Автономная некоммерческая организация высшего образования

«Российский новый университет»

Факультет информационных систем и компьютерных технологий

Курсовая работа/курсовой проект

по дисциплине: Математические модели нелинейной динамики\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

на тему: Обучающиеся системы. Нейронные сети\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

студента\_\_449\_\_ группы\_\_\_\_4\_\_\_\_ курса

\_\_\_\_\_очной\_\_\_\_\_\_\_\_\_\_\_ формы обучения

направления подготовки:

Математическое обеспечение и администрирование информационных систем

направленности (профиля): Технологии программного обеспечения

Семенова Кирилла Андреевича\_\_.

(фамилия, имя, отчество)

Руководитель:

Крюковский Андрей Сергеевич научный руководитель\_\_

(должность, звание, фамилия, имя, отчество)

Работа допущена к защите «\_\_\_»\_\_\_\_\_\_20\_\_г.

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Регистрационный номер\_\_\_\_\_\_\_\_ от «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_20\_\_\_г.

Москва

2023 г.

Содержание………………………………………………………………………

1. Введение……………………………………………………………………...3

1.1 История происхождения нейронных сетей………………………...4

1.2 Аналогия биологической и искусственной нейронных сетей.......7

1.3 Применение нейронных сетей……………………………………...8

1.4 Новейшие разработки в области нейронных сетей……………...10

2. Принцип работы нейронных сетей………………………………………..11

2.1. Основные компоненты и принципы работы нейросети………...11

2.2. Алгоритм обратного распространения ошибки……....................16

3. Свёрточные нейросети……………………………………………………..18

3.1. О свёрточных нейронных сетях….…………………………….…18

3.2. История и эволюция…………………………………….…………18

3.3 Применение…………………………………………………………18

3.4 Особенности………………………………………………………...19

3.5 Вывод………………………………………………………………..20

4. Заключение …..……………………………………………………………..21

5. Библиографический список.……………………………………………….22

Приложение: Листинг программы, визуализация обучения модели и дополнительная информация…….…………………………………………..23

1. **Введение**

В эпоху быстрого развития технологий искусственного интеллекта, нейронные сети, вдохновленные биологическими нейросетями человеческого мозга, играют ключевую роль в решении сложных задач распознавания образов. Они представляют собой мощный инструмент, способный обучаться на больших объемах данных и успешно адаптироваться к разнообразным сценариям.

Цель данной курсовой работы заключается в глубоком изучении нейронных сетей, сосредотачивая внимание на их применении в области распознавания образов. Распознавание образов – это ключевая область, где нейронные сети проявляют свою выдающуюся эффективность, позволяя машинам интерпретировать и анализировать визуальные данные с уровнем точности, сравнимым с человеческим восприятием.

Мы рассмотрим разнообразные архитектуры нейронных сетей, оптимизированных для распознавания образов, выделяя сверточные нейронные сети (CNN), которые успешно применяются в анализе изображений.

Особое внимание уделено исследованию методов обучения нейронных сетей для достижения высокой точности распознавания образов. Примеры включают в себя использование маркированных данных для обучения и техники передачи обучения, позволяющие эффективно применять предобученные модели к новым задачам.

Нейронные сети, способные распознавать образы, играют существенную роль в современном обществе, применяясь в таких областях, как медицина, автомобильная промышленность, безопасность и многие другие. В этой работе мы стремимся разглядеть не только технические аспекты функционирования таких сетей, но и их влияние на повседневную жизнь и перспективы развития этой захватывающей области исследований.

Данная работа направлена на выполнение следующих целей:

1. Ознакомиться с основными принципами работы нейросетей.
2. Изучить сходства и различия различных типов нейронных сетей.
3. Изучить особенности и самим реализовать нейронную сеть сверточного типа, предназначенную для классификации предметов одежды, аксессуаров и обуви.

**1.1 История происхождения нейронных сетей.**

Термин "нейронная сеть" появился в середине XX века. Первые работы, в которых были получены основные результаты в данном направлении, были проделаны Мак-Каллоком и Питтсом. В 1943 году ими была разработана компьютерная модель нейронной сети на основе математических алгоритмов и теории деятельности головного мозга. Они выдвинули предположение, что нейроны можно упрощённо рассматривать как устройства, оперирующие двоичными числами, и назвали эту модель "пороговой логикой". Подобно своему биологическому прототипу нейроны Мак-Каллока-Питтса были способны обучаться путём подстройки параметров, описывающих синаптическую проводимость. Исследователи предложили конструкцию сети из электронных нейронов и показали, что подобная сеть может выполнять практически любые вообразимые числовые или логические операции. Мак-Каллок и Питтс предположили, что такая сеть в состоянии также обучаться, распознавать образы, обобщать, т. е. обладает всеми чертами интеллекта.

Данная модель заложила основы двух различных подходов исследований нейронных сетей. Один подход был ориентирован собственно на изучение биологических процессов в головном мозге, другой - на применение нейронных сетей как метода искусственного интеллекта для решения различных прикладных задач.

В 1949 году канадский физиолог и психолог Хебб высказал идеи о характере соединения нейронов мозга и их взаимодействии. Он первым предположил, что обучение заключается в первую очередь в изменениях силы синаптических связей. Теория Хебба считается типичным случаем самообучения, при котором испытуемая система спонтанно обучается выполнять поставленную задачу без вмешательства со стороны экспериментатора. В более поздних вариантах теория Хебба легла в основу описания явления долговременной потенциации.

В 1954 году в Массачусетском технологическом институте с использованием компьютеров Фарли и Кларк разработали имитацию сети Хебба. Также исследования нейронных сетей с помощью компьютерного моделирования были проведены Рочестером, Холландом, Хебитом и Дудой в 1956 году.

В 1957 году Розенблаттом были разработаны математическая и компьютерная модели восприятия информации мозгом на основе двухслойной обучающейся нейронной сети. При обучении данная сеть использовала арифметические действия сложения и вычитания. Розенблатт описал также схему не только основного перцептрона, но и схему логического сложения. В 1958 году им была предложена модель электронного устройства, которое должно было имитировать процессы человеческого мышления, а два года спустя была продемонстрирована первая действующая машина, которая могла научиться распознавать некоторые из букв, написанных на карточках, которые подносили к его "глазам", напоминающим кинокамеры.

Интерес к исследованию нейронных сетей угас после публикации работы по машинному обучению Минского и Пейперта в 1969 году. Ими были обнаружены основные вычислительные проблемы, возникающие при компьютерной реализации искусственных нейронных сетей. Первая проблема состояла в том, что однослойные нейронные сети не могли совершать "сложение по модулю 2", то есть реализовать функцию "Исключающее ИЛИ". Второй важной проблемой было то, что компьютеры не обладали достаточной вычислительной мощностью, чтобы эффективно обрабатывать огромный объём вычислений, необходимый для больших нейронных сетей.

Исследования нейронных сетей замедлились до того времени, когда компьютеры достигли больших вычислительных мощностей. Одним из важных шагов, стимулировавших дальнейшие исследования, стала разработка в 1975 году Вербосом метода обратного распространения ошибки, который позволил эффективно решать задачу обучения многослойных сетей и решить проблему со "сложением по модулю 2".

В 1975 году Фукусимой был разработан когнитрон, который стал одной из первых многослойных нейронных сетей. Фактическая структура сети и методы, используемые в когнитроне для настройки относительных весов связей, варьировались от одной стратегии к другой. Каждая из стратегий имела свои преимущества и недостатки. Сети могли распространять информацию только в одном направлении или перебрасывать информацию из одного конца в другой, пока не активировались все узлы и сеть не приходила в конечное состояние. Достичь двусторонней передачи информации между нейронами удалось лишь в сети Хопфилда (1982), и специализация этих узлов для конкретных целей была введена в первых гибридных сетях.

Алгоритм параллельной распределённой обработки данных в середине 1980 годов стал популярен под названием коннективизма. В 1986 году в работе Руммельхарта и Мак-Клелланда коннективизм был использован для компьютерного моделирования нейронных процессов.

Несмотря на большой энтузиазм, вызванный в научном сообществе разработкой метода обратного распространения ошибки, это также породило многочисленные споры о том, может ли такое обучение быть на самом деле реализовано в головном мозге. Отчасти это связывали с тем, что механизм обратного прохождения сигнала не был очевидным в то время, так как не было явного источника обучающего и целевого сигналов. Тем не менее, в 2006 году было предложено несколько неконтролируемых процедур обучения нейронных сетей с одним или несколькими слоями с использованием так называемых алгоритмов глубокого обучения. Эти алгоритмы могут быть использованы для изучения промежуточных представлений, как с выходным сигналом, так и без него, чтобы понять основные особенности распределения сенсорных сигналов, поступающих на каждый слой нейронной сети.

Как и во многих других случаях, задачи высокой сложности требуют применения не одного, а нескольких методов решения или их синтеза. Не исключение и искусственные нейронные сети. С самого начала нынешнего столетия в работах различных исследователей активно описываются нейро-нечёткие сети, ячеечно-нейросетевые модели. Также нейронные сети используются, например, для настройки параметров нечётких систем управления. В общем, нет никаких сомнений и в дальнейшей интеграции методов искусственного интеллекта между собой и с другими методами решения задач.[1]

**1.2 Аналогия биологической и искусственной нейронных сетей.**

Нейронная сеть — это последовательность нейронов, соединенных между собой синапсами. Структура нейронной сети пришла в мир программирования прямиком из биологии. Благодаря такой структуре, машина обретает способность анализировать и даже запоминать различную информацию. Нейронные сети также способны не только анализировать входящую информацию, но и воспроизводить ее из своей памяти. Другими словами, нейросеть это машинная интерпретация мозга человека, в котором находятся миллионы нейронов передающих информацию в виде электрических импульсов.[2]

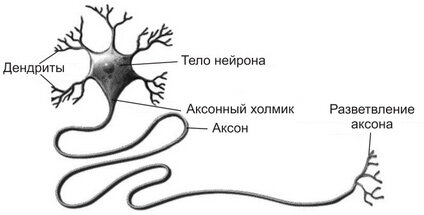


Рис.1 Биологический нейрон.

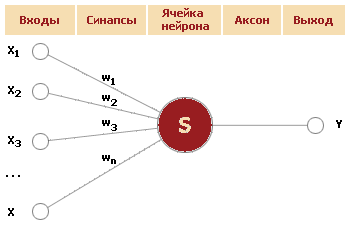


Рис.2 Нейрон искусственной нейронной сети.

На рисунках 1 и 2 можно увидеть, насколько схожи нейроны мозга и искусственной нейросети.

**1.3 Применение нейронных сетей.**

Нейронные сети широко применяются в различных областях благодаря своей способности обучаться на данных и решать сложные задачи.

Ниже приведены некоторые из основных областей применения нейронных сетей:

1. **Обработка изображений и компьютерное зрение:**

Распознавание образов, классификация объектов, сегментация изображений, распознавание лиц и др.

1. **Обработка естественного языка:**

Машинный перевод, распознавание и синтез речи, анализ текста, автоматическая генерация текста.

1. **Медицинская диагностика:**

Анализ медицинских изображений (рентген, МРТ, КТ), предсказание диагнозов, исследование геномных данных.

1. **Финансовая аналитика:**

Прогнозирование финансовых рынков, обнаружение мошенничества, кредитный скоринг.

1. **Автоматическое управление и робототехника:**

Автоматическое управление транспортными средствами, управление производственными роботами, автономные транспортные средства.

1. **Игры и развлечения:**

Создание искусственного интеллекта для видеоигр, персонализированные рекомендации для пользователей развлекательных платформ.

1. **Промышленность и производство:**

Прогнозирование отказов оборудования, оптимизация производственных процессов, контроль качества продукции.

1. **Энергетика:**

Прогнозирование энергопотребления, управление распределенной энергетикой, диагностика состояния энергетического оборудования.

1. **Биоинформатика:**

Анализ биологических данных, предсказание структуры белков, исследование геномов.

1. **Интернет вещей (IoT):**

Мониторинг и управление умными устройствами, анализ данных от датчиков, автоматизация домашней техники.

Эти области представляют лишь небольшой перечень множества сфер, в которых нейронные сети применяются для решения разнообразных задач. С появлением новых методов обучения и улучшением архитектур, ожидается, что области применения нейронных сетей будут продолжать расширяться.

**1.4 Новейшие разработки в области нейронных сетей.**

Американские ученые из Университета штата Индиана в Блумингтоне создали новый тип нейросети с помощью живых органоидов мозга, выращенных из стволовых клеток. Исследование опубликовано в научном журнале Nature Electronics (NatElectron).

Органоиды имеют мало общего с настоящим мозгом человека, представляя собой миниатюрный объект из специфического типа тканей. Хотя в нем не могут появиться мысли, эмоции и тем более сознание, такие структуры помогают изучать строение органа без необходимости проведения экспериментов на людях.

Ученые обратились к органоидам для проектирования нейросетей, поскольку биологический мозг требует намного меньше энергии по сравнению с полностью электронными системами искусственного интеллекта (ИИ). Например, нашему мозгу достаточно около 20 Вт энергии для работы в течение дня, тогда как основанному на микропроцессорах ИИ требуется около 8 млн Вт, чтобы достичь чего-то отдаленного сопоставимого.

Новая разработка под названием Brainware представляет собой органоид диаметром менее нанометра, установленный на многоэлектродную матрицу высокой плотности. Чип способен отправлять электрические сигналы в нервную ткань и считывать возникающую там электрическую активность.

Готовую кибернетическую систему протестировали в двух типах задач — распознавании речи и решении нелинейных уравнений. В первом тесте точность расшифровки фраз на японском за двое суток увеличилась с 51% до 78%.

В математических расчетах Brainware оказалась заметно быстрее искусственных нейросетей (ИНС) без блока долговременной памяти и лишь немного уступила в точности ИНС с долговременной памятью. Однако ученые отметили, что более результативные искусственные нейросети тратили гораздо больше времени на обучение по сравнению с основанным на органоиде ИИ. В равных условиях Brainware учился на 90% быстрее полностью электронных аналогов.

По словам исследователей, их открытие стало важным шагом на пути к новому типу компьютерной архитектуры. В то же время ученые подчеркнули, что при дальнейшем развитии подобных технологий нужно учитывать этические аспекты. Кроме того, для поддержания органоидов требуются системы жизнеобеспечения, необходимость в которых отчасти сводит на нет экономию в энергопотреблении.[3]

**2. Принцип работы нейронных сетей.**

Принцип работы нейронной сети базируется на моделировании функций человеческого мозга и процесса обучения. Нейронные сети состоят из множества соединенных и взаимодействующих искусственных нейронов, объединенных в слои.[4]

**2.1 Основные компоненты и принципы работы нейросети:**

**1. Нейроны:**

- Нейроны являются базовыми строительными блоками нейронной сети. Каждый нейрон принимает входные сигналы, обрабатывает их с использованием весов (параметров) и активационной функции, а затем передает выходной сигнал на следующий слой.[5]

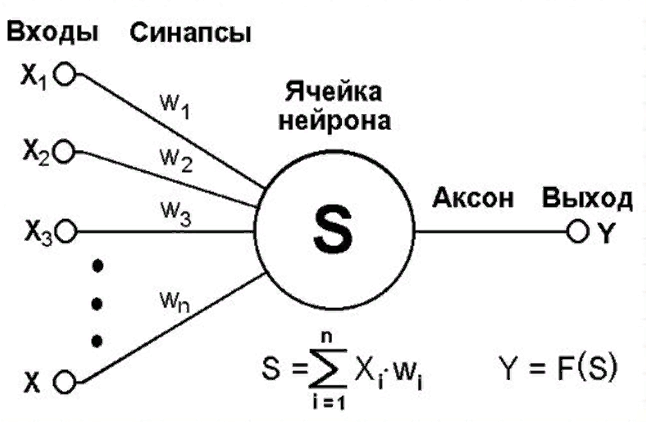


Рис. 3 Общий вид искусственного нейрона.

**2. Слои:**

Нейроны организованы в слои. В типичной нейронной сети можно выделить три основных типа слоев: входной слой, скрытые слои и выходной слой. Входной слой принимает входные данные, скрытые слои выполняют вычисления, а выходной слой предоставляет результат работы сети.[6]

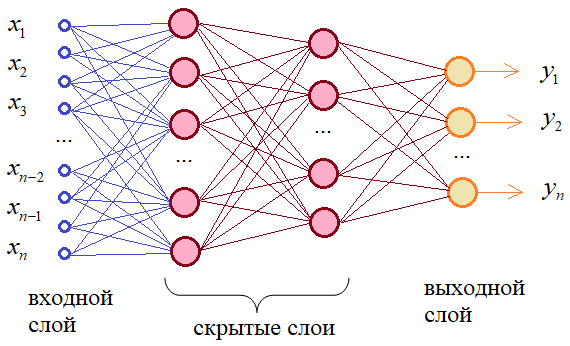


Рис.4 Структура многослойной нейронной сети.

**3. Веса и связи:**

Веса определяют силу связей между нейронами. Каждая связь имеет свой вес, который умножается на входной сигнал, передаваемый между нейронами. Веса подлежат обучению в процессе тренировки сети.

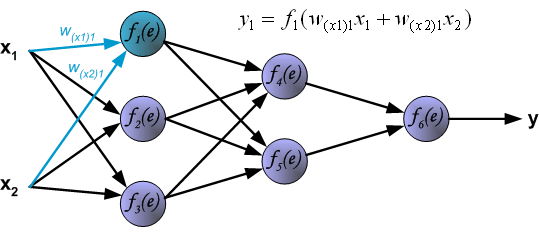


Рис.5 Иллюстрация нейросети с указанием значений весов и связей.

**4. Активационные функции:**

Активационные функции определяют, будет ли активирован нейрон и передаст ли он свой выходной сигнал следующему нейрону. Популярные активационные функции включают сигмоидальную, гиперболическую тангенс и ReLU (Rectified Linear Unit).[7]

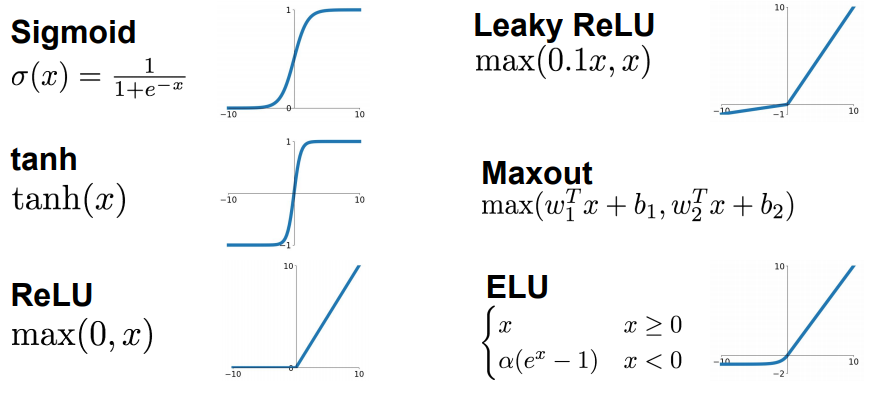


Рис.6 Популярные функции активации.

**5. Нейрон смещения**

Нейрон смещения (bias neuron) представляет собой дополнительный компонент в искусственных нейронных сетях, который не имеет входных связей и предназначен для управления смещением выхода нейрона. В контексте нейронных сетей он также известен как "свободный член" или "смещение" (bias).

Каждый нейрон в нейронной сети, помимо входных связей с входными данными и весами, имеет дополнительный параметр — смещение. Смещение позволяет нейрону сдвигать свою активацию вверх или вниз, что является важным элементом в обучении нейронных сетей.

Использование нейрона смещения позволяет учитывать случаи, когда все входы нейрона равны нулю, и вносит гибкость в регулирование активации нейрона в зависимости от контекста.

В обучении нейронных сетей параметр смещения, как и веса, подлежит обновлению в процессе обратного распространения ошибки с целью минимизации функции потерь и улучшения производительности сети.

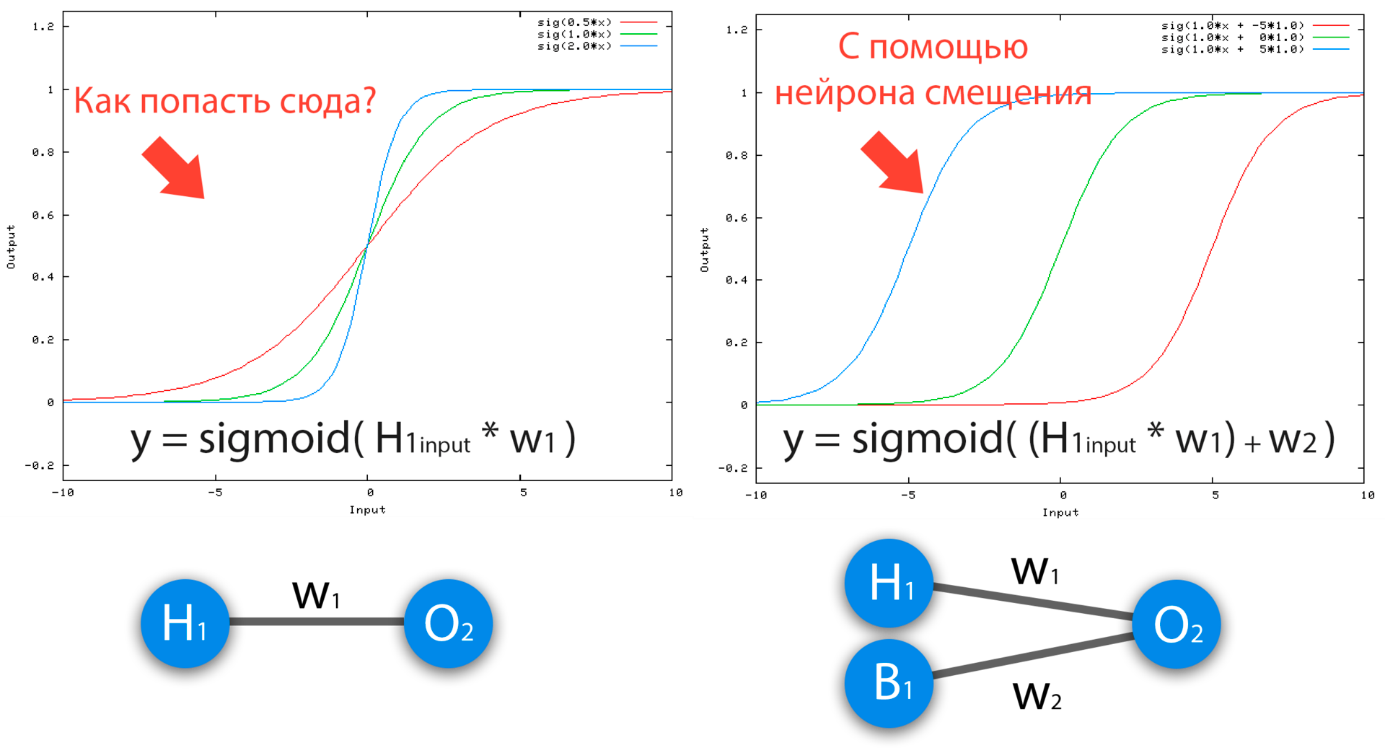


Рис.7 Визуализация применения нейрона смещения

**6. Функция потерь и оптимизация:**

Функция потерь измеряет разницу между предсказанными значениями сети и фактическими значениями. В процессе обучения используется метод оптимизации для минимизации функции потерь, такой как градиентный спуск. Самыми популярными являются **MSE(Mean Squared Error) - среднеквадратичная ошибка, Binary Cross-Entropy – бинарная кросс-энтропия, Categorical Cross-Entropy – категориальная кросс-энтропия**. Выбор функции зависит от конкретной задачи, типа данных и структуры модели. Важно подобрать функцию потерь, которая соответствует постановке задачи и природе данных.

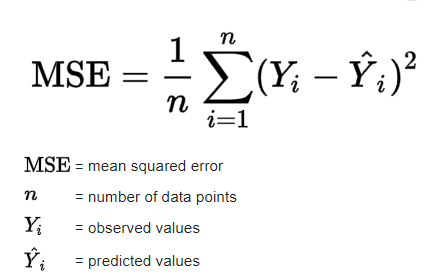


Рис.8 Формула среднеквадратичной ошибки.

**7. Обучение:**

Обучение нейронной сети включает в себя предоставление ей обучающих данных и коррекцию весов сети на основе ошибок прогноза. Этот процесс осуществляется путем обратного распространения ошибки (Backpropagation), где градиент функции потерь передается назад по сети для коррекции весов.

Рассмотрим его подробнее, так как это основной алгоритм, отвечающий за обучение нейронной сети.

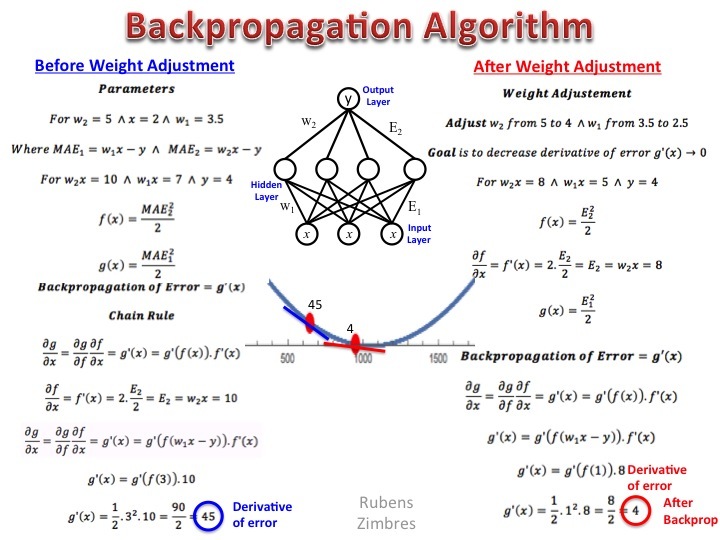


Рис.9 Алгоритм обратного распространения ошибки(Backpropagation).

**2.2 Алгоритм обратного распространения ошибки**

**Обратное распространение ошибки (Backpropagation)** - это метод обучения нейронных сетей, который заключается в распространении ошибки от выходов сети к ее входам. Этот процесс используется для коррекции весов в сети, с тем чтобы минимизировать ошибку предсказания модели.[8]

**Шаги обратного распространения ошибки:**

**1. Прямое распространение (Forward Propagation):**

На этапе прямого распространения входные данные передаются через нейроны сети от входного слоя к выходному. Каждый нейрон выполняет взвешенную сумму входных данных с использованием соответствующих весов и применяет активационную функцию.[9]

**2. Оценка ошибки:**

Разница между предсказанными значениями сети и фактическими значениями измеряется с использованием функции потерь. Функция потерь измеряет, насколько далеко предсказания сети отклоняются от истинных меток.

**3. Обратное распространение ошибки (Backward Propagation):**

На этом этапе градиент ошибки по отношению к весам сети распространяется в обратном направлении. Градиент - это частная производная функции потерь по весам. Используя правило цепочки, градиент вычисляется для каждого веса в сети.

**4. Обновление весов:**

С использованием градиента и метода оптимизации (например, градиентного спуска), веса сети обновляются с целью уменьшения функции потерь. Обновление весов происходит в направлении, противоположном градиенту.[10]

**5. Итерационный процесс:**

Процедура обратного распространения ошибки повторяется на нескольких эпохах обучения, пока сеть не достигнет удовлетворительной производительности. Каждая эпоха представляет собой один полный проход по всем обучающим данным.

**Преимущества обратного распространения ошибки:**

- Эффективность: Этот метод позволяет эффективно обучать глубокие нейронные сети с множеством весов.

- Гибкость: Применим к различным архитектурам сетей и задачам машинного обучения.

- Автоматизация: Веса обновляются автоматически, и нет необходимости вручную настраивать каждый параметр.[11]

**Проблемы обратного распространения ошибки:**

- Исчезающий градиент: В глубоких сетях градиент может становиться очень малым по мере распространения к началу сети.

- Переобучение: Возможность переобучения, особенно при наличии недостаточного количества данных или избыточной сложности модели.

Обратное распространение ошибки остается одним из основных методов обучения нейронных сетей и является фундаментом для различных архитектур и техник обучения в современном машинном обучении.[12]

**8. Прямое распространение:**

Принцип работы сети включает в себя прямое распространение сигнала от входного слоя через скрытые слои до выходного слоя. В результате этого процесса сеть делает прогноз или классификацию на основе входных данных.

Процесс обучения идет до тех пор, пока сеть не достигнет достаточной точности на обучающих данных. Когда сеть обучена, она способна делать прогнозы на новых, ранее не виденных данных. Этот принцип работы нейронной сети является основой для ее применения в различных задачах машинного обучения и искусственного интеллекта.[13]

**3. Свёрточные нейросети**

**3.1 О свёрточных нейронных сетях**

Свёрточные нейросети (Convolutional Neural Networks, CNNs) представляют собой мощный класс алгоритмов машинного обучения, разработанный специально для обработки данных с пространственной структурой, таких как изображения. Их создание было важным шагом в развитии компьютерного зрения и широко расширило возможности автоматического анализа и распознавания визуальной информации.[14]

**3.2 История и Эволюция**

**Свёрточные нейросети** впервые были представлены в работе **"Gradient-Based Learning Applied to Document Recognition"** Яна ЛеКуна, Леона Боту и Патрика Хаффера в 1998 году. Они были разработаны для задачи распознавания рукописных символов. С тех пор свёрточные нейросети претерпели значительные изменения и стали ключевым элементом во многих областях компьютерного зрения и не только.

**3.3** **Применение**

1. Распознавание изображений:

- Свёрточные нейросети успешно применяются в задачах классификации изображений, а также в детекции и сегментации объектов.

2. Обработка видео:

- В анализе видео CNNs используются для детекции движущихся объектов, распознавания действий и генерации описаний.

3. Медицинский анализ:

- В области медицинского изображения CNNs используются для диагностики, сегментации и анализа различных патологий.

4. Автономные транспортные средства:

- В разработке автономных транспортных средств CNNs применяются для обнаружения дорожных знаков, пешеходов и других объектов.

5. Генерация изображений:

- В задачах генерации изображений CNNs используются для создания искусственных изображений, стилизации и улучшения качества.[15]

**3.4 Особенности**

1. Свёрточные слои:

- Основная идея заключается в использовании свёрточных слоев для выделения локальных пространственных признаков в изображениях.



Рис.10 Стандартная нейронная сеть.



Рис.11 Свёрточная нейронная сеть.

2. Пулинг слои:

- Служат для уменьшения размерности представления и повышения инвариантности к масштабу и трансляциям.

3. Локальность и разделяемость параметров:

- CNNs используют локальные свёрточные операции, которые позволяют им улавливать пространственные зависимости. Параметры разделяются для уменьшения вычислительной сложности.

4. Передача обучения:

- CNNs часто используют предварительно обученные модели на крупных наборах данных (например, ImageNet), что повышает эффективность обучения на новых задачах.

**3.5 Вывод**

Свёрточные нейросети стали ключевым инструментом в обработке визуальной информации и обладают огромным потенциалом в решении различных задач. Их эволюция продолжается, и современные архитектуры, такие как ResNet, Inception и EfficientNet, продолжают улучшать результаты в разнообразных областях применения.[16]

1. **Заключение**

В данной курсовой работе мы рассмотрели эволюцию и принципы работы нейронных сетей, сфокусировав внимание на сверточных нейронных сетях (CNNs), предназначенных для обработки данных с пространственной структурой, таких как изображения. Исследование привело нас от первых шагов нейросетей, вдохновленных биологическим мозгом, до современных сверточных архитектур. Таким образом, все поставленные цели данной курсовой работы были достигнуты.

Аналогия между биологическими нейронами и искусственными нейронами создает основу для моделирования сложных процессов анализа, запоминания и воспроизведения информации. Нейронные сети находят применение в различных областях, включая обработку изображений, обработку естественного языка, медицинскую диагностику, финансовую аналитику и многие другие.

Свёрточные нейросети представляют собой ключевой инструмент в области компьютерного зрения, обеспечивая эффективное распознавание образов. Их история началась в конце 1990-х с задач распознавания рукописных символов и с тех пор привела к созданию мощных архитектур, таких как ResNet и EfficientNet.

Они применяются в различных сферах, начиная от распознавания изображений и обработки видео до медицинского анализа и автономных транспортных средств.

Современные тенденции в разработке нейросетей указывают на их постоянное развитие и улучшение производительности. Работы, такие как использование органоидов мозга для создания нового типа нейросетей, подчеркивают активные исследования в области совершенствования методов и инструментов машинного обучения.

Таким образом, нейронные сети играют решающую роль в преобразовании подходов к обработке информации и прогнозированию, открывая новые горизонты для применения в различных отраслях исследований и промышленности. С развитием новых методов и технологий, они продолжат привносить инновации и решать сложные задачи, что делает их ключевым элементом в работе многих людей в эпоху искусственного интеллекта.

1. **Библиографический список**
2. Сергей Дударов, История возникновения нейронных сетей. 2013 − //neuronus.com/history/5-istoriya-nejronnykh-setej.html
3. Сергей Николенко, Артур Кадурин, Екатерина Архангельская. «Глубокое обучение. Погружение в мир нейронных сетей.», СП: «Питер», 2018, 481с.
4. Артем Новиков, «Ученые использовали живую мозговую ткань для создания кибернейросети» - //[gazeta.ru/tech/news/2023/12/12/21909031](https://www.gazeta.ru/tech/news/2023/12/12/21909031.shtml), 2023.
5. Гудфеллоу Я., Бенджио И., Курвилль А. – «Глубокое обучение». М: «ДМК-Пресс», 2016, 652с.
6. Laurene V. Fausett “Fundamentals of Neural Networks: Architectures, Algorithms And Applications”, London: «Pearson», 1993, 480p.
7. Жерон Орельен. «Прикладное машинное обучение с помощью Scikit-Learn, Keras и TensorFlow. Концепции, инструменты.», М: «Диалектика», 2018, 1040с.
8. Франсуа Шолле. «Глубокое обучение на Python», СП: «Питер», 2018, 400c.
9. Юрий Черкасов - «Алгоритм обучения многослойной нейронной сети методом обратного распространения ошибки (Backpropagation)» - [//habr.com/ru/articles/198268/](https://habr.com/ru/articles/198268/), 2013.
10. Арнис, «Нейронные сети для начинающих. Часть1» - [//habr.com/ru/articles/312450/](https://habr.com/ru/articles/312450/), 2016.
11. Арнис, «Нейронные сети для начинающих. Часть2» - [//habr.com/ru/articles/313216/](https://habr.com/ru/articles/313216/), 2017.
12. Эндрю Траск. «Грокаем глубокое обучение», СП: «Питер», 2019, 352с.
13. Бурков Андрей. «Машинное обучение без лишних слов», СП: «Питер», 2020, 192с.
14. Sandro Skansi. “Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence”, NY: “Springer”, 2018, 204p.
15. Тарик Рашид. «Создаем нейронную сеть», М: «Вильямс», 2023, 272с.
16. Владимир Ростовцев. «Искусственные нейронные сети. Учебник», М: «Лань», 2019, 216с.
17. Андрей Шмиг, «CS231n: Свёрточные нейронные сети для распознавания образов» - [//habr.com/ru/articles/456186/](https://habr.com/ru/articles/456186/), 2019. (картинки вида нейронных сетей)

**ПРИЛОЖЕНИЕ 1.**

**Листинг программы, визуализация обучения модели и дополнительная информация**

**Информация о среде разработки, выбранном ЯП и сведения о программе.**

В работе использовался язык программирования **Python версии 3.11.0**

В качестве среды разработки была использована программа **PyCharm** от компании **JetBrains.**

Использованные в работе файлы:

* net\_algorithm.py
* saved\_net.py
* utility\_file.py

Использованные в работе библиотеки:

* Numpy
* MatPlotLib
* Pillow

Входными данными для программы являются:

1. Путь к файлу изображения.
2. Файл формата .npz с предобученной нейросетью

**Листинг файлов программы.**

net\_algorithm.py

import numpy as np  
import utility\_file  
  
""" Загружаем данные из файлов датасета, заранее приведя их в нужный нам вид"""  
images, labels = utility\_file.load\_dataset()  
  
""" Веса, сгенерированные рандомно с помощью numpy (в виде матрицы)"""  
weights\_input\_to\_hidden = np.random.uniform(-0.5, 0.5, (20, 784)) # Итого 784 нейрона на входном слое  
weights\_hidden\_to\_output = np.random.uniform(-0.5, 0.5, (10, 20)) # 20 на скрытом слое и 10 на выходном слое  
 # Так как у нас 10 возможных вариантов ответа  
""" Нейроны смещения, также сегенерированные с помощью numpy(в виде матрицы) """  
bias\_input\_to\_hidden = np.zeros((20, 1))  
bias\_hidden\_to\_output = np.zeros((10, 1))  
  
epochs = 5 # Количество эпох обучения  
e\_loss = 0 # Переменные для вычисления ошибки  
e\_correct = 0  
learning\_rate = 0.01 # Лучшая скорость обучения  
  
for epoch in range(epochs): # Обучаем нейросеть конкретное количесто эпох  
 print(f"Epoch №{epoch}")  
  
 for image, label in zip(images, labels):  
 image = np.reshape(image, (-1, 1))  
 label = np.reshape(label, (-1, 1))  
  
 # Передача данных от входного слоя на скрытый  
 # Forward propagation (прямое распространение)  
 # Складываем матрицы скрытого слоя и произведение весов на входе с данными изображений  
 hidden\_raw = bias\_input\_to\_hidden + weights\_input\_to\_hidden @ image  
 hidden = 1 / (1 + np.exp(-hidden\_raw)) # Сигмоида - функция активации  
  
 # Forward propagation от скрытого к выходному слою  
 output\_raw = bias\_hidden\_to\_output + weights\_hidden\_to\_output @ hidden  
 output = 1 / (1 + np.exp(-output\_raw))  
  
 # Вычисляем ошибку в выхлопе нейросети  
 e\_loss += 1 / len(output) \* np.sum((output - label) \*\* 2, axis=0)  
 e\_correct += int(np.argmax(output) == np.argmax(label))  
  
 # Обучение нейросети (алгоритм Backpropagation) на выходном слое -> к скрытому  
 delta\_output = output - label  
 weights\_hidden\_to\_output += -learning\_rate \* delta\_output @ np.transpose(hidden)  
 bias\_hidden\_to\_output += -learning\_rate \* delta\_output  
  
 # Обучение нейросети (алгоритм Backpropagation) на скрытом слое -> к входному  
 delta\_hidden = np.transpose(weights\_hidden\_to\_output) @ delta\_output \* (hidden \* (1 - hidden))  
 weights\_input\_to\_hidden += -learning\_rate \* delta\_hidden @ np.transpose(image)  
 bias\_input\_to\_hidden += -learning\_rate \* delta\_hidden  
  
 # Визуализация процесса обучения, мы видим ошибки и точность выраженные в процентах  
 print(f"Loss: {round((e\_loss[0] / images.shape[0]) \* 100, 3)}%")  
 print(f"Accuracy: {round((e\_correct / images.shape[0]) \* 100, 3)}%")  
 e\_loss = 0  
 e\_correct = 0  
  
# Сохранение параметров обученной нейросети в один файл  
np.savez('neural\_network\_params\_5.npz',  
 weights\_input\_to\_hidden=weights\_input\_to\_hidden,  
 weights\_hidden\_to\_output=weights\_hidden\_to\_output,  
 bias\_input\_to\_hidden=bias\_input\_to\_hidden,  
 bias\_hidden\_to\_output=bias\_hidden\_to\_output)

saved\_net.py

import numpy as np  
import matplotlib.pyplot as plt  
import utility\_file  
from utility\_file import image\_converter  
  
classes = ['футболка', 'брюки', 'свитер', 'платье', 'пальто',  
 'туфли', 'рубашка', 'кроссовки', 'сумка', 'ботинки']  
  
""" Загружаем данные из файлов датасета, заранее приведя их в нужный нам вид"""  
images, labels = utility\_file.load\_dataset\_test()  
  
# Загрузка параметров из файла  
loaded\_params = np.load('neural\_network\_params\_5.npz')  
weights\_input\_to\_hidden = loaded\_params['weights\_input\_to\_hidden']  
weights\_hidden\_to\_output = loaded\_params['weights\_hidden\_to\_output']  
bias\_input\_to\_hidden = loaded\_params['bias\_input\_to\_hidden']  
bias\_hidden\_to\_output = loaded\_params['bias\_hidden\_to\_output']  
  
path = 'shirt.jpg'  
test\_image = image\_converter(path)  
image = np.reshape(test\_image, (-1, 1))  
  
# Передаем данные об изображении на вход нейросети  
# Forward propagation от входного слоя к скрытому слою  
hidden\_raw = bias\_input\_to\_hidden + weights\_input\_to\_hidden @ image  
hidden = 1 / (1 + np.exp(-hidden\_raw)) # Сигмоида - функция активации  
  
# Forward propagation от скрытого к выходному слою  
output\_raw = bias\_hidden\_to\_output + weights\_hidden\_to\_output @ hidden  
output = 1 / (1 + np.exp(-output\_raw))  
  
plt.imshow(test\_image.reshape(28, 28), cmap="Greys")  
plt.title(f"Предполагаемый нейросетью объект: {classes[output.argmax()]}")  
plt.show()

utility\_file.py

import numpy as np  
from PIL import Image

def load\_dataset():  
 image\_path = 'datasets/train-images-idx3-ubyte'  
  
 with open(image\_path, 'rb') as f:  
 # Чтение магического числа и количества изображений  
 magic\_number = int.from\_bytes(f.read(4), 'big')  
 num\_images = int.from\_bytes(f.read(4), 'big')  
 num\_rows = int.from\_bytes(f.read(4), 'big')  
 num\_cols = int.from\_bytes(f.read(4), 'big')  
  
 # Чтение пиксельных значений изображений  
 image\_data = np.frombuffer(f.read(), dtype=np.uint8)  
 # Конвертация данных изображений в матрицу с размерностью (60000, 784)  
 # Таким образом у нас на входном слое будет 784 нейрона  
 x\_train = image\_data.reshape(60000, 784)  
 x\_train = x\_train.astype('float32') / 255  
  
 labels\_path = 'datasets/train-labels-idx1-ubyte'  
  
 with open(labels\_path, 'rb') as f:  
 # Чтение магического числа и количества меток  
 l\_magic\_number = int.from\_bytes(f.read(4), 'big')  
 l\_num\_images = int.from\_bytes(f.read(4), 'big')  
   
 # Чтение и конвертация значений меток в матрицу с размерностью (60000,10), т.к. у нас 10 возможных значений меток  
 label\_data = np.frombuffer(f.read(), dtype=np.uint8)  
 y\_train = label\_data  
 y\_train = np.eye(10)[y\_train]  
 return x\_train, y\_train  
  
def load\_dataset\_test():  
 image\_path = 't10k-images-idx3-ubyte'  
  
 with open(image\_path, 'rb') as f:  
 # Чтение магического числа и количества изображений  
 magic\_number = int.from\_bytes(f.read(4), 'big')  
 num\_images = int.from\_bytes(f.read(4), 'big')  
 num\_rows = int.from\_bytes(f.read(4), 'big')  
 num\_cols = int.from\_bytes(f.read(4), 'big')  
  
 # Чтение пиксельных значений изображений  
 image\_data = np.frombuffer(f.read(), dtype=np.uint8)  
 # Конвертация данных изображений в матрицу с размерностью (60000, 784)  
 # Таким образом у нас на входном слое будет 784 нейрона  
 x\_train = image\_data.reshape(10000, 784)  
 x\_train = x\_train.astype('float32') / 255  
  
 labels\_path = 't10k-labels-idx1-ubyte'  
  
 with open(labels\_path, 'rb') as f:  
 # Чтение магического числа и количества меток  
 l\_magic\_number = int.from\_bytes(f.read(4), 'big')  
 l\_num\_images = int.from\_bytes(f.read(4), 'big')  
  
 # Чтение и конвертация значений меток в матрицу с размерностью (60000,10), т.к. у нас 10 возможных значений меток  
 label\_data = np.frombuffer(f.read(), dtype=np.uint8)  
 y\_train = label\_data   
 y\_train = np.eye(10)[y\_train]  
  
 return x\_train, y\_train  
  
def image\_converter(path):  
 image\_path = 'Путь\_к\_файлу.jpg'  
 orig\_image = Image.open(path)  
 resized\_image = orig\_image.resize((28, 28))  
 gray\_image = resized\_image.convert('L')  
 inverted\_image = Image.eval(gray\_image, lambda x: 255 - x)  
 image\_array = np.array(inverted\_image)  
 normalized\_image = image\_array / 255.0  
 flattened\_image = normalized\_image.flatten()  
 return flattened\_image

**Визуализация процесса обучения модели нейронной сети.**

Ниже представлен пример обучения модели без применения алгоритма обратного распространения ошибки (Backpropagation) и вместе с ним. Посмотрев на выходные данные, мы можем сделать вывод, что нейросеть не обучается на предоставленных ей данных, так как не происходит «подгонки» весов.

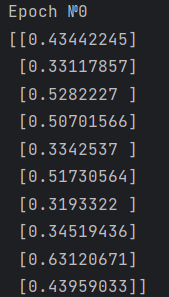
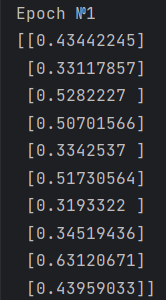
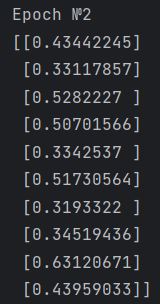
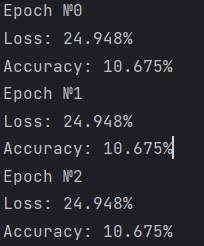
   

Рис. 12-16 – Процесс «необучения» модели при смене эпох.

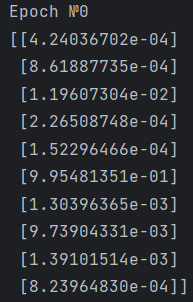
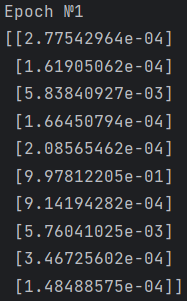
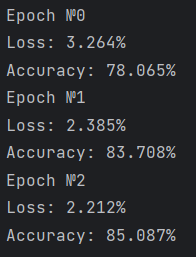
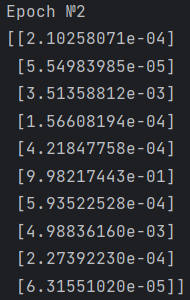
  

Рис. 17-20 – Процесс обучения модели при смене эпох.

При добавлении алгоритма обратного распространения ошибки, мы можем видеть, как меняются выходные данные в массивах. Также, визуализированы количество потерь и точность выходных данных в процентах.

**Процесс работы с обученной нейросетью.**

Записываем в переменную “path”, содержащуюся в файле saved\_net.py, имя файла, который мы хотим проанализировать. В данном случае это файл, содержащий изображение синей футболки.



Рис. 21 – файл “Blue\_Tshirt.jpg”.

Затем код в файле utility\_file.py преобразует данное изображение в формат, понятный для нейросети. Далее, обученная нейросеть идентифицирует изображение как футболку:



Рис.22 – Предполагаемый нейросетью объект.