3. Разработка «Программное обеспечение для автоматической авторизации пользователей ОС Unix»

3.1. Процесс разработки и методология разработки ПО

При обсуждении технологии программирования широко используется понятие жизненного цикла программы. Жизненный цикл программы — это некая абстрактная модель, однако элементы жизненного цикла являются тем материалом, из которого строятся различные конкретные модели технологии программирования.

Жизненный цикл программы — это совокупность и последовательность изменений формы программы за всё время ее существования.

На рисунке 3.1 текст внутри фигур обозначает деятельность, проводимую в данном состоянии, а текст над стрелкой — условие или событие, управляющее данным переходом.

Для упорядочения циклической структуры используется понятие выпуска. Выпуск (release) — характерная точка в жизни программы, отмечающая прохождение одного польного цикла.

Следует отметить три существенных момента этой схемы:

- подразумевается одна долгоживущая программа;
- жизненный цикл имеет видимое начало;
- жизненный цикл потенциально бесконечен.

При описании процессов, связанных с разработкой программного обеспечения, основными структурными составляющими являются фазы и витки. Фаза (phase) — часть процесса разработки. Обычно каждая фаза

характеризуется вехой, достижение которой знаменует завершение фазы.

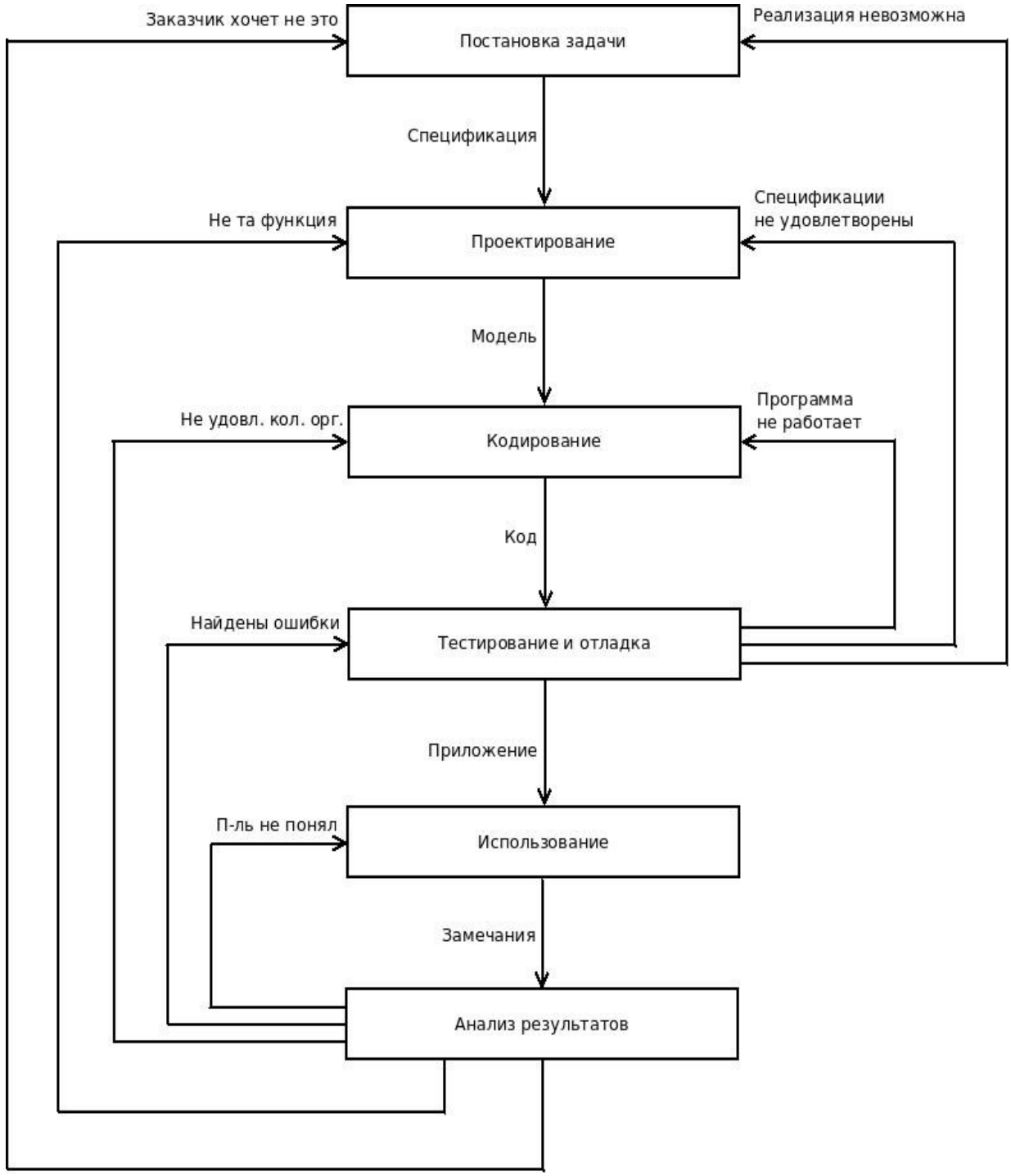

Рисунок 3.1 - Жизненный цикл программы

Например, «Требования определены» - это веха, достижение которой знаменует завершение фазы определения и анализа требований. Когда эта веха достигнута, процесс переходит в другую фазу, обычно в фазу архитектурного проектирования.

Наборы фаз, которые включают в модель процесса разработки, различны в разных технологиях программирования. Чаще всего встречаются следующие фазы:

- Извлечение и анализ требований;
- Архитектурное и детальное проектирование;
- Реализация и кодирование;
- Тестирование и верификация;
- Сопровождение и продолжающаяся разработка.

Разработка требований — это первая из основных фаз процесса создания программных систем. Включает в себя (рисунок 3.2):

- Анализ предметной области. Позволяет выделить сущности предметной области, определить первоначальные требования к функциональности и определить границы проекта;
- Анализ осуществимости. Должен выполняться для новых программных систем. На основании анализа предметной области, общего описания системы и ее назначения принимается решение о продолжении или завершении проекта;
- Формирование и анализ требований. Взаимодействуя с пользователями, обсуждая и анализируя с ними задачи, возлагаемые на систему, разрабатывая модели и прототипы, разработчики формулируют пользовательские требования;
- Документирование требований. Сформированные на предыдущем этапе пользовательские требования должны быть документированы. При этом нужно учесть, что основными читателями этого документа будут пользователи, поэтому основными требованиями к нему будут ясность и понятность;

- Детализация требований. Разработчики детализируют требования пользователей, формируя более точные подробные системные требования;
- Согласование и утверждение требований. На этом этапе пользовательские и системные требования должны быть оформлены в виде единого документа, содержащего все функциональные и нефункциональные требования. Такой документ, обычно, называется спецификацией требований. Спецификация требований должна быть однозначной, завершенной и согласованной.

Архитектура программного обеспечения — это представление системы программного обеспечения, дающее информацию о компонентах, составляющих систему, о взаимосвязях между этими компонентами и правилах, регламентирующих эти взаимосвязи. Она способствует эффективной разработке проекта такой системы.

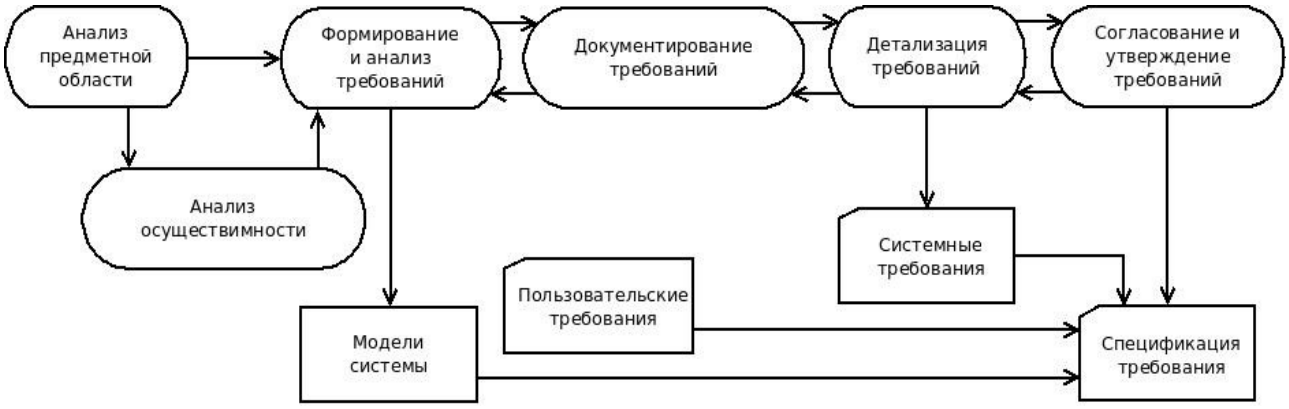

Рисунок 3.2 - Разработка требований

Здесь можно выделить:

- Потоки данных (data flows). Независимые процессы обработки данных запускаются, когда им на вход поступают данные;
- Независимые компоненты. Независимые компоненты взаимодействуют, обмениваясь сообщениями;
- Виртуальные машины. Определяется предметно-ориентированный

внутренний язык и процессор этого языка.

Фаза детального проектирования обычно предусматривает применение большого числа различных средств, инструментов и приемов. Как правило, здесь наблюдается разнообразие больше, чем в фазе кодирования и реализации. При кодировании и реализации возможностей для выбора и принятия решений не так много — все они предопределены выбранной системой программирования. Фаза детального проектирования, напротив, требует постоянного выбора среди множества возможных альтернатив. Разрабатываемые требования и выбранная архитектура полностью предписывают, что нужно сделать, но в очень малой степени являются подсказкой при поиске ответа на вопрос как это можно сделать.

Инкрементная модель разработки (англ. incremental model) — модель процесса разработки, в которых процесс делится на витки, или итерации, каждая из которых в свою очередь делится на фазы (например, фазы анализа, планирования, разработки и стабилизации). Каждая фаза кончается вехой (концепция, спецификации, код, выпуск). После выпуска раскручивается очередной виток спирали.

Таким образом, на каждой итерации происходит выпуск продукта. Последовательные выпуски отличаются некоторым приращением — реализованными функциями или изменениями других свойств. Такое приращение иногда называют инкрементом, что и дало название этой группе моделей.

Наиболее известной инкрементной моделью является Унифицированный процесс (Rational Unified Process) показанной на рисунке 3.3.

3.2. Описание используемых методов

Существующие алгоритмы обнаружения лиц можно разбить на две широкие категории. К первой категории относятся методы, отталкивающиеся от опыта человека в распознавании лиц и делающие попытку формализовать и

алгоритмизировать этот опыт, построив на его основе автоматическую систему распознавания. Вторая категория опирается на инструментарий распознавания образов, рассматривая задачу обнаружения лица, как частный случай задачи распознавания.

Рисунок 3.3 - Унифицированный процесс

Человеческий мозг справляется с задачей обнаружения лиц на изображениях более чем успешно. Естественно было бы попробовать определить и использовать принципы, которыми руководствуется мозг при решении задачи распознавания. Среди методов, делающих такую попытку, можно выделить два направления: методы распознавания "сверху-вниз" основанные на знаниях и методы распознавания "снизу-вверх" основанные на особенностях.

Распознавание "сверху-вниз" означает построение некоторого набора правил, которым должен отвечать фрагмент изображения, для того чтобы быть признанным человеческим лицом. Этот набор правил является попыткой формализовать эмпирические знания о том, как именно выглядит лицо на изображениях и чем руководствуется человек при принятии решения лицо он видит или нет. Довольно легко построить набор простых и очевидных (как

кажется) свойств изображения лица, например: лицо обычно симметрично, черты лица (глаза, носа, рот) отличаются от кожи по яркости (обычно им также соответствуют области резкого изменения яркости), черты лица расположены вполне определенным образом. Опираясь на перечисленные свойства, можно построить алгоритм проверяющий их наличие на фрагменте изображения. К этому же семейству методик можно также отнести распознавание с помощью шаблонов, заданных разработчиком (predefined template matching). Шаблоны задают некий стандартный образ изображения лица, например, путем описания свойств отдельных областей лица и их возможного взаимного расположения. Обнаружение лица с помощью шаблона заключается в проверке каждой из областей изображения на соответствие заданному шаблону.

Принципы шаблонов и другие методы распознавания "сверху-вниз" использовались, в основном, в ранних работах по обнаружению лица [9], [10], [11], [12], [13]. Это были первые попытки формализации признаков изображений лица, к тому же вычислительные мощности компьютеров в те годы не позволяли эффективно использовать более сложные методы распознавания изображений. Несмотря на некоторую наивность алгоритмов, не стоит недооценивать значение этих работ, поскольку многие методики, успешно применяемые в настоящее время, были разработаны или адаптированы к данной конкретной проблеме именно в них.

Распознавание "снизу-вверх" использует инвариантные свойства (invariant features) изображений лиц, опираясь на предположение, что раз человек может без усилий распознать лицо на изображении независимо от его ориентации, условий освещения и индивидуальных особенностей, то должны существовать некоторые признаки присутствия лиц на изображений, инвариантные относительно условий съемки. Алгоритм работы методов распознавания "снизу-вверх" может быть кратко описан следующим образом:

1) Обнаружение элементов и особенностей (features), которые характерны для изображения лица;

2) Анализ обнаруженных особенностей, вынесение решения о количестве и расположении лиц.

Второе семейство методов подходит проблеме с другой стороны, и, не пытаясь в явном виде формализовать процессы, происходящие в человеческом мозге, стараются выявить закономерности и свойства изображения лица неявно, применяя методы математической статистики и машинного обучения. Методы этой категории опираются на инструментарий распознавания образов, рассматривая задачу обнаружения лица, как частный случай задачи распознавания. Изображению (или его фрагменту) ставится в соответствие некоторым образом вычисленный вектор признаков, который используется для классификации изображений на два класса - лицо/не лицо. Самый распространенный способ получения вектора признаков это использование самого изображения: каждый пиксель становится компонентом вектора, превращая черно-белое изображение $n \times m$ в вектор пространства $R^{n \times m}$. Недостатком такого представления является чрезвычайно высокая размерность пространства признаков. Достоинство заключается том, что используя все изображение целиком вместо вычисленных на его основе характеристик, из всей процедуры построения классификатора (включая выделение устойчивых признаков для распознавания) полностью исключается участие человека, что потенциально снижает вероятность ошибки построения неправильной модели изображения лица вследствие неверных решений и заблуждений разработчика.

Обычно поиск лиц на изображениях с помощью методов, основанных на построении математической модели изображения лица, заключается в полном переборе всех прямоугольных фрагментов изображения всевозможных размеров и проведения проверки каждого из фрагментов на наличие лица. Поскольку схема полного перебора обладает такими безусловными недостатками, как избыточность и большая вычислительная сложность, авторами применяются различные методы сокращения количества рассматриваемых фрагментов.

К методам моделирования изображения лица относятся:

- Моделирование класса изображений лиц с помощью Метода Главных Компонент (Principal Components Analysis, PCA);
- Моделирование класса изображений лиц с помощью Факторного анализа (Factor Analysis, FA);
- Моделирование распределения векторов лиц с помощью смеси многомерных нормальных распределений (mixture of Gaussians);
- Линейный Дискриминантный Анализ (Linear Discriminant Analysis,LDA);
- Метод Опорных Векторов (Support Vector Machines, SVM);
- Искусственные Нейронные Сети (Neural Networks, NN);
- Sparse Network of Winnows (SnoW);
- Скрытые Марковские Модели (Hidden Markov Models, HMM);
- Active Appearance Models (AAM).

Основа методов первой категории - эмпирика, является одновременно их сильной и слабой стороной. Большая изменчивость объекта распознавания, зависимость вида лица на изображении от условий съемки и освещения позволяют без колебаний отнести обнаружение лица на изображении к задачам высокой сложности. Применение эмпирических правил позволяет построить некоторую модель изображения лица и свести задачу к выполнению некоторого количества относительно простых проверок. Однако, несмотря на безусловно разумную посылку - попытаться использовать и повторить уже успешно функционирующий инструмент распознавания - человеческое зрение, методы первой категории пока далеки по эффективности от своего прообраза, поскольку исследователи, решившие избрать этот путь, сталкиваются с рядом серьезных трудностей. Во-первых, процессы, происходящие в мозгу во время решения задачи распознавания изображений изучены далеко не полностью, и тот набор эмпирических знаний о человеческом лице, которые доступны

исследователям на "сознательном уровне", далеко не исчерпывает инструментарий, используемый мозгом "подсознательно". Во-вторых, трудно эффективно перевести неформальный человеческий опыт и знания в набор формальных правил, поскольку чересчур жесткие рамки правил приведут к тому, что в ряде случаев лица не будут обнаружены, и напротив, слишком общие правила приведут к большому количеству случаев ложного обнаружения.

Можно перечислить следующие проблемы, общие для методов второй категории:

1. Зависимость от ориентации и масштаба лица. Большинство классификаторов не являются инвариантными к повороту лица в плоскости изображения и изменению его размера. Поэтому для успешного обнаружения лица, отличного по размеру или ориентации от лиц в тренировочном наборе, требуется дополнительная обработка входного изображения (масштабирование, поворот). Проблему изменения масштаба решают, обычно, путем полного перебора всех возможных прямоугольных фрагментов изображения всех возможных размеров. Попытка же рассматривать еще и все возможные углы поворота лиц в плоскости изображения приведет к тому, что время выполнения и без того долгой процедуры перебора фрагментов превысит все мыслимые пределы. Если говорить о повороте головы вне плоскости изображения, то это является проблемой для многих методов из обоих категорий, поскольку при значительном повороте лицо на изображении изменяется настолько сильно, что многие признаки и правила (заданные разработчиком или полученные неявно) распознавания фронтального изображения лица становятся совершенно непригодными;

2. Неявный способ определения признаков для распознавания лица таит в себе потенциальную опасность: классификатор, обладающий недостаточно репрезентативным набором изображений лиц, теоретически

может выделить вторичные или ложные признаки в качестве важных. Одно из следствий - потенциальная зависимость от освещения, которое преобладало в тренировочном наборе. В ряде случаев [14] применяется дополнительная предобработка изображения для компенсации влияния освещения;

3. Высокая вычислительная сложность. Во-первых, сами классификаторы часто включают в себя большое количество достаточно сложных вычислений; во-вторых, полный перебор всех возможных прямоугольных фрагментов изображения сам по себе занимает большое количество времени. Это затрудняет использование некоторых методов в системах реального времени (например - отслеживании перемещения лица в видеопотоке).

Сравнивать между собой качество распознавания методов разных категорий достаточно тяжело, поскольку в большинстве случаев, опираться можно только на данные испытаний, предоставляемые самими авторами, поскольку провести крупномасштабное исследование по реализации большинства известных методов и сравнения их между собой на едином наборе изображений не представляется возможным по причине невообразимой трудоемкости этой задачи.

На основе информации, предоставляемой авторами методов, также сложно провести корректное сравнение, поскольку проверка методов часто производится на разных наборах изображений, с разной формулировкой условий успешного и неуспешного обнаружения. К тому же проверка для многих методов первой категории производилась на значительно меньших наборах изображений.

Заметное различие между первой и второй категорией описанных методов заключается еще и в том, что эмпирические методы часто довольно просты в реализации (особенно относительно методов второй категории), и предоставляют возможность гибкой настройки под конкретную задачу путем

модификации интуитивно понятных параметров. Методы, опирающиеся на инструментарий распознавания образов, требуют значительных усилий по формированию тренировочных наборов изображений и обучению классификатора. Влияние параметров, контролирующих классификатор, на его поведение часто далеко не очевидно. Однако трудоемкость создания работающих прототипов методов второй категории частично компенсируется высокими заявленными показателями качества распознавания на больших коллекциях изображений.

Что касается рекомендаций по выбору метода для решения задачи обнаружения лиц, то можно сказать, что выбирать подходящий метод, исходя из цифровых показателей качества распознавания вряд ли целесообразно. Скорее, все зависит от конкретной задачи и условий в которых должен функционировать разрабатываемый алгоритм. Построение универсального метода, обеспечивающего высокий уровень распознавания при отсутствии ограничений на исходные изображения в настоящее время не представляется возможным, однако для большинства конкретных задач можно создать методы, предоставляющие достаточный уровень распознавания.

В качестве условий, влияющих на выбор метода решения задачи, можно перечислить следующие:

- Предполагаемое разнообразие лиц: ограниченный набор людей, ограничения на возможный тип лица (раса, присутствие растительности на лице, очков и т.д.), отсутствие ограничений;
- Ориентация лиц на изображении: строго вертикальная (или наклон под известным углом), в определенных границах вблизи известного угла наклона, любая;
- Цветное или черно-белое изображение;
- Масштаб лиц, разрешение и качество изображения (зашумленность, степень сжатия);

- Предполагаемое количество лиц, присутствующих на изображении: известно, примерно известно, неизвестно;
- Условия освещения: фиксированные известные, приблизительно известные, любые;
- Фон: фиксированный, контрастный однотонный, слабоконтрастный зашумленный, неизвестный;
- Что важнее - не пропустить ни одного лица или минимизировать количество случаев ложного обнаружения.

3.2.1. Искусственные нейронные сети

Нейросети давно и успешно применяются для решения многих задач распознавания. Для решения задачи обнаружения лица применялось большое количество нейронных сетей различных архитектур[14], в частности: многослойные персептроны[15], probabilistic decision-based neural networks (PDBNN) [16], и т.д. Достоинством использования нейросетей для решения задачи обнаружения лица является возможность получения классификатора, хорошо моделирующего сложную функцию распределения изображений лиц $p(x \mid face)$. Недостатком же является необходимость в тщательной и кропотливой настройке нейросети для получения удовлетворительного результата классификации.

Искусственные нейронные сети индуцированы биологией, так как они состоят из элементов, функциональные возможности которых аналогичны большинству элементарных функций биологического нейрона. Эти элементы затем организуются по способу, который может соответствовать (или не соответствовать) анатомии мозга. Несмотря на такое поверхностное сходство, искусственные нейронные сети демонстрируют удивительное число свойств присущих мозгу. Например, они обучаются на основе опыта, обобщают предыдущие прецеденты на новые случаи и извлекают существенные свойства из поступающей информации, содержащей излишние данные.

Несмотря на такое функциональное сходство, даже самый оптимистичный их защитник не предположит, что в скором будущем искусственные нейронные сети будут дублировать функции человеческого мозга. Реальный «интеллект», демонстрируемый самыми сложными нейронными сетями, находится ниже уровня дождевого червя, и энтузиазм должен быть умерен в соответствии с современными реалиями. Однако равным образом было бы неверным игнорировать удивительное сходство в функционировании некоторых нейронных сетей с человеческим мозгом. Эти возможности, как бы они ни были ограничены сегодня, наводят на мысль, что глубокое проникновение в человеческий интеллект, а также множество революционных приложений, могут быть не за горами.

Нервная система человека, построенная из элементов, называемых нейронами, имеет ошеломляющую сложность. Около 10^{11} нейронов участвуют в примерно 10^{15} передающих связях, имеющих длину метр и более. Каждый нейрон обладает многими качествами, общими с другими элементами тела, но его уникальной способностью является прием, обработка и передача электрохимических сигналов по нервным путям, которые образуют коммуникационную систему мозга.

На рисунке 3.4 показана структура пары типичных биологических нейронов. Дендриты идут от тела нервной клетки к другим нейронам, где они принимают сигналы в точках соединения, называемых синапсами. Принятые синапсом входные сигналы подводятся к телу нейрона. Здесь они суммируются, причем одни входы стремятся возбудить нейрон, другие – воспрепятствовать его возбуждению. Когда суммарное возбуждение в теле нейрона превышает некоторый порог, нейрон возбуждается, посылая по аксону сигнал другим нейронам. У этой основной функциональной схемы много усложнений и исключений, тем не менее большинство искусственных нейронных сетей моделируют лишь эти простые свойства.

Искусственный нейрон имитирует в первом приближении свойства

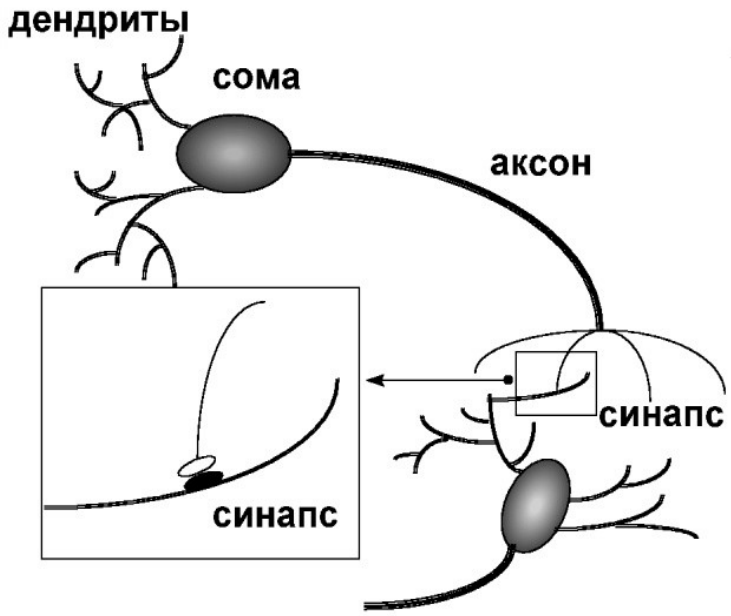

Рисунок 3.4 - Биологический нейрон

биологического нейрона. На вход искусственного нейрона поступает некоторое множество сигналов, каждый из которых является выходом другого нейрона. Каждый вход умножается на соответствующий вес, аналогичный синаптической силе, и все произведения суммируются, определяя уровень активации нейрона. На рисунке 3.5 представлена модель, реализующая эту идею. Хотя сетевые парадигмы весьма разнообразны, в основе почти всех их лежит эта конфигурация. Здесь множество входных сигналов, обозначенных x_1, x_2,…, x_n, поступает на искусственный нейрон. Эти входные сигналы, в совокупности обозначаемые вектором X, соответствуют сигналам, приходящим в синапсы биологического нейрона. Каждый сигнал умножается на соответствующий вес w_1, w_2,…, w_n, и поступает на суммирующий блок, обозначенный Σ. Каждый вес соответствует «силе» одной биологической синаптической связи. (Множество весов в совокупности обозначается вектором W.) Суммирующий блок, соответствующий телу биологического элемента, складывает взвешенные входы алгебраически, создавая выход, который мы будем называть NET. В векторных обозначениях это может быть компактно записано следующим образом:

NET = XW.

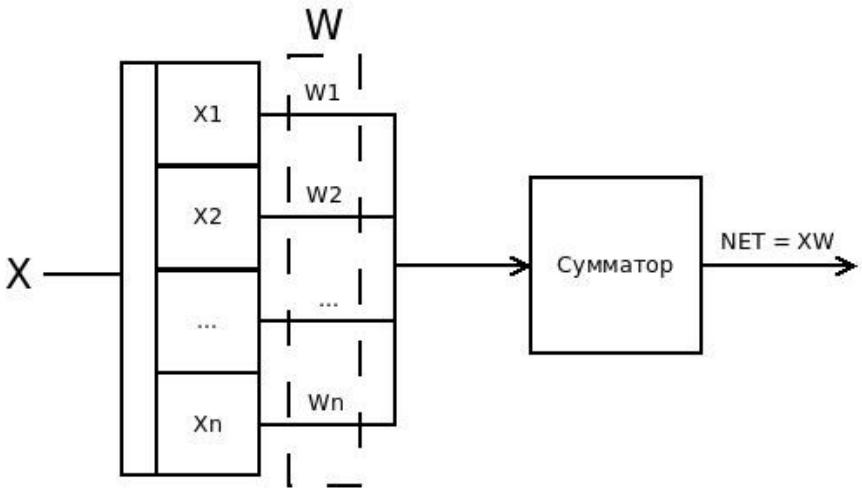

Рисунок 3.5 - Искусственный нейрон

Сигнал NET далее, как правило, преобразуется активационной функцией *F* и дает выходной нейронный сигнал OUT.

На рисунке 3.6 блок, обозначенный *F,* принимает сигнал NET и выдает сигнал OUT. Если блок *F* сужает диапазон изменения величины NET так, что при любых значениях NET значения OUT принадлежат некоторому конечному интервалу, то *F* называется «сжимающей» функцией.

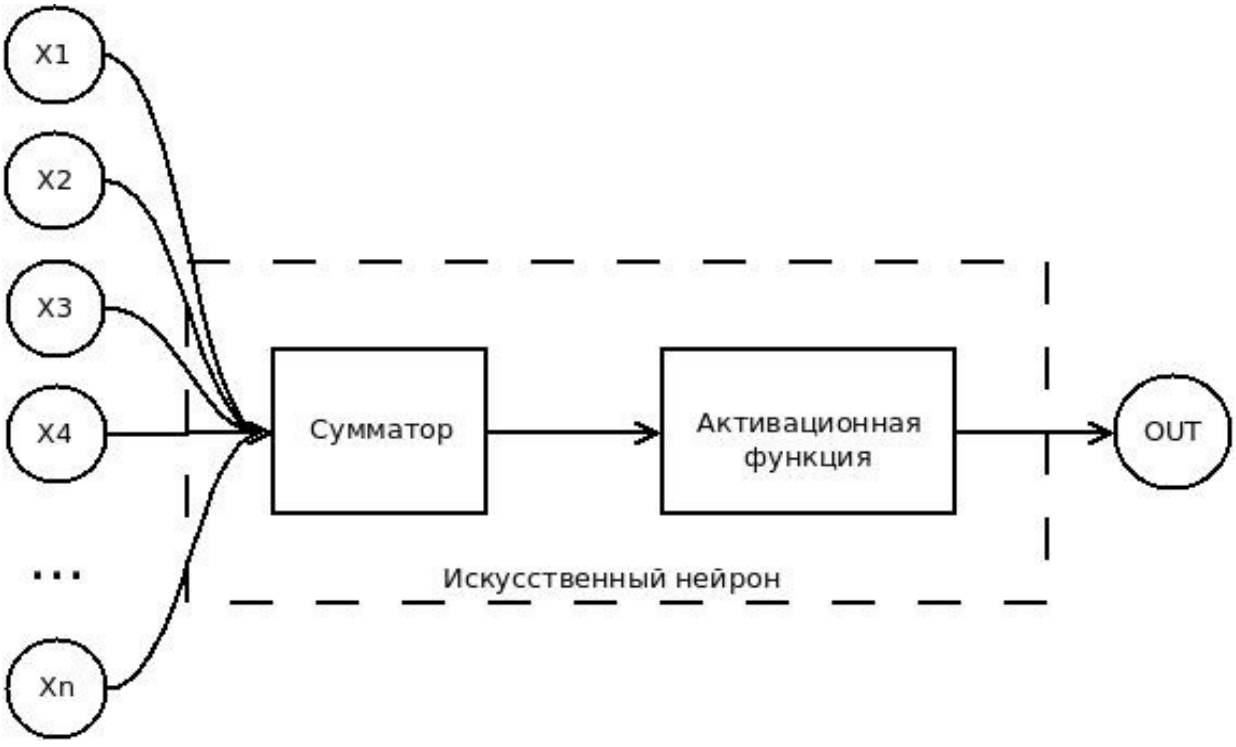

Рисунок 3.6 - Искусственный нейрон с активационной функцией

Широко используемой активационной функцией является гиперболический тангенс. По форме она сходна с логистической функцией и часто используется биологами в качестве математической модели активации нервной клетки. В качестве активационной функции искусственной нейронной сети она записывается следующим образом:

$$OUT = th(x).$$

Подобно логистической функции гиперболический тангенс является S-образной функцией, но он симметричен относительно начала координат, и в точке NET = 0 значение выходного сигнала OUT равно нулю (рисунок 3.7). В отличие от логистической функции гиперболический тангенс принимает значения различных знаков, что оказывается выгодным для ряда сетей.

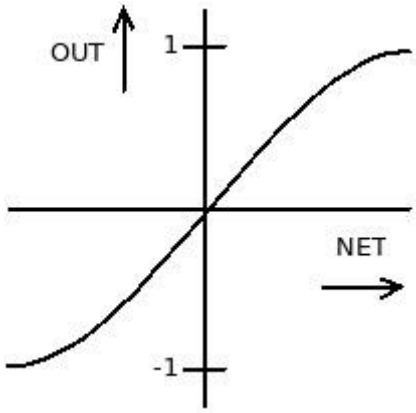

Рисунок 3.7 - Функция гиперболического тангенса

Рассмотренная простая модель искусственного нейрона игнорирует многие свойства своего биологического двойника. Например, она не принимает во внимание задержки во времени, которые воздействуют на динамику системы. Входные сигналы сразу же порождают выходной сигнал. И, что более важно, она не учитывает воздействий функции частотной модуляции или синхронизирующей функции биологического нейрона, которые ряд исследователей считают решающими.

Несмотря на эти ограничения, сети, построенные из этих нейронов, обнаруживают свойства, сильно напоминающие биологическую систему.

Только время и исследования смогут ответить на вопрос, являются ли подобные совпадения случайными или следствием того, что в модели верно схвачены важнейшие черты биологического нейрона.

Более крупные и сложные нейронные сети обладают, как правило, и большими вычислительными возможностями. Хотя созданы сети всех конфигураций, какие только можно себе представить, послойная организация нейронов копирует слоистые структуры определенных отделов мозга. Оказалось, что такие многослойные сети обладают большими возможностями, чем однослойные, и в последние годы были разработаны алгоритмы для их обучения.

Многослойные сети могут образовываться каскадами слоев. Выход одного слоя является входом для последующего слоя. Подобная сеть показана на рисунке 3.8 и снова изображена со всеми соединениями.

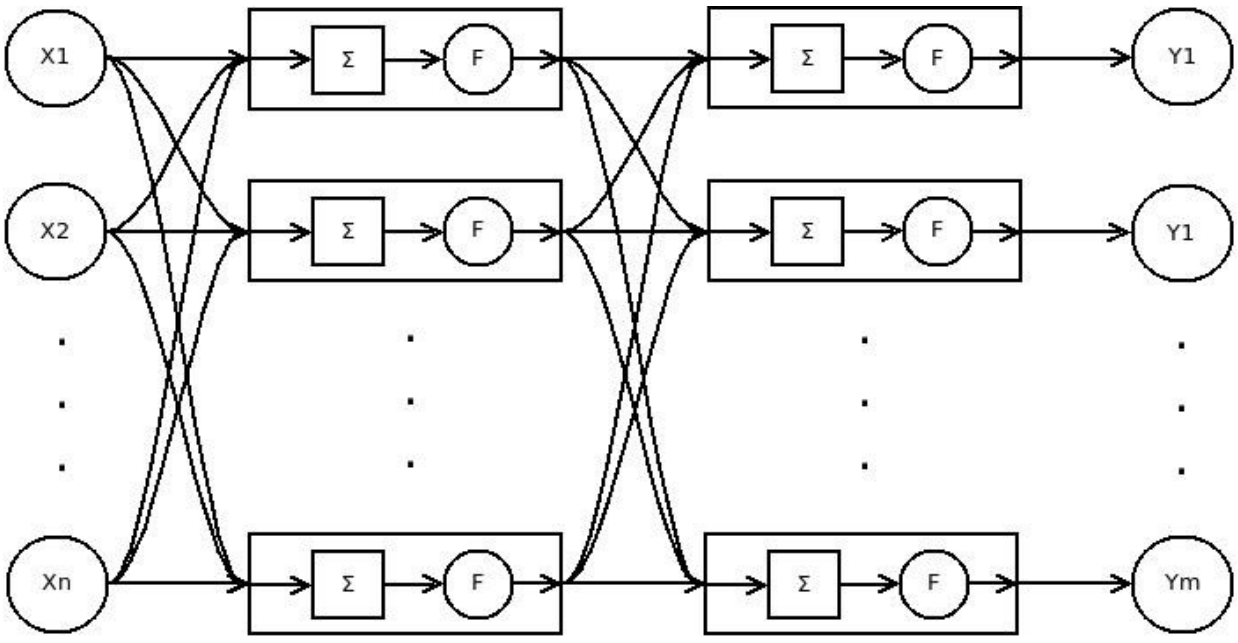

Рисунок 3.8 - Двухслойная нейронная сеть

3.2.2. Сверточные нейронные сети

Нейронные сети успешно применяют в решении многих проблем распознавания образов [17–19]: распознавание символов, распознавание объектов, и многих других. Проблема обнаружения образа лица очень трудна

из-за большого разнообразия искажений, таких как различное выражение лица, условия съемки и т. д. Преимущество использования нейронных сетей для обнаружения лица – обучаемость системы для выделения ключевых характеристик лица из обучающих выборок.

В настоящее время наиболее часто в задачах распознавания и идентификации изображений используют классические нейросетевые архитектуры (многослойный персептрон, сети с радиально-базисной функцией и др.), но, как показывает анализ данных работ, применение классических нейросетевых архитектур к данной задаче является неэффективным по следующим причинам:

- к данной задаче обычно применяется ансамбль нейронных сетей (2–3 нейронные сети, обученные с различными начальными значениями синаптических коэффициентов и порядком предъявления образов), что отрицательно сказывается на вычислительной сложности решения задачи и соответственно на времени выполнения;
- как правило, классические нейросетевые архитектуры используются в совокупности с вспомогательными методами выделения сюжетной части изображения (сегментация по цвету кожи, выделение контуров и т. д.), которые требуют качественной и кропотливой предобработки обучающих и рабочих данных, что не является эффективным;
- нейросетевые архитектуры являются крайне чувствительными к влиянию различных внешних факторов (изменения условий съемки, присутствие индивидуальных особенностей на изображении, изменение ориентации);

Дополнительно возникают трудности применения традиционных нейронных сетей к реальным задачам распознавания и классификации изображений.

Во-первых, как правило, изображения имеют большую размерность, соответственно вырастает размер нейронной сети (количество нейронов и т. п.).

Большое количество параметров увеличивает вместимость системы и соответственно требует большей обучающей выборки, что увеличивает время и вычислительную сложность процесса обучения.

Во-вторых, недостаток полносвязной архитектуры – то, что топология ввода полностью игнорируется. Входные переменные могут быть представлены в любом порядке, не затрагивая цель обучения. Напротив, изображения имеют строгую 2-мерную местную структуру: переменные (пиксели), которые являются пространственно соседними, чрезвычайно зависимы.

От данных недостатков свободны так называемые свёрточные нейронные сети. Свёрточные нейронные сети обеспечивают частичную устойчивость к изменениям масштаба, смещениям, поворотам, смене ракурса и другим искажениям. Свёрточные нейронные сети объединяют три архитектурных идеи, для обеспечения инвариантности к изменению масштаба, повороту, сдвигу и пространственным искажениям:

- локальные рецепторные поля (обеспечивают локальную двумерную связность нейронов);
- общие веса (обеспечивают детектирование некоторых черт в любом месте изображения и уменьшают общее число весовых коэффициентов);
- иерархическая организация с пространственными подвыборками.

Топология нейронной сети, используемой в работе, изображена на рисунке 3.9.

Свёрточная нейронная сеть является многослойной. Используются слои двух типов: свёрточные и подвыборочные. Свёрточные и подвыборочные слои чередуются друг с другом. В свою очередь, каждый из этих слоёв состоит из набора плоскостей, причём нейроны одной плоскости имеют одинаковые веса (так называемые общие веса), ведущие ко всем локальным участкам предыдущего слоя (как в зрительной коре человека). Изображение предыдущего слоя сканируется небольшим окном и пропускается сквозь набор весов, а

результат отображается на соответствующий нейрон текущего слоя.

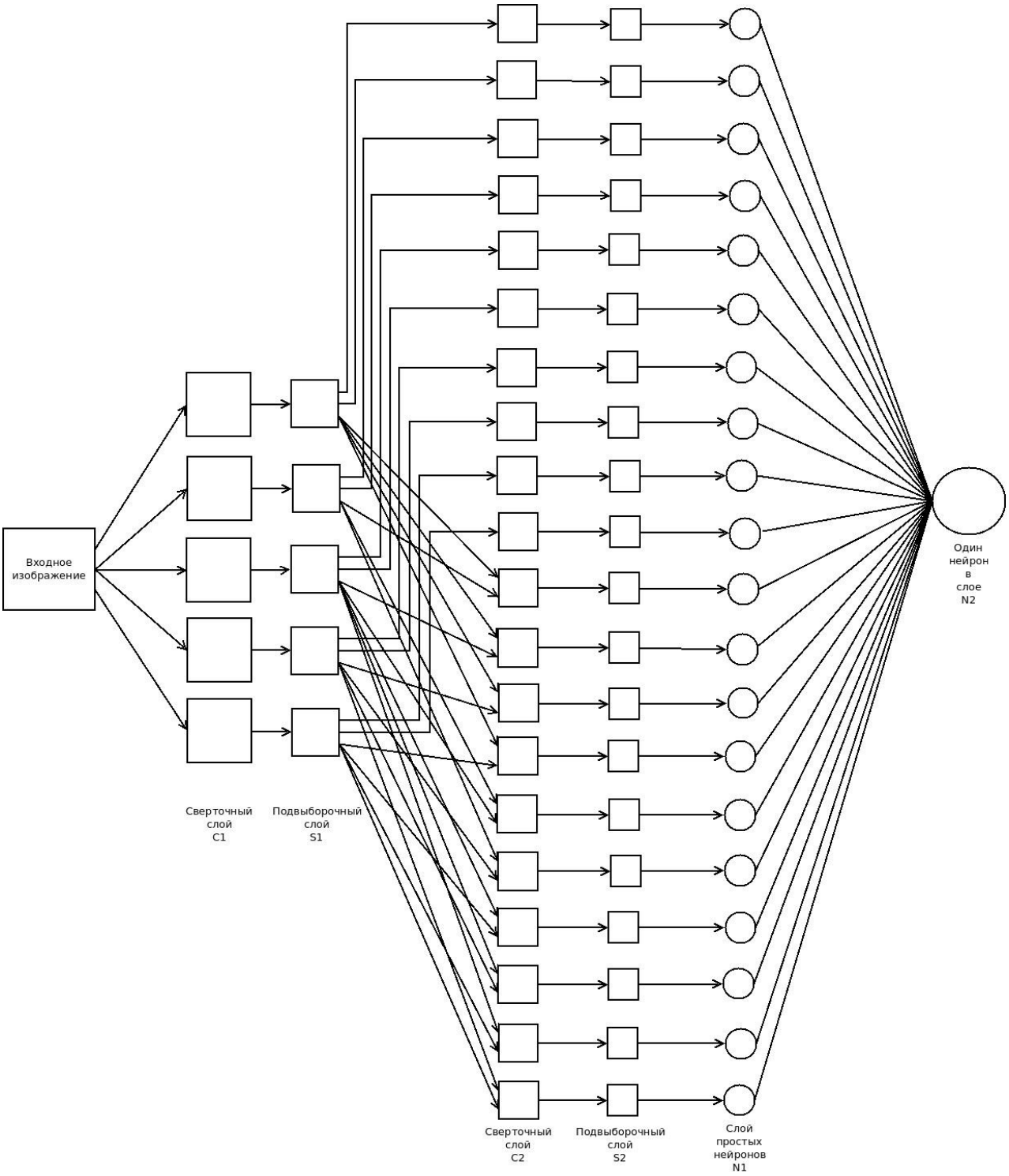

Рисунок 3.9 - Схема используемой нейронной сети

Таким образом, набор плоскостей предствляет собой карты характеристик, и каждая плоскость находит «свои» участки изображения в любом месте предыдущего слоя.

Используемая в работе нейронная сеть состоит из шести слоев. Входными данными нейронной сети являются полутоновые изображения размером 32x36 пикселей, которые классифицируются как лицо или «нелицо». Так как задача, решаемая нейронной сетью, – классификация, то для ее решения достаточно одного выхода. Выходное значение нейронной сети находится в интервале $[-1;1]$, что соответственно означает отсутствие или присутствие лица на классифицируемом изображении.

Входной слой размером 32x36 нейронов не несет какой-либо функциональной нагрузки и служит лишь для подачи входного образа в нейронную сеть. Следом за входным слоем находится сверточный слой C1. Каждый нейрон в плоскости свёрточного слоя получает свои входы от некоторой области предыдущего слоя (локальное рецептивное поле), то есть входное изображение предыдущего слоя как бы сканируется небольшим окном и пропускается сквозь набор весов, а результат отображается на соответствующий нейрон свёрточного слоя.

Процесс функционирования нейрона свёрточного слоя задается выражением:

$$y_k^{(i,j)} = b_k + \sum_{s=1}^{K} \sum_{t=1}^{K} w_{k,s,t} x^{((i-1)+s,(j+t))},$$

где $y_k^{(i,j)}$ – нейрон k-ой плоскости свёрточного слоя, b_k – нейронное смещение k-ой плоскости, K – размер рецептивной области нейрона, $w_{k,s,t}$ – элемент матрицы синаптических коэффициентов, x – выходы нейронов предыдущего слоя.

Слой C1 состоит из 5 свёрточных плоскостей и выполняет свёртывание входного изображения с помощью синаптической маски размером 5x5, таким образом, слой C1 осуществляет 5 свёрток входного изображения.

Размер свёрточной плоскости определяется в соответствии со следующими выражениями:

$$w_c = w_u - K + 1,$$
$$h_c = h_u - K + 1,$$

где w_c, h_c – ширина и высота свёрточной плоскости соответственно, *w, h* – ширина и высота плоскости предыдущего слоя, K – ширина (высота) окна сканирования.

Исходя из выражений для расчета размера свёрточной плоскости имеем, что размер плоскости сверточного слоя C1 – 28x32 нейрона. Нейроны в слое организованы в плоскости, в пределах которых все нейроны имеют один и тот же набор синаптических коэффициентов. Набор выходных сигналов в такой плоскости называют картой характеристик. Полный свёрточный слой составлен из нескольких карт характеристик с различными наборами синапсов так, чтобы множественные характеристики могли быть извлечены в каждом местоположении. Таким образом, набор плоскостей представляет собой карты характеристик, и каждая плоскость находит «свои» участки изображения в любом месте предыдущего слоя.

Как указано выше каждая плоскость слоя C1 имеет собственную синаптическую маску и нейронное смещение, рецептивные области нейронов пересекаются, нейроны извлекают одни и те же особенности входного изображения, независимо от их точного местоположения. Таким образом, слой C1 имеет всего лишь 130 настраиваемых параметров (синапсов).

Следующий за слоем C1 подвыборочный слой S1 состоит из 5 карт характеристик и обеспечивает локальное усреднение и подвыборку. Этот слой также состоит из плоскостей количество плоскостей такое же, как и в предыдущем слое. Рецепторная область каждого нейрона – 2x2 область в соответствующей карте особенностей предыдущего слоя. Каждый нейрон вычисляет среднее его четырех входов, умножает на синаптический коэффициент, добавляет нейронное смещение и передает результат через активационную функцию. Процесс функционирования нейрона подвыборочного слоя задается следующим соотношением:

$$y_k^{(i,j)} = b_k + 1/4\, w_k \sum_{s=1}^{2} \sum_{t=1}^{2} x^{(((i,j))+s,((i,j)))}.$$

Затем полученный результат подвыборки передается через активационную функцию. После операции подвыборки, точное местоположения и специфические признаки каждой особенности изображения становятся менее важными, что дает нейронной сети довольно большую степень инвариантности.

Смежные нейроны в подвыборочном слое имеют непересекающиеся рецептивные области. Следовательно, карта особенности слоя подвыборки имеет половину числа рядов и колонок карты особенности в предыдущем слое. В качестве активационной функции используется гиперболический тангенс $y = 1{,}7159 \tanh(2/3\, x)$ [20].

Каждая плоскость слоя S1 связана лишь с одной плоскостью слоя C1. Размер каждой плоскости слоя S1 – 14 16 нейронов, что вдвое меньше чем размер плоскости предыдущего слоя. Каждая плоскость слоя S1 имеет единственный синаптический коэффициент и нейронное смещение, что дает в итоге 10 настраиваемых параметров.

Свёрточный слой C2 состоит из 20 плоскостей, слои S1 и C2 перекрестно связаны. Плоскости слоя C2 формируются следующим образом: каждая из 5 плоскостей слоя S1 свёрнута 2 различными синаптическими масками 3x3, обеспечивая 10 плоскостей в C2, другие 10 плоскостей C2 получены, суммируя результаты 2 свёртываний на каждой возможной паре плоскостей слоя S1. Таким образом, сети добавляется способность объединять различные виды характеристик, чтобы составлять новые менее зависящие от искажений входного изображения.

Размер плоскости слоя C2 – 12 14 нейронов. Таким образом, данный слой имеет 290 синаптических коэффициентов. Слой S2 состоит из 20 плоскостей, размер каждой 6x7 нейронов. Каждая плоскость слоя S2 имеет единственный синаптический коэффициент и нейронное смещение, что дает в итоге 40 настраиваемых параметров.

Слои N1 и N2 содержат простые нейроны. Роль этих слоев состоит в обеспечении классификации, после того, как выполнены извлечение особенностей и сокращение размерности входа. В слое N1 находится 20 нейронов (по одному на каждую плоскость слоя S2), каждый нейрон полностью связан с каждым нейроном только одной плоскости слоя S2, он выполняет взвешенное суммирование своих 42 входов, добавляет нейронное смещение и пропускает результат через активационную функцию. Таким образом, данный слой содержит 860 синаптических коэффициентов.

Единственный нейрон слоя N2 полностью связан со всеми нейронами слоя N1. Роль этого нейрона в вычислении окончательного результата классификации. Выход этого нейрона используется для классификации входного образа на лица и не лица.

Использование принципа объединения весов дает эффект уменьшения количества настраиваемых параметров нейронной сети. Данная нейронная сеть имеет 1351 синаптический коэффициент.

Способность к обучению является фундаментальным свойством мозга. В контексте искусственных нейронных сетей процесс обучения может рассматриваться как настройка архитектуры сети и весов связей для эффективного выполнения специальной задачи. Процесс функционирования нейронной сети зависит от величин синаптических связей, поэтому, задавшись определенной структурой нейронной сети, отвечающей какой-либо задаче, необходимо найти оптимальные значения всех переменных коэффициентов (некоторые синаптические связи могут быть постоянными). Этот этап называется обучением нейронной сети, и от того, насколько качественно он будет выполнен, зависит способность сети решать поставленные перед ней проблемы во время эксплуатации. В основе всех алгоритмов обучения положен единый принцип – минимизация эмпирической ошибки. Функция ошибки, оценивающая данную конфигурацию сети, задается извне в зависимости от того, какую цель преследует обучение. Но далее сеть начинает постепенно

модифицировать свою конфигурацию – состояние всех своих синаптических весов таким образом, чтобы минимизировать эту ошибку.

Для обучения описанной нейронной сети был использован управляемый алгоритм обучения нейронной сети на основе генетического поиска и имитации отжига.

3.2.3. Генетические алгоритмы

Нейронные сети были созданы в результате наблюдения за естественными процессами, происходящими в нервной системе живых существ, и попыток воспроизведения этих процессов. Термин нейрон, обозначающий основной исполнительный элемент искусственных нейронных сетей, был непосредственно заимствован из теории природных нервных систем.

Аналогично, генетические алгоритмы возникли в результате наблюдения и попыток копирования естественных процессов, происходящих в мире живых организмов, в частности эволюции и связанной с ней селекцией (естественного отбора) популяций живых существ. Конечно, при подобном сопоставлении нейронных сетей и генетических алгоритмов следует обращать внимание на принципиально различную длительности протекания упоминаемых естественных процессов, т. е. На чрезвычайно быструю обработку информации в нервной системе и очень медленный процесс естественной эволюции. Однако при компьютерном моделировании эти различия оказываются несущественными.

Идею генетических алгоритмов (рисунок 3.10) высказал Дж. Холланд в конце шестидесятых — начале семидесятых годов XX века.

Он заинтересовался свойствами процессов естественной эволюции (в том числе фактом, что эволюционируют хромосомы, а не сами живые существа). Холланд был уверен в возможности составить и реализовать в виде компьютерной программы алгоритм, который будет решать сложные задачи так, как это делает природа — путем эволюции. Поэтому он начал трудиться над

алгоритмами, оперировавшими последовательностями двоичных цифр (единиц и нулей), получившими название хромосом. Эти алгоритмы имитировали эволюционные процессы в поколениях таких хромосом. В них были реализованы механизмы селекции и репродукции, аналогичные применяемым при естественной эволюции. Так же, как и в природе, генетические алгоритмы осуществляли поиск «хороших» хромосом без использования какой-либо информации о характере решаемой задачи. Требовалась только некая оценка каждой хромосомы, отражающая ее приспособленность. Механизм селекции (рисунок 3.11) заключается в выборе хромосом с наивысшей оценкой (т. е. наиболее приспособленных), которые репродуцируют чаще, чем особи с более низкой оценкой (хуже приспособленные).

Рисунок 3.10 - Схема простого генетического алгоритма

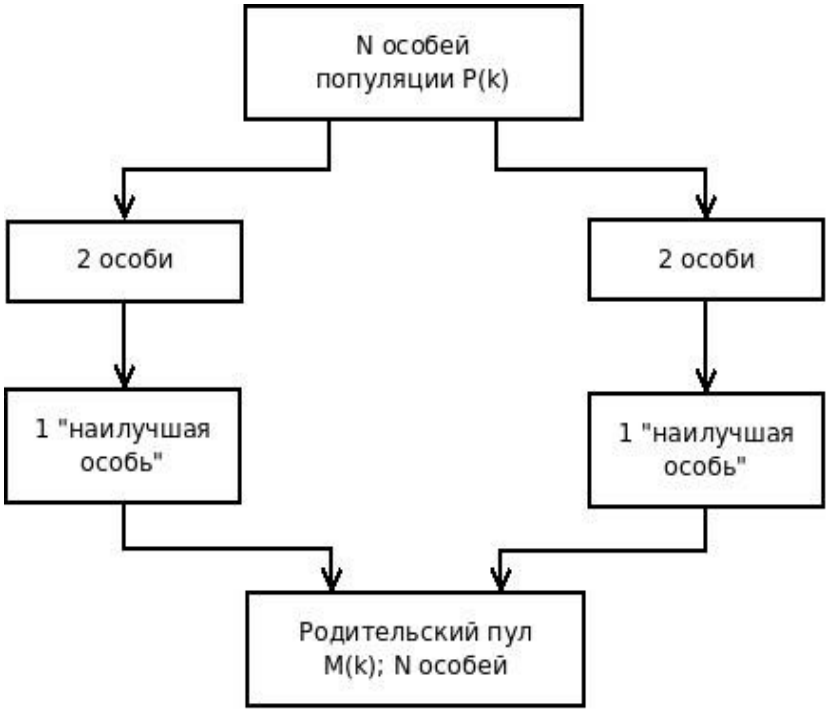

Рисунок 3.11 - Схема турнирной селекции

Репродукция означает создание новых хромосом в результате рекомбинации генов родительских хромосом.

Рекомбинация — это процесс, в результате которого возникают новые комбинации генов. Для этого используются две операции: скрещивание (рисунок 3.12), позволяющее создать две совершенно новые хромосомы потомков путем комбинирования генетического материала пары родителей, а так же мутация (рисунок 3.13), которая может вызывать изменения в отдельных хромосомах.

В генетических алгоритмах применяются ряд терминов, заимствованных из генетики, прежде всего гены и хромосомы, а также популяция, особь, аллель, генотип, фенотип.

Генетические алгоритмы применяются при разработке программного обеспечения, в системах искусственного интеллекта, оптимизации, искусственных нейронных сетях и в других отраслях знаний. Следует отметить, что с их помощью решаются задачи, для которых ранее использовались только

нейронные сети. В этом случае генетические алгоритмы выступают просто в роли независимого от нейронных сетей альтернативного метода, предназначенного для решения той же самой задачи.

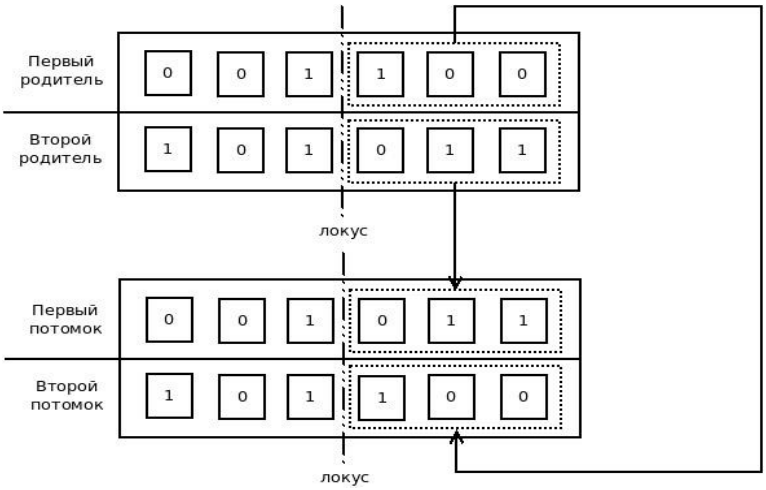

Рисунок 3.12 - Иллюстрация оператора скрещивания

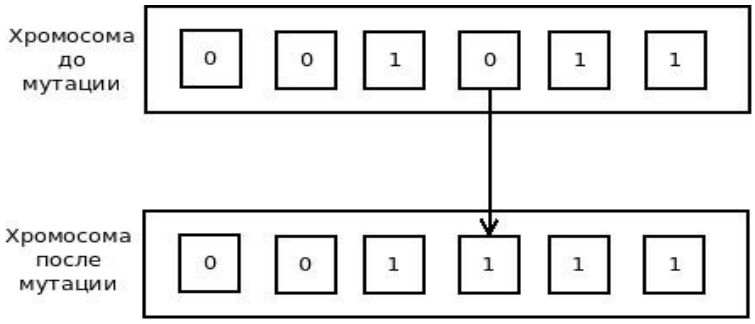

Рисунок 3.13 - Иллюстрация оператора мутации

Генетические алгоритмы часто используются совместно с нейронными сетями. Они могут поддерживать нейронные сети или наоборот, либо оба метода взаимодействуют в рамках гибридной системы, предназначенной для решения конкретной задачи [21].

3.2.4. Генетический алгоритм с вещественным кодированием

Идея использовать в виде хромосомы вектор вещественных значений появилась у исследователей в области ГА при решении задач непрерывной

оптимизации. Двоичное представление хромосом влечет за собой определенные трудности при выполнении поиска в непрерывных пространствах, которые связаны с большой размерностью пространства поиска.

Как известно, для кодирования признака, принимающего действительные значения в некотором диапазоне, в битовую строку, используется специальный прием. Интервал допустимых значений признака x_i разбивают на участки с требуемой точностью. Для преобразования целочисленного значения гена g_i из множества $\{0,...,2^N\}$ в вещественное число r_i из интервала пользуются формулой:

$$r_i = \frac{g_i * b_i - a_i}{2^N - 1} + a_i,$$

где N - количество разрядов для кодирования битовой строки. Чаще всего используются значения N = 8; 16; 32.

При увеличении N пространство поиска расширяется и становится огромным. В иностранных источниках по RGA часто приводится такой пример. Пусть для 100 переменных, изменяющихся в интервале [-500; 500], требуется найти минимум с точностью до шестого знака после запятой. В этом случае при использовании ГА с двоичным кодированием длина строки составит 3000 элементов, а пространство поиска - около 10 в степени 1000.

Для решения таких задач в непрерывных пространствах возник новый тип ГА - генетический алгоритм с вещественным кодированием (англ.: Real-coded Genetic Algorithm, RGA) [22], [23], [24]. Основная идея RGA заключается в том, чтобы напрямую представлять гены в виде вещественных чисел, т.е. генотип объекта становится идентичным его фенотипу. Вектор хромосомы состоит из вектора вещественных чисел, и точность найденного решения будет определяться не количеством разрядов для кодирования битовой строки, а будет ограничена возможностями ЭВМ, на которой реализуется вещественный ГА.

Применение вещественного кодирования может повысить точность найденных решений и повысить скорость нахождения глобального минимума

или максимума. Скорость повышается из-за отсутствия процессов кодирования и декодирования хромосом на каждом шаге алгоритма.

Для RGA стандартные операторы скрещивания и мутации не подходят, т.к. алгоритм работает только с вещественными числами. По этой причине были разработаны и исследованы специальные операторы. Наиболее полный их обзор приведен в [22].

Рассмотрим их подробно. Пусть и $C_2=(c_1^2, c_2^2, \dots, c_n^2)$ - две хромосомы, выбранные оператором селекции для проведения кроссовера.

1. Плоский кроссовер (англ.: flat crossover). Создается потомок $H=(h_1, \dots, h_i, \dots, h_n)$, где $h_i, i=\overline{1,n}$ - случайное число их интервала $[c_i^1, c_i^2]$;
2. Арифметический кроссовер (англ.: arithmetical crossover). Создаются два потомка $H_1=(h_1^1, \dots, h_n^1)$ и $H_2=(h_1^2, \dots, h_n^2)$:

   $h_i^1=\eta c_i^1+(1-\eta)c_i^2, h_i^2=\eta c_i^2+(1-\eta)c_i^1; i=\overline{1,n}, \eta\in[0,1]$ - константа;
3. $BLX-\alpha$ кроссовер. Генерируется один потомок $H=(h_1, \dots, h_i, \dots, h_n)$, где h_i - случайное число из интервала $[c_{min}-\Delta*\alpha, c_{max}+\Delta*\alpha], c_{max}=max(c_i^1, c_i^2), \Delta=c_{max}-c_{min}, i=\overline{1,n}$;
4. Линейный кроссовер (англ.: linear crossover). Создаются три потомка, рассчитываемые по формулам:

   $h_i^1=\frac{c_i^1+c_i^2}{2}, h_i^2=\frac{3c_i^1-c_i^2}{2}, h_i^3=\frac{-c_i^1+3c_i^2}{2}$.

На этапе селекции в линейном кроссовере выбираются две особи с наибольшими приспособленностями.

В качестве оператора мутации наибольшее распространение получили: случайная и неравномерная мутация Михалевича.

При случайной мутации ген, подлежащий изменению, принимает случайное значение из интервала своего изменения.

В неравномерной мутации значение гена после оператора мутации рассчитывается по формуле:

$$\tilde{c}_i = \begin{cases} c_i + \delta(t, b_i - c_i) \text{ при } \chi = 0 \\ c_i - \delta(t, c_i - a_i) \text{ при } \chi = 1 \end{cases},$$

$$\delta(t, y) = y\left(1 - r^{\left(1 - \frac{t}{\xi_{max}}\right)^b}\right),$$

где χ - целое случайное число, принимающее значение 0 или 1; $r \in [0,1]$ - случайное вещественное число; ξ_{max} - максимально количество эпох алгоритма; b - параметр, задаваемый исследователем.

В работе [22] проведен ряд экспериментов на тестовых функциях с использованием разных типов операторов скрещивания и мутации для ГА с вещественным кодированием (RGA). Например, классической тестовой функцией является N-мерная функция Розенброка, имеющая вид:

$$f(x_1, \ldots, x_{N+1}) = \sum_{i=1}^{N} (100 * (x_{i+1} - x_i^2)^2 + (1 - x_i)^2) \to min.$$

Оптимальные значения переменных при $f(X) = 0$ равны $x_i^{\cdot} = 1, i = \overline{1, N+1}$, $x_i \in [-5,5]$, $i = \overline{1, N+1}$. Данная функция имеет выраженный овражный характер.

Кроме того, полученные результаты сравнивались с результатами работы ГА с двоичным (бинарным) кодированием BGA. В большинстве случаев генетический алгоритм с вещественным кодированием справляется с задачей отыскания оптимума лучше и быстрее, чем с двоичным кодированием. Самым эффективным оператором скрещивания признан $BLX - \alpha$ кроссовер с $\alpha = 0.5$. Особенность данного оператора в том, что при скрещивании генов $c_i^1, c_i^2 (c_i^1 < c_i^2)$ начения потомка могут лежать в некоторой области, выходящей за границы значений этих генов на величину $\Delta * \alpha$, т. е. $h_i \in [c_i^1 - \Delta\alpha, c_i^2 - \Delta\alpha]$. В других кроссоверах, например, в плоском или арифметическом, $h_i \in [c_i^1, c_i^2]$.

3.2.5. Генетические алгоритмы для обучения нейронных сетей

Мысль о том, что нейронные сети могут обучаться с помощью генетического алгоритма, высказывались различными исследователями. Первые работы на эту тему касались применения генетического алгоритма в качестве метода обучения небольших однонаправленных нейронных сетей, но в следующем было реализовано применение этого алгоритма для сетей с большей размерностью [21].

Как правило, задача заключается в оптимизации весов нейронной сети, имеющей априори заданную топологию. Веса кодируются в виде двоичных последовательностей (хромосом). Каждая особь популяции характеризуется полным множеством весов нейронной сети. Оценка приспособленности особи определяется функцией приспособленности, задаваемой в виде суммы квадратов погрешностей, т. е. разностей между ожидаемыми (эталонными) и фактически получаемыми значениями на выходе сети для различных входных данных.

Приведем два важнейших аргументов в пользу применения генетических алгоритмов для оптимизации весов нейронной сети. Прежде всего, генетические алгоритмы обеспечивают глобальный просмотр пространства весов и позволяют избегать локальные минимумы. Кроме того, они могут использоваться в задачах, для которых информацию о градиентах получить очень сложно либо она оказывается слишком дорогостоящей.

Эволюционный подход к обучению нейронных сетей состоит из двух этапов. Первый из них — это выбор соответствующей схемы представления весов связей. Он заключается в принятии решения — можно ли кодировать эти веса двоичными последовательностями или требуется какая-то другая форма. На втором этапе уже осуществляется сам процесс эволюции, основанный на генетическом алгоритме.

После выбора схемы хромосомного представления генетический алгоритм применяется к популяции особей (хромосом, содержащих закодированное множества весов нейронной сети) с реализацией типового цикла эволюции,

состоящее из четырех шагов.

1) Декодирование каждой особи (хромосомы) текущего поколения для восстановления множества весов и конструирование соответствующей этому множеству нейронной сети с априорно заданной архитектурой и правилом обучения.
2) Расчет общей среднеквадратичной погрешности между фактическими и заданными значениями на всех выходах сети при подаче на ее входы обучающих образов. Эта погрешность определяет приспособленность особи (сконструированной сети); в зависимости от вида сети функция приспособленности может быть задана и другим образом.
3) Репродукция особей с вероятностью, соответствующей их приспособленности, либо согласно их рангу (в зависимости от способа селекции — например, по методу рулетки или ранговому методу).
4) Применение генетических операторов — таких как скрещивание, мутация и - или инверсия для получения нового поколения.

Блок схема, иллюстрирующая эволюцию весов, предоставлена на рисунке 3.14. В соответствии с первым этапом типового цикла эволюции априорно задаются и остаются неизменными архитектура сети, определяющая количество слоев, число нейронов в каждом слое и топологию межнейронных связей, а так же правило обучения сети. Приспособленность каждой особи (генотипа) оценивается значением среднеквадратичной погрешности, рассчитанной по соответствующей этой особи нейронной сети (фенотипу).

В предоставленном процессе эволюционного обучения реализуется режим так называемого пакетного обучения (batch training mode), при котором значения весов изменяются только после предъявления сети всех обучающих образов. Такой прием отличается от применяемого в большинстве последовательных алгоритмов обучения — например, в методе обратного

распространения ошибки веса уточняются после предъявления сети каждой обучающей выборки.

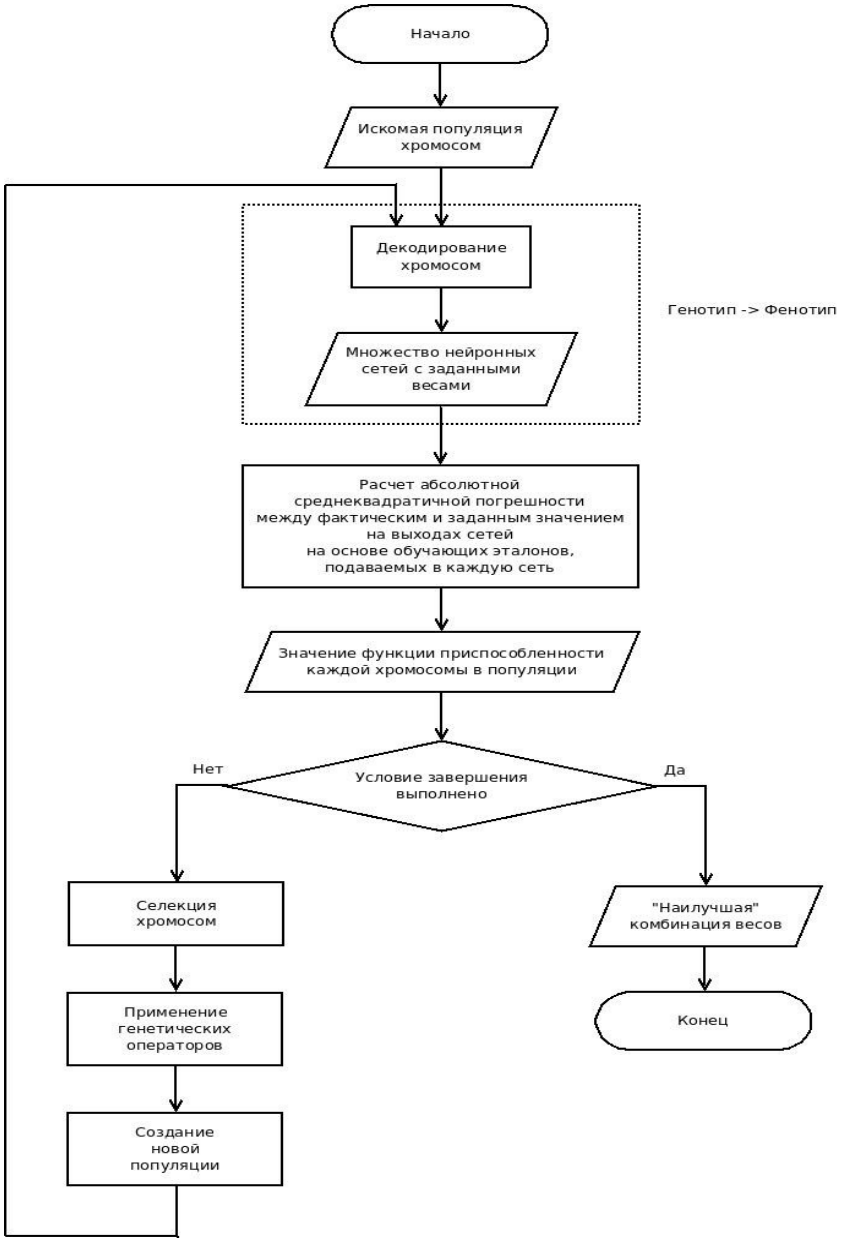

Рисунок 3.14 - Блок-схема генетического алгоритма поиска наилучшего набора весов нейронной сети (случай эволюции весов).

Рассмотрим более подробно первый этап эволюционного подхода к обучению, связанный с фиксацией схемы представления весов. Как уже отмечалось, необходимо выбрать между бинарным представлением и кодированием весов действительными числами. Помимо традиционного двоичного кода, может применяться код Грея, логарифмическое кодирование либо другие более сложные формы записи данных [21].

В роли ограничителя выступает требуемая точность представления значений весов. Если для записи каждого веса используется слишком мало битов, то обучение может продолжаться слишком долго и не принести ни какого эффекта, поскольку точность аппроксимации отдельных комбинаций действительных значений весов дискретными значениями весов часто оказывается недостаточной. С другой стороны, если используется слишком много битов, то двоичные последовательности, представляющие нейронные сети большой размерности, оказывается очень длинными, что сильно удлиняет процесс эволюции и делает эволюционный подход к обучению не рациональным с практической точки зрения. Вопрос оптимизации количества битов для представления конкретных весов все еще остается открытым [21].

3.2.6. Управляемый алгоритм обучения нейронной сети на основе генетического поиска и имитации отжига

Алгоритм имитации отжига (simulated annealing) основывается на понятии тепловой энергии, введенной С. Кирпатриком. Автор алгоритма использовал «тепловой шум» для выхода из локальных минимумов и для повышения вероятности попадания в более глубокие минимумы. При решении сложных задач, когда вычислительные затраты на решение задачи оптимизации аналогичны энергии шарика, перемещающегося по поверхности, поиск более дешевых решений разумно начинать в ситуации с высоким уровнем «теплового шума», а в дальнейшем постепенно уменьшать его, этот процесс Кирпатрик назвал «имитацией отжига».

Метод имитации (моделирования) отжига базируется на аналогии с процессом отжига металла, в результате которого металл приобретает новые свойства. При отжиге металл вначале подвергается нагреву почти до точки плавления, а потом медленно охлаждается до обычной температуры. Эта процедура делает металл более гибким и позволяет легко придать ему необходимую форму. Когда металл быстро нагревается до высокой температуры, его атомы начинают двигаться случайно, и, если его также быстро

остудить, то атомы фиксируются в случайных состояниях. Но если металл охлаждать медленно, то атомы пытаются выстроиться некоторым регулярным образом. Поэтому основой процесса отжига металла является процедура управления графиком снижения температуры.

Аналогично методу имитации отжига металла можно действовать при поиске глобального экстремума (например, минимума) некоторой функции. Вначале выбирается диапазон, в котором будут оцениваться значения функции в случайно выбранных точках, при этом запоминается наименьшее из них. Затем диапазон уменьшается, и снова оцениваются значения функций в случайно выбранных точках, при этом сохраняются меньшие из этих значений. Процесс повторяется с постепенным уменьшением диапазона исследования функции.

Выбор диапазона и постепенное его уменьшение аналогично установке начальной температуры и изменению этой температуры на последующих стадиях.

Алгоритм имитации отжига обладает следующими характеристиками:

- простота структуры, легкость понимания и реализации, гибкая настройка на специфику решаемой задачи (посредством выбора функции генерации m и коэффициента α, значение которого обычно выбирается из диапазона от 0,8 до 0,9999, а начальное значение T выбирается так, что для всех изменений функций E имеет место неравенство $\exp(-\Delta E / T) \geq 0{,}9999$;
- допускается использование алгоритма для решения оптимизационных задач различной природы, для оптимизации недифференцируемых, разрывных, зашумленных функций;
- пространство оптимизируемых параметров может быть разнородным и включать как комбинаторные структуры, так и действительные параметры;
- алгоритм комбинирует методы как глобальной, так и локальной

оптимизации: на начальных стадиях оптимизации, когда температура высока, исследуется пространство допустимых решений и с высокой вероятностью отыскивается область «хороших решений»; при снижении температуры осуществляется быстрая сходимость к точке локального минимума.

Значительное влияние на эффективность метода имитации отжига оказывает выбор начальной температуры T, значения коэффициента уменьшения температуры α, а так же количества циклов, выполняемых на каждом температурном уровне.

Максимальная температура и коэффициент α обычно определяются путем проведения большого числа предварительных экспериментов и статической оценки результатов. Конечное число циклов удается сократить более частым изменением температуры при сохранении общего объема итераций. Наибольшее ускорение процесса имитации отжига при обучении НС может быть достигнуто, если предварительно определить значения весовых коэффициентов с использованием любых доступных методов статической обработки исходных данных.

3.2.6.1. Модификация оператора генной мутации

Применение оператора случайной мутации в ГА фактически означает формирование новых генов, что, в конечном итоге, приводит к расширению области поиска и повышению вероятности нахождения оптимального решения. Однако случайные мутации с равной вероятностью могут привести как к увеличению значения функции фитнесса, так и к ее уменьшению. Поэтому на этапе сходимости генетического алгоритма к оптимуму целесообразно уменьшать вероятность случайной мутации. Таким образом, желательно динамически управлять вероятностью случайной мутации в процессе работы ГА: на начальном этапе поиска значение вероятности должно быть достаточно высоким (0,05 ...0,1), а на конечном этапе - стремиться к нулю.

Для реализации этой процедуры воспользуемся аналогией имитации отжига. В соответствии с формулой Н. Метрополиса (Metropolis N.) вероятность принятия изменения состояния системы пропорциональна управляющему параметру Т (аналога температуры). При больших значениях Т вероятность принятия измененного состояния велика, что соответствует большой величине вероятности случайной мутации на начальном этапе работы алгоритма (для увеличения генетического разнообразия). Чем меньше Г, тем меньше вероятность принятия измененного состояния, что имеет место на этапе сходимости ГА. Таким образом, вероятность случайной мутации должна динамически изменяться от итерации к итерации (при переходе от одного поколения к другому), для этого:

$$p_m = p_m^0 \exp[-1/T],$$

где p_m^0 - начальное значение вероятности случайной мутации; Т — управляющий параметр.

3.2.6.2. Модификация операторов селекции

Применение стандартных и разработанных операторов кроссинговера требует определения стратегии выбора пар хромосом в операциях скрещивания на отдельных этапах работы ГА с целью повышения качества получаемого решения. Организация динамической стратегии скрещивания на основе имитации отжига позволяет варьировать методы селекции пар хромосом: случайный выбор, лучший с лучшим, лучший с худшим, «близкое родство», «дальнее родство», лучший со всеми и т. д.

Случайный выбор пар хромосом позволяет разнообразить генофонд на ранних стадиях работы ГА. Вероятность этого выбора должна снижаться при эволюции поколений. Тогда :

$$p_{сл} = p_{сл}^0 \frac{\exp[-1/T]}{r},$$

где r — размер популяции; $p_{сл}^0$ - начальное значение вероятности

случайного выбора пар.

Вероятность скрещивания лучших хромосом с худшими p , оцениваемыми по значению функции Fit, также должна уменьшаться при эволюции поколений.

Выбор первой хромосомы H_i осуществляется с вероятностью:

$$p_{лх}^{ш} = \frac{Fit_i}{\sum_{k=1}^{r} Fit_k} \exp[-1/T].$$

Выбор второй хромосомы H_j :

$$p_{лх}^{j} = \frac{1}{Fit_j} \cdot \frac{1}{\sum_{k=1}^{r} \frac{1}{Fit_k}} \exp[-1/T].$$

«Близкое родство». Вероятность выбора хромосом, подлежащих скрещиванию, определяется следующим образом:

- для первой хромосомы H_i :

$$p_{бр}^{J} = \sqrt{p_{бр}^{0}}(1 - \exp[-1/T]),$$

где $p_{бр}^{0}$ - вероятность «близкого родства» на последних стадиях работы алгоритма;

- для второй хромосомы H_j вероятность $p_{бр}^{J}$ вычисляется по приведенной выше формуле, но из оставшихся хромосом $P \setminus \{H_j\}$ где P- текущая популяция.

Затем вычисляется Хеммингово расстояние между выбранными хромосомами текущей популяции $dist(H_i, H_j)$, равное количеству позиций с несовпадающими значениями генов в хромосомах. Хромосомы подлежат скрещиванию, если dist<R, где R - радиус скрещивания, задаваемый априорно. Вероятности $p_{бр}^{i}$ и $p_{бр}^{j}$ возрастают на конечных стадиях работы алгоритма.

«Дальнее родство». В этом случае вероятность выбора хромосом H_i и H_j осуществляется по формуле :

$$p_{др} = \sqrt{p_{др}^{0}} \exp[-1/T],$$

где $p_{\partial p}^{0}$ вероятность «дальнего родства» на начальных стадиях работы ГА, с учетом особенностей вычисления для первой и второй хромосомы.

Хромосомы H_i и H_j подлежат скрещиванию, если Хеммингово расстояние между ними dist>R. Вероятности $p_{\partial p}^{i}$ и $p_{\partial p}^{j}$ уменьшаются на конечных стадиях поиска оптимального решения.

3.2.6.3. Модификация операторов отбора

Для формирования репродукционной группы в комбинированном алгоритме на основе ГА и имитации отжига использовались две схемы отбора:

1) равновероятный отбор с вероятностью :

$$p(H_k)=1/r,$$

где r — размер популяции;

2) пропорциональный:

$$p(H_k)=\frac{Fit_k}{\sum_{i=1}^{r} Fit_i},$$

где Fit_k - значение фитнеса для k-ой хромосомы.

Для динамической стратегии отбора на основе имитации отжига имеем:

$$p(H_k)=\frac{1}{r}\exp[-1/T]+\frac{Fit_k}{\sum_{i=1}^{r} Fit_i}(1-\exp[-1/T]),$$

Тогда при $t\to\infty\ p(H_k)\to 1/r$ (равновероятный отбор);

при $T\to 0\ p(H_k)\to\frac{Fit_k}{\sum_{i=1}^{r} Fit_i}$ (пропорциональный отбор).

Таким образом, с помощью операторов отбора на ранних стадиях работы комбинированного алгоритма происходит выбор хромосом без учета значений их функции $Fit(H_k)$, т. е. имеет место случайный отбор, на заключительной стадии определяющим фактором при отборе является значение функции

$Fit(H_k)$: чем выше $Fit(H_k)$, тем выше вероятность отбора H_k в следующую популяцию.

Совокупность рассмотренных динамических стратегий на основе имитации отжига для основных генетических операторов (мутации, селекции пар хромосом для скрещивания и отбора) определяет механизм управления процессом генетического поиска. Данный механизм в процессе поиска оптимального решения позволяет на начальных стадиях работы комбинированного алгоритма проводить больше случайных операций с целью повышения генетического разнообразия популяций: большой процент случайной генной мутации, «дальнее родство», скрещивание лучших и худших хромосом, случайный отбор.

На заключительной стадии проводится уменьшение случайных операций и постепенно увеличивается процент направленных операций: «близкое родство», скрещивание лучших хромосом, пропорциональный отбор.