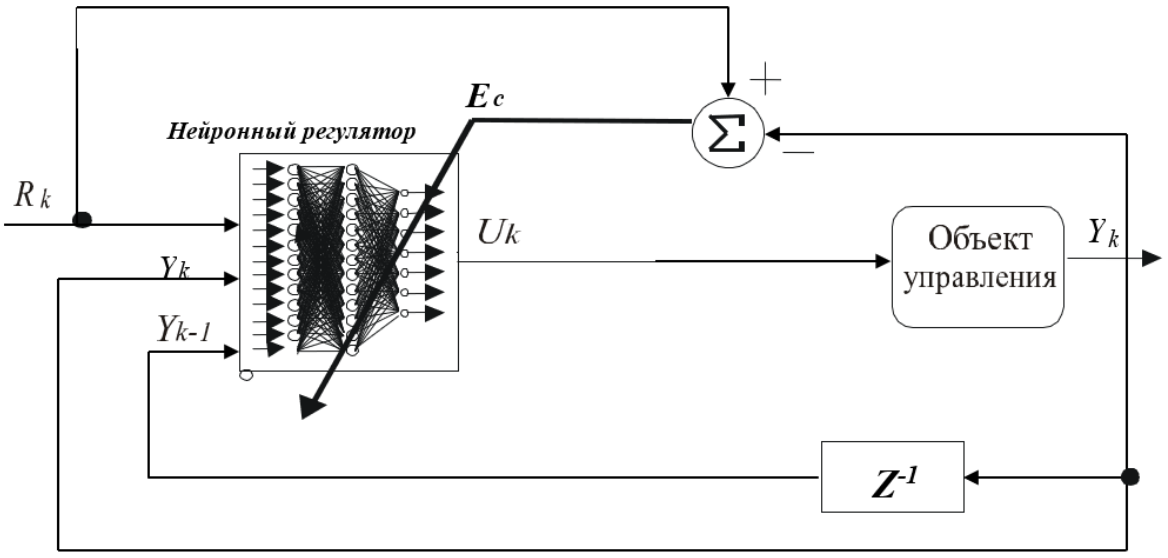


Рисунок 13 – Структурная схема с опорной моделью эталонного переходного процесса

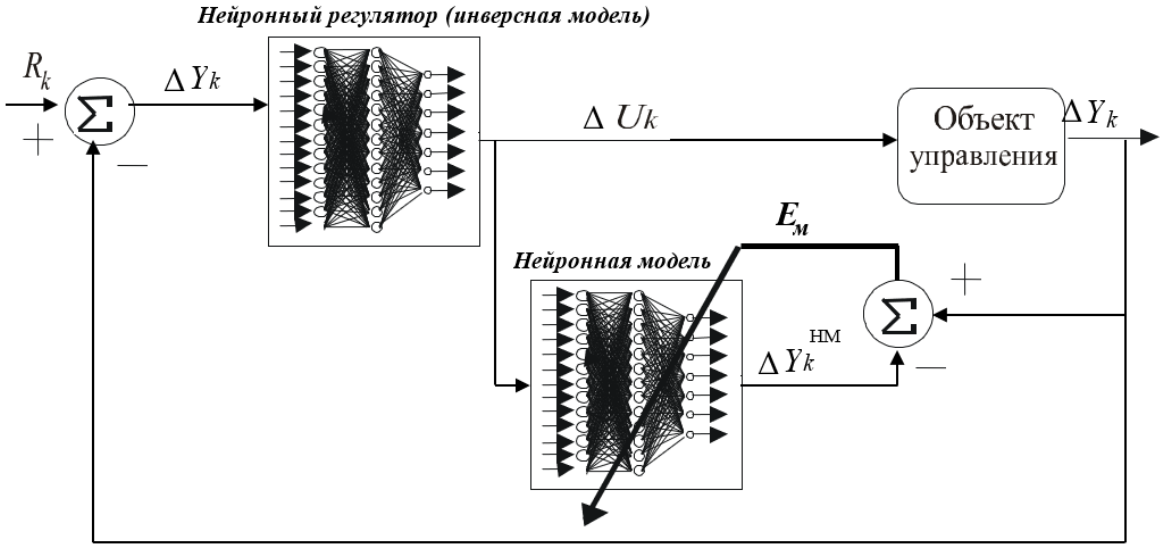


Рисунок 14 – Структурная схема инверсного использования нейронной модели в качестве регулятора

Структурная схема инверсного использования нейронной модели в качестве регулятора представлена на рис. 14. Данная схема является наиболее простой и заключается в обучении нейронной модели объекта управления с последующей инверсией ее в нейронный регулятор. Благодаря такой схеме исключается двойное обучение сначала модели, а потом – регулятора.

Основная проблема использования такой схемы заключается в необходимости обращения матрицы весов нейронной сети. Для небольших размерностей векторов входных и выходных данных обращение матрицы сложностей не вызывает. При слишком больших размерностях операция обращения матрицы дает значительную погрешность вычисления весов, что сказывается на качестве работы нейронного регулятора. Кроме того, такая схема

может использоваться только в объектах, имеющих статические характеристики и не требующих учета динамики.

Для косвенного учета динамических свойств объекта следует использовать первую схему (см. рис. 12) , а вторая схема (см. рис. 13) учитывает динамику напрямую.

6.2. Методы создания обучающих выборок. Использование априорной информации об объекте

Управление осуществляется обученным модулем нейронного регулятора с учетом ограничений на управляющие воздействия. Период переобучения нейронного регулятора зависит от того, как часто ошибка управления выходит за пределы допустимых значений. Нейронная модель процесса необходима при обучении нейронного регулятора, кроме того, она может использоваться для прогнозирования при ручном управлении технологическим процессом.

Ситуации такого рода могут возникать перед пуском нового оборудования, когда еще не сгенерирована первая обучающая выборка.

Пример нейросетевой системы управления профилем бумажного полотна показан на рис. 15.

После каждого переобучения нейронной модели ее следует проверять на адекватность объекту управления. Проверка модели осуществляется с использованием ряда тестовых примеров, соответствующих по времени примерам из обучающей выборки, но не входящих в нее.

Поскольку нейронные сети имеют предел обучаемости, т. е. существует предельное количество информации, которую сеть способна запоминать, обобщать и воспроизводить, то переобучение устаревшего модуля нейронного регулятора через некоторое время приведет к тому, что нейронная сеть, перенасыщенная информацией, перестанет адекватно воспринимать вновь поступающие данные о процессе. Этот недостаток можно преодолеть двумя путями:

1) ввести так называемый «коэффициент забывания» в алгоритм обучения нейронной сети. Этот коэффициент уменьшает синаптические веса сети на каждом такте обучения. Соответственно, те веса, которые устарели и используются редко, через определенное время станут близкими к нулю. Однако это связано с эмпирической подстройкой такого коэффициента в зависимости от того, насколько быстро меняется содержащаяся в обучающей выборке аппроксимируемая функция, что создает дополнительные трудности при настройке алгоритма обучения [1];

2) каждый раз обучать новый модуль нейронного регулятора. При этом обучающая выборка должна быть скользящей по времени, т. е. включать в себя некоторое количество данных из предыдущей выборки.

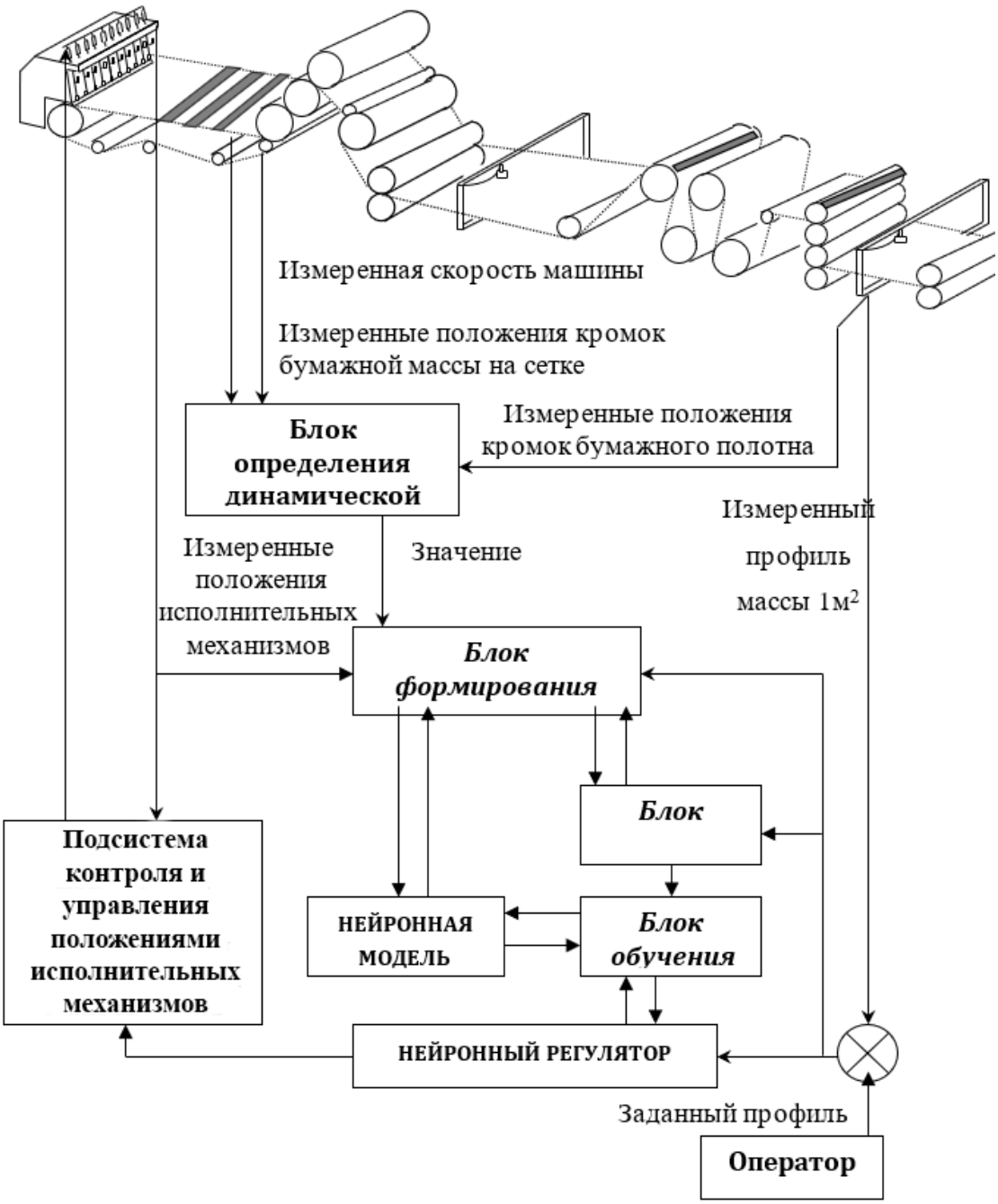


Рисунок 15 – Схема нейросетевой системы управления массой 1 м^2 бумаги

Последний способ является наиболее простым и надежным, так как при появлении нежелательных отклонений в работе нейронного регулятора, связанных со случайными процессами при обучении, они полностью исчезнут после замены модуля регулятора.

То есть новый модуль регулятора не наследует и, соответственно, не накапливает устаревшую информацию о процессе, полностью ориентируясь на информацию, содержащуюся в текущей обучающей выборке. Алгоритм адаптации представлен на рис. 16. Он содержит алгоритм формирования обучающих выборок, показанный отдельно на рис. 17.

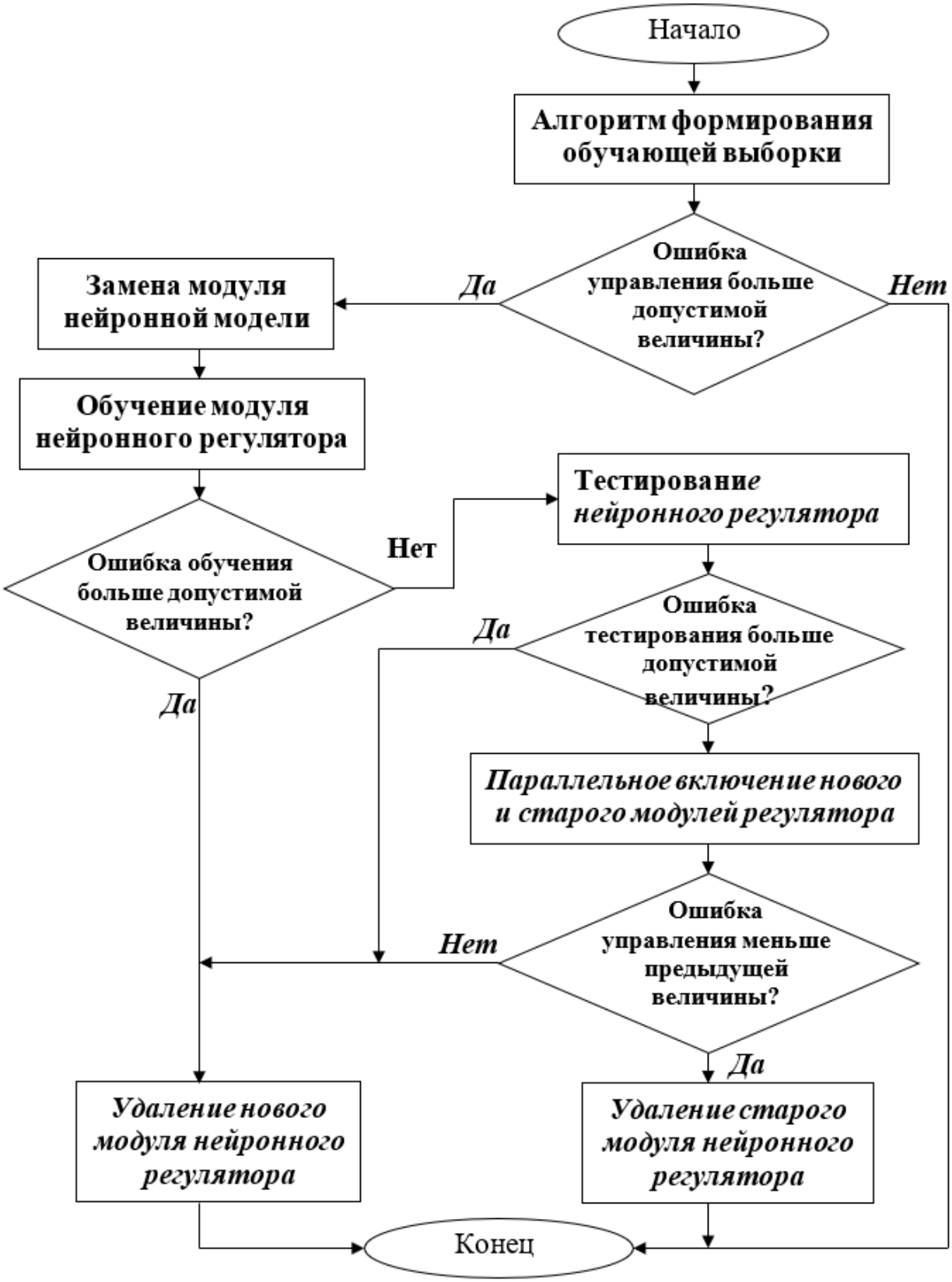


Рисунок 16 – Алгоритм функционирования адаптивной системы управления с нейронным регулятором

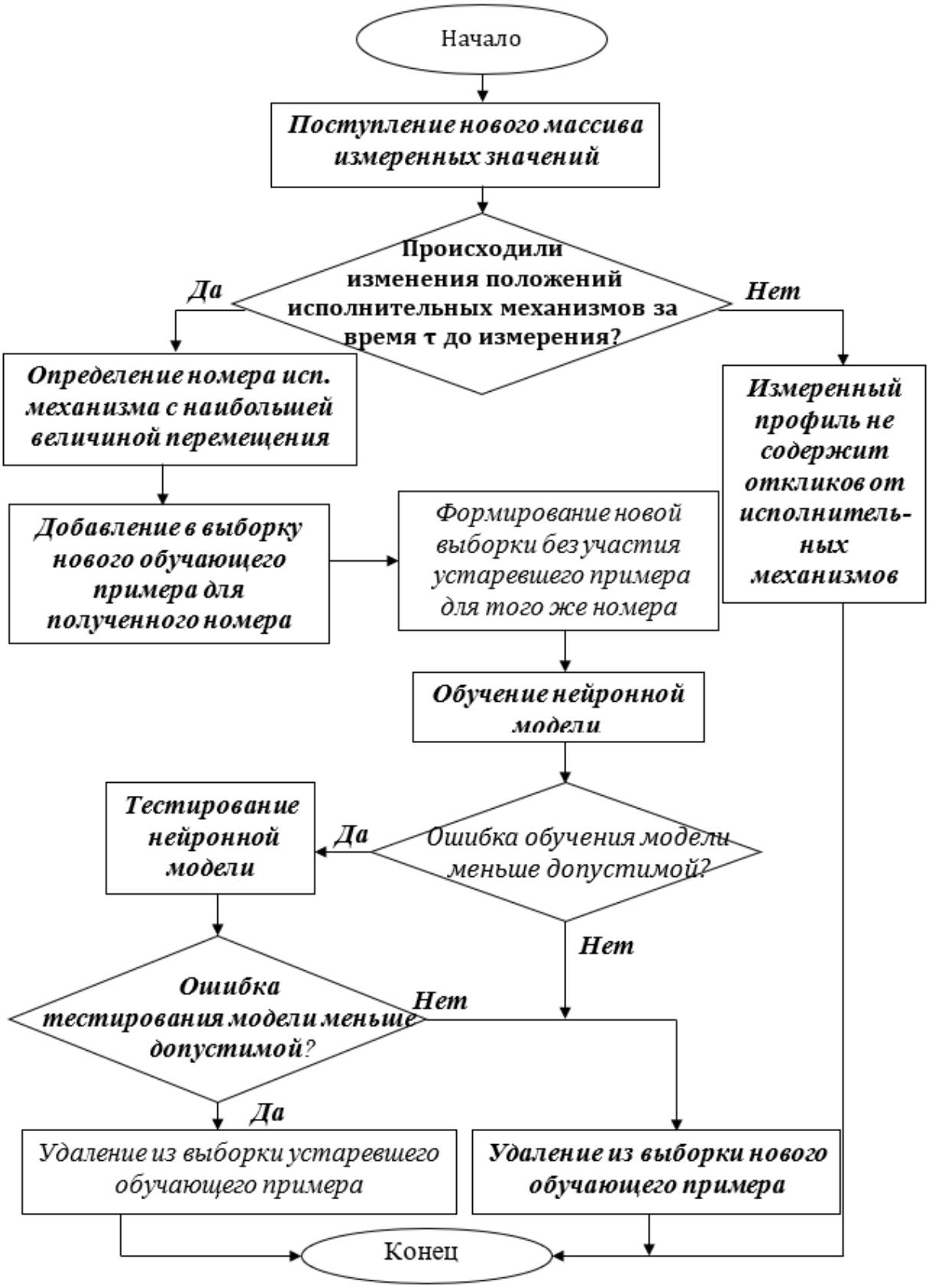


Рисунок 17 – Алгоритм формирования обучающей выборки

Алгоритм начинает функционировать, когда средняя величина ошибки управления, оцениваемая по величине 2σ, превышает заданное значение. В случае если новый модуль нейронного регулятора ухудшает качество управления, то осуществляется возврат к старому модулю. Следующее обучение нейронного регулятора инициализируется после 20-50 % обновления обучающей выборки с момента неудачного обучения регулятора.

При обновлении нейронного регулятора возникает вопрос о плавном переходе между старым и новым модулями, тем более что нейронный регулятор использует, кроме текущей ошибки управления, еще и ее предыдущие значения. Данную проблему можно решить, включив в работу новый модуль регулятора параллельно со старым и усредняя значения управляющих воздействий от обоих модулей в течение времени переключения, как это показано на рис. 18.

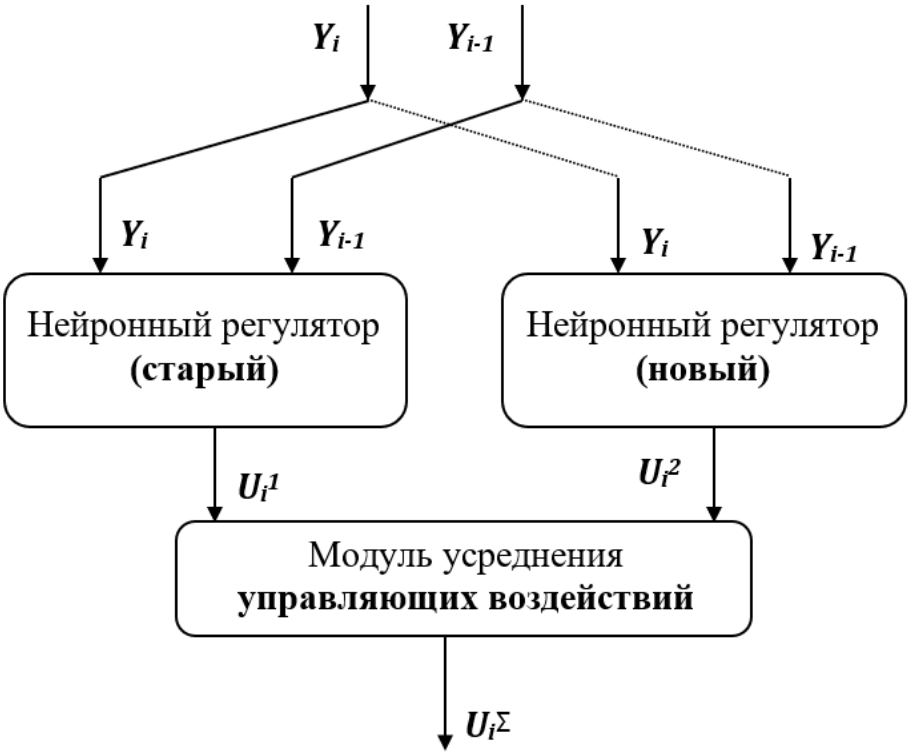


Рисунок 18 – Структурная схема для обеспечения плавного перехода при смене модуля нейронного регулятора

При больших ошибках управления и соответствующих им значительных управляющих воздействиях необходимое количество обучающих примеров накапливается значительно быстрее. При достаточном качестве управления существенных управляющих воздействий не наблюдается и, соответственно, накопление обучающих примеров происходит намного медленнее.

6.3. Методика улучшения качества переходных процессов системы управления с нейронным регулятором

Для подавления возможных колебаний системы у точки равновесия и снижения влияния шумов введена небольшая зона нечувствительности, в районе нуля сигнала рассогласования, и коррекция коэффициента усиления регулятора, пропорционально производной сигнала U_i.

С целью повышения надежности оценки производной для управляющего воздействия используется дифференцирующий полиномиальный фильтр:

$$U'_f[k] = \sum_{j=0}^{2m+1}(m-j)U[k-j] \Big/ \sum_{j=0}^{2m+1} j^2 ,$$

где $к$ – номер шага (текущее время);

m – объем выборки.

В простейшем случае, при выборке объема $(2m+1)=3$ и аппроксимирующем полиноме первой степени получим:

$$U'_f[k] = (U[k]-U[k-2]) \Big/ 2 .$$

Структура системы управления с использованием такого фильтра представлена на рис. 19. Этот фильтр уменьшает колебательность переходного процесса и приводит систему к устойчивому состоянию, как это видно на графиках переходных процессов (рис. 20) системы управления объектом чистого (транспортного) запаздывания.

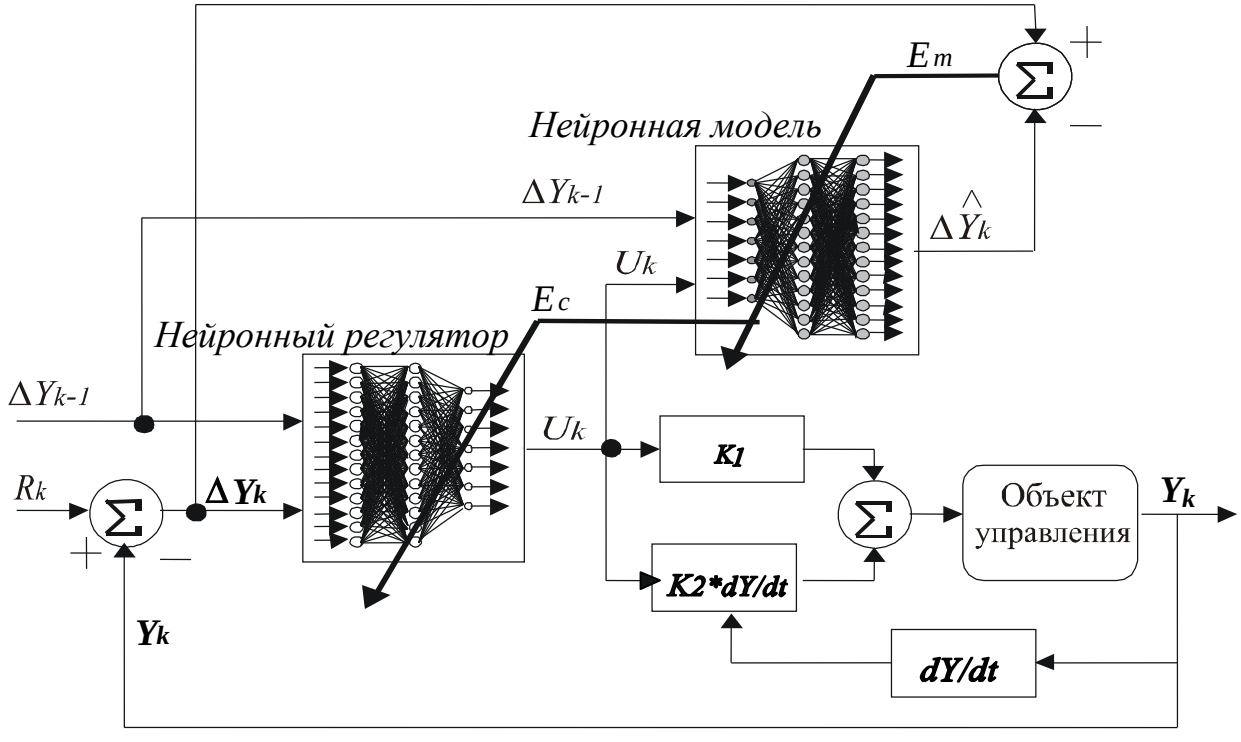
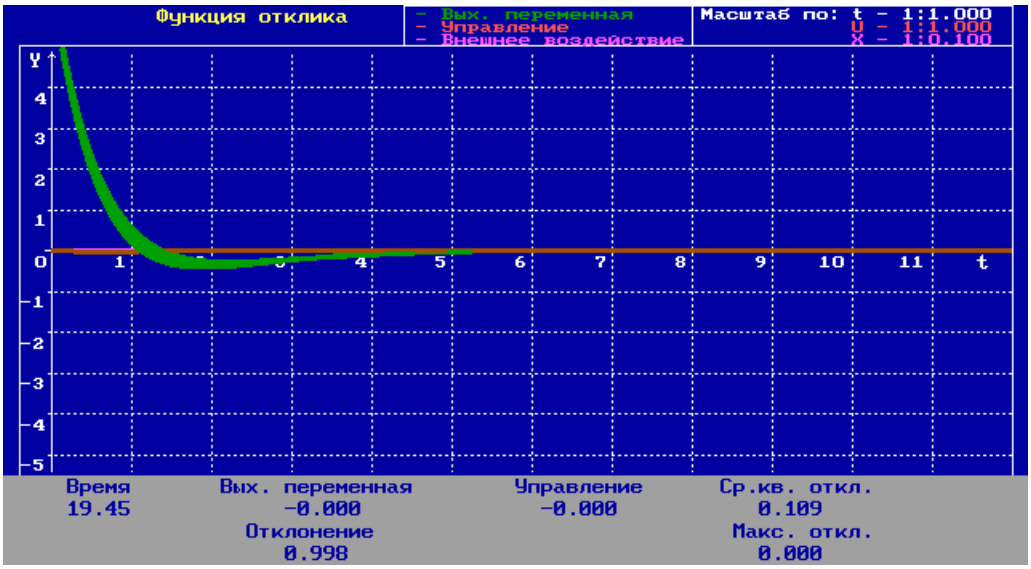


Рисунок 19 – Структурная схема системы управления с нейронным регулятором и фильтрацией сигнала управления

а)



б)

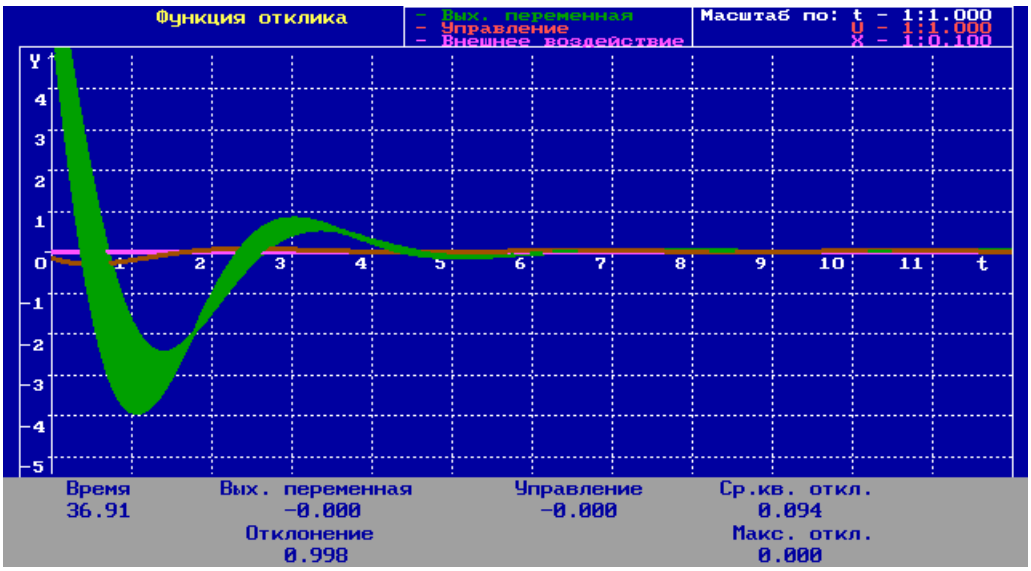


Рисунок 20 – Переходные процессы в системе управления
с нейронным регулятором:
а – с фильтрацией сигнала управления; *б* – без фильтрации

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Хайкин, С. Нейронные сети: полный курс / С. Хайкин ; пер. с англ. – М.: Вильямс, 2008. – 1103 с. – Текст: непосредственный.
2. Мешалкин, В. П. Экспертные системы в химической технологии. Основы теории, опыт разработки и применение / В. П. Мешалкин. – М.: Химия, 1995. – 366 с. – Текст: непосредственный.
3. Искусственный интеллект. В 3-х кн. Кн. 2. Модели и методы: справочник / ред. Д. А. Поспелова. – М.: Радио и связь, 1990. – 303 с. – Текст: непосредственный.
4. Змитрович, А. И. Интеллектуальные информационные системы / А. И. Зимитрович. – Минск: НТООО «ТетраСистем», 1997. – 368 с. – Текст: непосредственный.
5. Черноруцкий, И. Г. Методы оптимизации и принятия решений: учебное пособие / И. Г. Черноруцкий. – СПб.: Лань, 2001. – 381 с. – Текст: непосредственный.
6. Малышев, Н. Г. Нечеткие модели для экспертных систем в САПР / Н. Г. Малышев, Л. С. Берштейн, А. В. Боженюк. – М.: Энергоатомиздат, 1991. – 136 с. – Текст: непосредственный.
7. Levin, A. U., Narenda, K. S. Control of nonlinear dynamical systems using neural networks: controllability and stabilization. IEEE Trans., 1993, NN-4, (2), pp. 192-206.
8. Chen, F. C., Khalil, H. K. Adaptive control of nonlinear systems using neural networks. Int. J. Control, 1992, 55, (6), pp.1299-1317.
9. Yabuta, T., Yamacla, T. Neural network controller characteristic with regard to adaptive control. IEEE Trans. 1992, SMS-22, (1), pp. 170-177.
10. Kasparian, V., Batur, C. Model reference based neural network adaptive controller // ISA Transactions, № 37, 1998, pp. 21-39.

Учебное издание

Бахтин Андрей Владимирович
Ремизова Ирина Викторовна

Интеллектуальные системы управления технологическими процессами

Учебное пособие

2-е издание, стереотипное

Редактор и корректор А. Н. Чернышева
Техн. редактор Д. А. Романова Темплан 2024 г., поз. 5062/24

Подписано к печати 21.03.2024. Формат 60x84/16. Бумага тип № 1.
Печать офсетная. Печ. л. 3,0. Уч.-изд. л. 3,0.
Тираж 30 экз. Изд. № 5062/24. Цена «С». Заказ №

Ризограф Высшей школы технологии и энергетики СПбГУПТД,
198095, Санкт-Петербург, ул. Ивана Черных, 4.