МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**КУРСОВАЯ РАБОТА**

**ВЕБ-СЕРВИС РАСПОЗНАВАНИЯ И РЕШЕНИЯ МАТЕМАТИЧЕСКИХ ПРИМЕРОВ НЕКОТОРОЙ ОБЛАСТИ**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.С. Прокопенко

(подпись, дата)

Направление подготовки 02.03.03 курс 3

Направленность (профиль) Технология программирования

Научный руководитель

канд. пед. наук, доц. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.Ю. Добровольская

(подпись, дата)

Нормоконтролер

ст. преп. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.В. Харченко

(подпись, дата)

Краснодар

2022

**РЕФЕРАТ**

Курсовая работа 36 страниц, 26 рисунков, 7 таблиц, 11 источников.

НЕЙРОННЫЕ СЕТИ, РАСПОЗНАВАНИЕ, LATEX, РАСПОЗНАВАНИЕ ТЕКСТА, РАСПОЗНАВАНИЕ МЕТЕМАТИЧЕСКИХ СИМВОЛОВ, РЕШЕНИЕ ИНТЕГРАЛОВ, ЦИФРОВИЗАЦИЯ ОБРАЗОВАНИЯ, IT-ТЕХНОЛОГИИ В ОБРАЗОВАНИИ

Цель курсовой работы – реализовать веб-сервис, распознающий математическую задачу и решающий ее.

В курсовой работе рассмотрены принципы построения нейронных сетей, изучены их архитектуры области применения, а также направления применения цифр технологий в образовании, проведен реферативный обзор приложений, позволяющих распознавать печатные и рукописные математические примеры и решать их.

В практической части курсовой работы предложен веб-сервис, при помощи HTML, CCS3, JavaScript, jQuery. Приложение решает задачу распознавания математический примеров, заданных в рукописной форме, и выдает Latex код.

**СОДЕРЖАНИЕ**

[Введение 4](#_Toc103293506)

[1 Механизмы распознавания 5](#_Toc103293507)

[1.1 Персептрон. Нейронные сети Гроссберга. 6](#_Toc103293508)

[1.2 Сверточная нейронная сеть 13](#_Toc103293509)

[1.3 Существующие решения нейронных сетей 15](#_Toc103293510)

[2 Анализ предметной области 17](#_Toc103293511)

[2.1 Реферативный обзор существующих приложений 17](#_Toc103293512)

[3 Используемый инструментарий и функционал приложения 23](#_Toc103293513)

[3.1 Блок распознавания 23](#_Toc103293514)

[3.2 Блок создания веб-сервиса 24](#_Toc103293515)

[3.3 Метод Симпсона 25](#_Toc103293516)

[3.4 Структура работы приложения 26](#_Toc103293517)

[3.5 Эксперимент 28](#_Toc103293518)

[Заключение 35](#_Toc103293519)

[Список использованных источников 36](#_Toc103293520)

# **ВВЕДЕНИЕ**

Цифровизация образования предполагает повсеместное внедрение IT-технологий как в собственном обучении, так и в организацию учебного процесса. Одной из проблем автоматизации математического обучения является цифровизация рукописных математических символов. Для решения этой проблемы можно использовать нейросетевые технологии, хорошо зарекомендовавших себя в распознавании образов

Нейронная сеть – это тип машинного обучения, который имитирует процессы обучения человеческого мозга. Управляемая мощными алгоритмами нейронная сеть выявляет шаблоны в обширных наборах данных, позволяя делать более точные обобщения о том, что мы изучаем, например, о почерке.

Нейронные сети строятся из математических моделей, «обученных» идентифицировать шаблоны в соответствии с заранее определенными переменными или «размерами». Тщательно запрограммированные алгоритмы многократно разбивают и сортируют данные в соответствии с этими измерениями, классифицируя и переклассифицируя до тех пор, пока не появятся четкие закономерности.

Применение нейросетевых технологий при распознавании математическим примеров позволит автоматизировать их решение, получать цифровой формат для дальнейшей обработки. Реализация алгоритма распознавания необходимо трансформировать удобную для пользователя форму, например веб-сервис.

1. **Механизмы распознавания**

Нейронная сеть для распознавания образов – это наиболее популярный способ применения нейронной сети. При этом вне зависимости от особенностей решаемых задач, она работает по этапам.

В качестве распознаваемых образов могут выступать самые разные объекты, включая изображения, рукописный или печатный текст, звуки и многое другое. При обучении сети ей предлагаются различные образцы с меткой того, к какому именно типу их можно отнести. В качестве образца применяется вектор значений признаков, а совокупность признаков в этих условиях должна позволить однозначно определить, с каким классом образов имеет дело нейронная сеть.

Важно при обучении научить сеть определять не только достаточное количество и значения признаков, чтобы выдавать хорошую точность на новых образах, но и не переобучиться, то есть, излишне не «подстроиться» под обучающую выборку из изображений. После завершения правильного обучения нейронная сеть должна уметь определять образы (тех же классов), с которыми она не имела дела в процессе обучения.

Важно учитывать, что исходные данные для нейросети должны быть однозначны и непротиворечивы, чтобы не возникали ситуации, когда НС будет выдавать высокие вероятности принадлежности одного объекта к нескольким классам.

В целом создание нейронной сети для распознавания образов показано на [рисунке 1](#рис1)



Рисунок 1 – Построение нейронной сети

Рассмотрим различные архитекторы нейронных сетей и применим их в своем приложении.

1. **Персептрон. Нейронные сети Гроссберга**

В распознавании образов широко применяются искусственные нейронные сети. Нейронная сеть прямого распространения и конкурирующие нейроны могут использоваться для распознавания текстовых символов, однако при распознавании зашумленных образов (текстовых символов), используются такие нейронные сети, как Хемминга или Гроссберга. Наибольший эффект, обусловленный пластичностью, дают сети Гроссберга.

Процесс обучения сходится в зависимости от значения параметра сходства. Кроме этого, зашумленность данных может привести к ошибочным результатам, так как используемая мера сходства в ряде случаев приводит к отнесению образа не к соответствующему классу. Норма двоичного вектора в зависимости от степени зашумленности может принимать одно и тоже значение при различных зашумлениях. Наиболее просто обучение распознаванию текстовых символов осуществляется нейронными сетями прямого распространения, однако, подобные связи малоэффективны при распознавании зашумленных символов. Так как сеть прямого распространения является наиболее простой, то задача распознавания зашумленных текстовых символов этой сетью является актуальной.

Основная задача есть повышение эффективности распознавания искаженных текстовых символов за счет сегментации признакового пространства представления изображения. Для достижения цели в каждом из сегментов разбиения плоскости существования изображения ставится в соответствие значение нуля или единицы, в зависимости от наличия в нем следа изображения. При этом след изображения предлагается отождествлять с одной из базисных функций, задаваемых положением прямой в сегменте. Для этого каждый сегмент разбивается на подсегменты таким образом, чтобы при кодировании подсегмента двоичным вектором, искажение компонент не приводило бы уменьшению значений компонент, отмечающих сегмент. В качестве такой меры выбирается порог суммы компонент, отмечаемых подсегменты соответствующих сегментов. Для определения бинарных значений подсегментов необходимо определить всевозможные (характеры) функции отображения символа в сегменте посредством бинарных компонент, отмечающих подсегменты. При распознавании текстовых символов такими функциями на плоскости являются: прямая параллельная оси абсцисс, прямая параллельная оси ординат, прямая с положительным наклоном относительно оси абсцисс и прямая с отрицательным наклоном относительно оси абсцисс.

Если для обучения персептрона распознавания символа текста область задания символа разбивалась на квадраты, нумеруемые целыми положительными числами, то для обучения персептрона распознавания зашумленных символов каждый квадрат разбивается на поля, аналогично разбиению. В результате для квадрата получаем упорядоченные поля , для ([таблица 1](#табл1)).

Таблица 1 – Область задания символа

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

В каждом из этих полей для текстовых символов может располагаться одна из приведенных ранее функций, либо поле может быть пустым. Выбор длины двоичных векторов для кодирования области существования символов определяется количеством и топологией символа и для текстовых символов определяется, как 5x5. Такой же размер может быть принят и для полей внутри квадрата. Как и в разбиении на сегменты, 0 – округляет отсутствие следа изображения, 1 – округляет наличие следа изображений при бинарном кодировании ([таблица 2](#табл2)).

Таблица 2 – Пустое поле

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |

Таблица 3 – Поле прямой параллельной оси абсцисс

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |

Таблица 4 – Поле прямой параллельной оси ординат

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 |

Таблица 5 – Поле прямой с положительным углом наклона

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |

Таблица 6 – Поле прямой с отрицательным углом наклона

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 |

Тогда нейронная сеть состоит из 5 нейронов ([рисунок 2](#рис2)) с бинарной функцией активации может одним из известных методов обучена кластеризации этих функций в каждом из сегментов. При выборе порога каждого нейрона выходной сигнал принимает единичное значение при достижении порога при весовых коэффициентах , что соответствует .

Тогда сеть из 5 нейронов будем обозначать символом и причем , если

, если двоичный вектор определяется линейным кодом, соответствующим массиву ([таблица 3](#табл3));

, если двоичный вектор соответствует массиву ([таблица 4](#табл4));

, если двоичный вектор соответствует массиву ([таблица 5](#табл5));

, если двоичный вектор соответствует массиву ([таблица 6](#табл6));

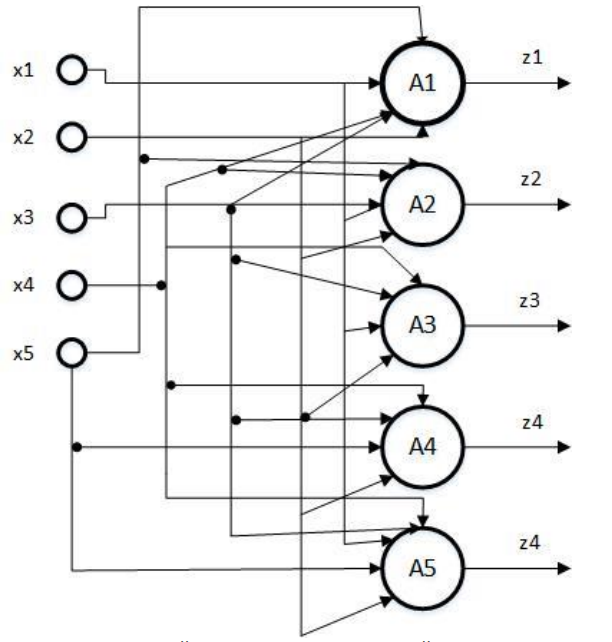


Рисунок 2 – Нейронная сеть из пяти нейронов

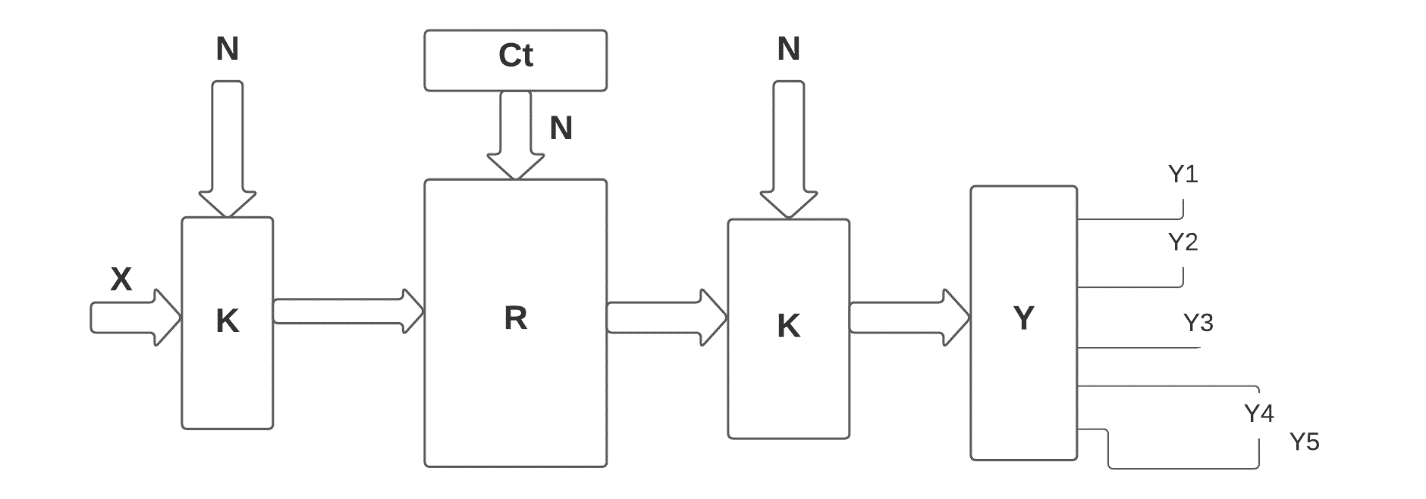


Рисунок 3 – Нейронная сеть

Сеть включает два слоя нейронов и два коммутатора , на управляющие входы которых поступают коды, задающие последовательность сегментов разбиения области задания изображения. Слой нейронов включает фиксаторы выходов слоя и нейронов по числу распознаваемых текстовых символов ([рисунок 3](#рис3)).

Слой фиксаторов представляет группы, каждая из которых содержит 4 за понимающих элемента, в которые заносятся единицы, отличающие соответствующего из 4 функций, изображение которой присутствует в выбранном сегменте, задаваемом кодом , формируемым счетчиком .

Слой выходных нейронов формирует единичный сигнал на выходе, определяемом истинностью логического , где – единичные выходы соответствующих фиксаторов выходов слоя .

Слой нейронов представляет сеть . Вход представляет битовую последовательность пятикомпонентного вектора. Длина последовательности определяется числом сегментов разбиения области задания изображения.

Сеть может не содержать коммутаторов, тогда слои и будут содержать группы нейронов по числу характерных функций для слоя в количестве, определяемом числом сегментов разбиения, а в слое количество нейронов определяется числом распознаваемых символов.

Допустимые искажения вектора округляются числом битов, искажения которых не приводят к переводу характеристической функции в .

Алгоритм обучения сети состоит в выделении в соответствии с задаваемым двоичным вектором в сегменте заданных характеристических функций, например, по правилу Хебба. Выделенные характеристики функции служат входами слоя , выходы которого задаются булевыми функциями входных переменных в виде таблицы истинности. Обучение слоя может быть проведено либо по правилу Хебба, либо с использованием дельта правила. При любом способе обучения сеть детерминирована, и каждому входному изображению соответствует единственный выход с сигналом равным единице.

При обучении слоя проводится модификация входного слоя для обеспечения устойчивости к искажению компонент вектора. Для чего вводится параметр сходства , который позволяет отнести искаженный входной вектор к идеальному. Если определить количество бит, искажения которых могут не учитываться при формировании изображений, то в этом случае, сеть, обученная при идеальных входных сигналах, будет давать правильную классификацию образов при допустимых искажениях.

* 1. **Сверточная нейронная сеть**

Сверточные нейронные сети, как архитектура искусственных нейронных сетей, были предложены Яном Лекуном (Yan LeCun) в 1988 году. Эта архитектура направлена на эффективное распознавание изображений. Пример сверточной нейронной сети представлен на [рисунок 4](#рис4).

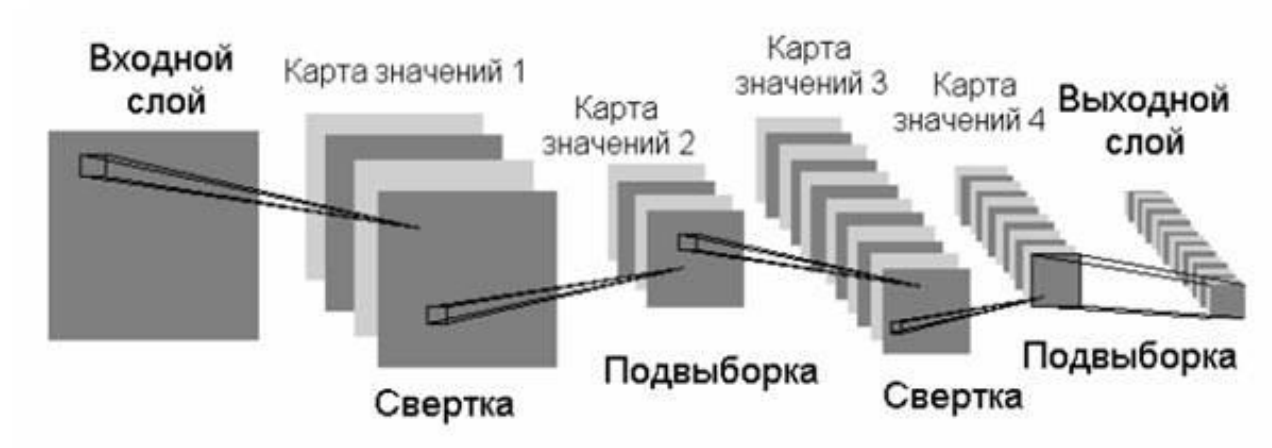


Рисунок 4 – Пример сверточной нейронной сети

Основной идеей является чередование слоев свертки и слоев подвыборки. За счет того, что учитывается двумерность изображения, сверточные нейронные сети показывают себя лучше на распознавании изображений, чем другие виды искусственных нейронных сетей.

Алгоритм работы слоя свертки заключается в следующем, каждый фрагмент исходного изображения умножается на матрицу (ядро) свертки, полученные результаты суммируются и записываются в исходное положение, но уже на другом изображении. Этот процесс изображен на [рисунок 6](#рис6).

Также, помимо слоев свертки, используются слои подвыборки. Такой слой позволяет уменьшить размерность изображения. Считается, что наличие признака, важнее его положения. Суть метода заключается в том, что группа пикселей (например, 2 на 2) сжимается до одного. Обычно для этого используется функция максимума. Пример работы слоя подвыборки представлен на [рисунок 5](#рис5).

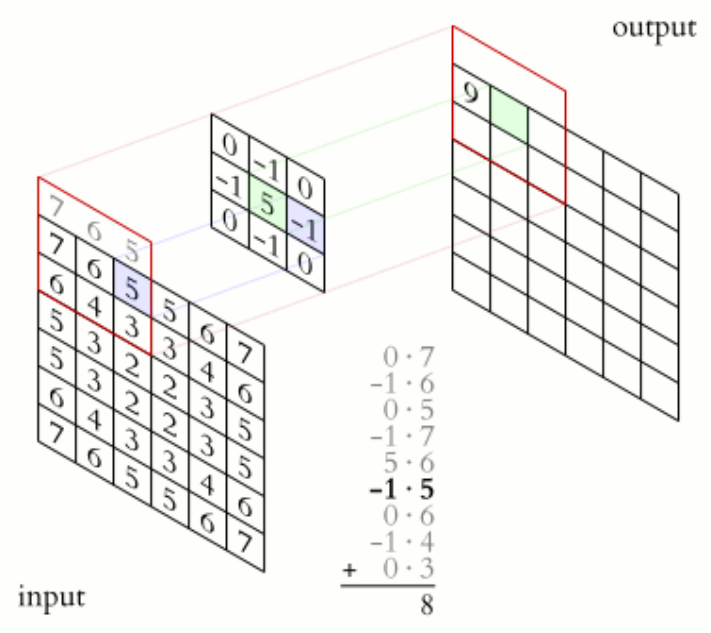


Рисунок 5 – Пример свертки изображения

После прохождения нескольких слоев свертки и подвыборки, количество каналов увеличивается, а размерности изображений уменьшаются. На этом этапе происходит переход к полносвязным слоям. В результате последний слой имеет размерность, соответствующую числу классов.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 6 – Пример работы слоя подвыборки с функцией максимума

* 1. **Существующие решения нейронных сетей**

Первая сверточная нейронная сеть для распознавания символов была представлена в 1998 году французским исследователем Яном Лекуном (Yann LeCun). Называется она LeNet. Структура данной сети представлена на [рисунок 7.](#рис7)

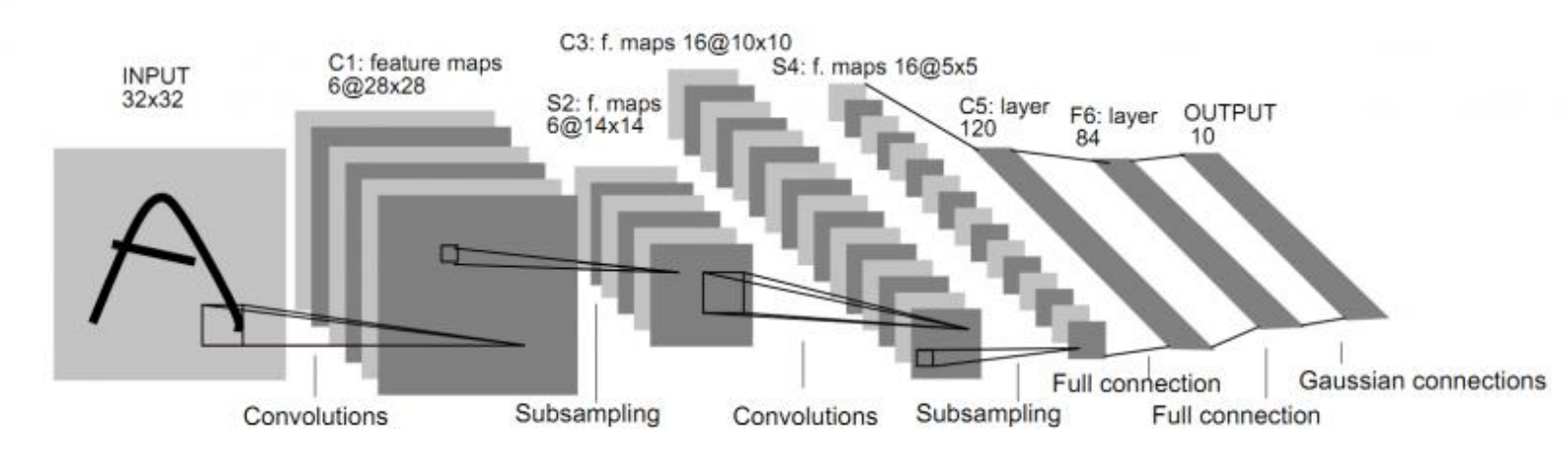


Рисунок 7 – Структура нейронной сети LeNet

Для языка программирования Python существует большое количество готовых библиотек для работы с нейронными сетями. Использование этих библиотек призвано в значительной степени упростить задачу разработки нейронных сетей. Далее в этом разделе будут рассмотрены наиболее популярные из них.

* TensorFlow

Очень популярная библиотека, разрабатываемая Google. Документация для данной библиотеки хорошо проработана. Также очень много статей в сети Интернет и примеров нейронных сетей для данной библиотеки.

Для представления нейронной сети здесь используются графы, а для хранения данных многомерные массивы – тензоры.

* Theano

Другая не менее популярная библиотека для Python. Библиотека получила свое название в честь имени жены древнегреческого философа и математика Пифагора – Феано (или Теано). Так же, как и других библиотеках, имеется интеграция с библиотекой NumPy. Присутствует возможность проведения вычислений как на CPU, так и на GPU.

* Keras

Библиотека, а если быть точнее, фреймворк для создания нейронных сетей. Keras, является надстройкой над TensorFlow и Theano и может использовать любую из этих двух библиотек для проведения вычислений. Из всех библиотек эта наиболее удобна и проста для понимания. Новые слои здесь добавляются лишь с использованием одной функции. Хорошая документация и множество готовых примеров – все это является большим преимуществом для Keras.

Благодаря полученным знаниям о нейронных сетях и различных решений, реализуем веб-сервис, который распознает и решает математические примеры.

1. **Анализ предметной области**

Распознавание математических выражений является более сложной задачей распознавания, чем распознавание символов, так как кроме распознавания непосредственно математического символа необходимо также распознать структуру математического выражения.

В научной литературе разделяют два вида распознавания рукописных символов: онлайн и оффлайн.

Онлайн распознавание подразумевает непосредственное рисование формулы на компьютере в режиме реального времени. При этом ведется сбор дополнительной информации в виде времени и координат каждого штриха.

Оффлайн распознавание подразумевает отсутствие информации о ходе написания символа, дано только изображение, содержащее формулу, на котором формулу требуется найти и распознать, то есть выделить символы на фоне, разбить на строки и т. д.

В данной работе будет рассматриваться задача онлайн распознавания.

1. **Реферативный обзор существующих приложений**

Рассмотрим предметную область, представленную некоторыми разработками, определим полезные свойства в каждом приложении и применим их в своем алгоритме.

* Приложение Photomath

Photomath полезный и наглядный инструмент для решения математических уравнений, в основе которого лежит технология распознавания данных (OCR-технология), поступающих с камеры мобильного устройства.

Для того чтобы рассчитать уравнение необходимо навести камеру смартфона на задание, выбрать границы уравнения (можно откорректировать вручную), сфотографировать пример и подождать, пока на экране появится решение ([рисунок 8](#рис8)). Приложение умеет сканировать задачи как с бумаги, так и с экрана, но во втором случае ему может потребоваться больше времени на обработку. Однако рукописный текст не всегда точно распознает выражения.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 8 – Пример работы приложения Photomath

Образовательная часть приложения заключается в том, что приложение выдает не только правильный ответ, но и позволяет просмотреть все этапы решения математического уравнения. Ко всем решенным с помощью приложения примерам можно вернуться в любой момент, поскольку они автоматически складируются в специальном разделе.

Основные задачи данного приложения заключаются в получении быстрого ответа и поэтапного решения математического примера.

* Веб-сервис Detexify

Данный веб-сервис позволяет рисовать символ онлайн, а затем распознает его и предоставляет данные наиболее вероятных символов. Пример работы сервиса представлен ниже ([рисунок 9](#рис9)).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 9 – Пример работы веб-сервиса Detexify

Стоит отметить, что данный сервис отлично справляется со своей задачей. Также его исходный код полностью открыт и выложен на GitHub.

Недостатком данного сервиса является то, что производится распознавание только одного символа. Также значительным минусом является невозможность загрузить свое изображение.

* Приложение MyScript Calculator

MyScript Calculator – это программное решение, которое позволяет использовать мобильное устройство в качестве кусочка бумаги для разных вычислений. Особенность MyScript Calculator заключается в том, что приложение работает только с рукописным вводом данных. Здесь даже отсутствуют кнопки, а все, что имеется — это чистое полотно на весь экран, имитирующее бумагу-миллиметровку. Условие математической задачи для вычисления пользователь пишет пальцем или с помощью стилуса.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 10 – Пример работы приложения MyScript Calculator

Система автоматически распознает формулу на полотне, а также проводит ее вычисление ([рисунок 10](#рис10)). Умеет решать лишь простые примеры, в одну строчку, без интегралов и т. д.

* Приложение Mathpix

Mathpix – приложение, которое позволяет решать и визуализировать решения. Здесь используется программное обеспечение MathpixOCR, которое распознает напечатанные или написанные от руки файлы документа STEM с изображения или файла PDF. На выходе система выдает LaTeX-код.

Mathpix быстро сканирует и считывает условие задачи. Тут же отправляет на сервер данные условия и почти мгновенно выдает ответ. При этом пользователю доступны инструменты для работы с задачами в режиме графика: редактирование вводных данных, добавление таблиц, заметок и дополнительных функций для нескольких графиков ([рисунок 11](#рис11)).

Однако приложение имеет сложный и непонятный интерфейс, который не позволяет новым пользователям начать быстро использовать калькулятор.

Изображение выглядит как текст, доска

Автоматически созданное описание

Рисунок 11 – Пример работы приложения Mathpix

* Мобильное приложение и веб-сервис Mathway

Программное решение Mathway является своеобразным инструментом для решения задач, который, помимо школьного курса математики, охватывает математический анализ, статистику, тригонометрии и линейную алгебру. Mathway предоставляет пользователю наиболее развернутые пошаговые решения задач, подкрепленные торической частью.

Mathway, как и другие рассмотренные приложения решает задачи с помощью камеры устройства ([рисунок 12](#рис12)), однако эта функция плохо реализована. Интерфейс камеры в программе крайне минималистичный, так как отсутствует области распознавания. Камера захватывает только часть приложения, отсюда и ошибки. К тому же камера автоматически настроена на макро-режим, поэтому алгоритмы распознавания часто плохо срабатывают и выдают неправильный ответ.

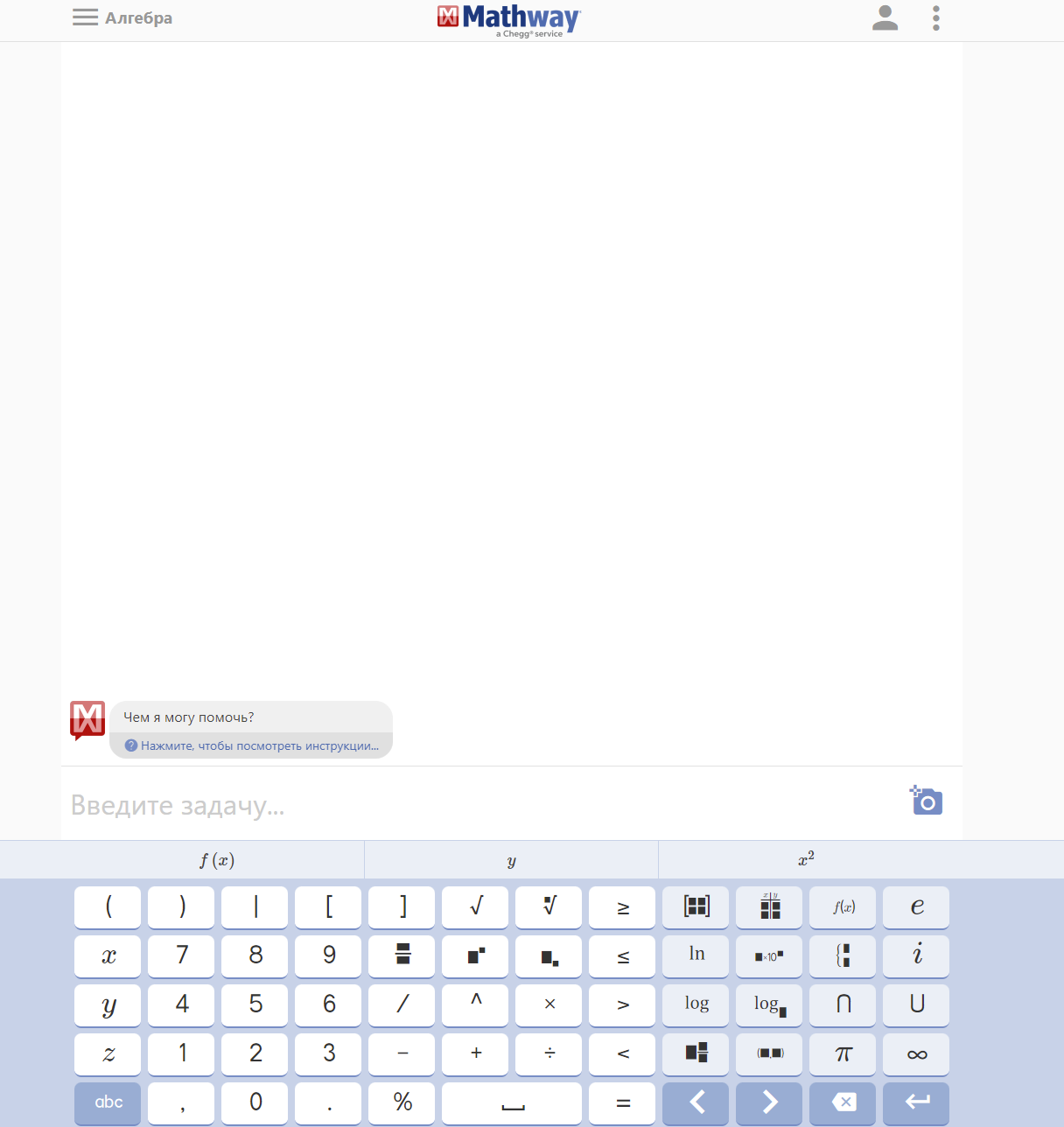


Рисунок 12 – Интерфейс веб-сервиса Mathway

Для задачи распознавания рукописных математических формул уже существует ряд подходов и их реализации. Однако большинство из них имеет существенные недостатки. Например, не позволяют загрузить собственное изображение и не предоставляют удобный графический интерфейс для работы. Также, большая часть этих инструментов не имеет удобного вывода результата в различные форматы (LaTeX и т. п.). Необходимо будет учесть эти недостатки при разработке собственного приложения.

1. **Используемый инструментарий и функционал приложения**
   1. **Блок распознавания**

Система распознавания рукописных математических формул будет представлять собой веб-сервис. С помощью этого сервиса у пользователя будет возможность изобразить на полотне математическую задачу, получить готовый результат в формате LaTeX, а также решение примера.

При разработке модуля распознавания использовалась библиотека JavaScript iinkJS, которая содержит базу данных рукописных цифр MNIST, которая включате обучающий набор из 60 000 примеров и тестовый набор из 10 000 примеров. Цифры были нормализованы по размеру и центрированы на изображении фиксированного размера. Пример работы iinks представлен на [рисунке 13](#рис13).

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 13 – Распознавание iinkJS

* 1. **Блок создания веб-сервиса**

Интерфейс веб-сервиса должен быть простым и понятным, но при этом должен полностью обеспечивать необходимую функциональность. Интерфейс веб-сервиса состоит из 3 основных частей:

* шапка сайта, которая содержит распознанный математический пример.
* основная секция, где находятся основные кнопки для работы с приложением, результат Latex распознавания и полотно-редактор.
* подвал, который содержит информацию об авторе.

Макет графического интерфейса пользователя изображен на [рисунке 14](#рис14).



Рисунок 14 – Макет графического интерфейса

* 1. **Метод Симпсона**

Для решения интегралов используется Метод Симпсона, который относится к приёмам численного интегрирования. Суть метода заключается в приближении подынтегральной функции на отрезке интерполяционным многочленом второй степени , то есть приближение графика функции на отрезке параболой ([рисунок 15](#рис15)). Метод Симпсона имеет порядок погрешности 4 и алгебраический порядок точности 3.

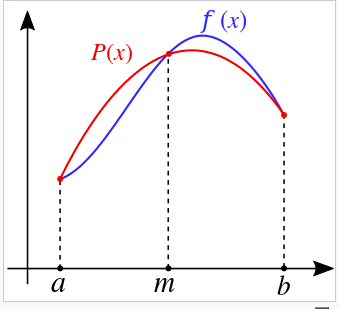


Рисунок 15 – Суть метода, аппроксимация функции (синий график) квадратичным полиномом (красный)

Формула Симпсона:

где ­ значения функции в соответствующих точках (на концах отрезка и в его середине).

Фрагмент кода метода Симпсона в программе представлен на [рисунке 16.](#рис16)

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 16 – Метод Симпсона в программе.

* 1. **Структура работы приложения**

Основная задача нашего алгоритма – выдать правильное решение математической задачи, посредством методов, взятых с математического анализа. Алгоритм хранит в себе как итерационные способы решения, так и прямые. Программа находит и выдает решение пользователю на экране.

Сперва пользователю предоставляется полотно, где нужно изобразить условие математической задачи. Одновременно с написанием формулы на полотне формируется Latex-код данного примера, который будет отображаться в специальном поле. Далее алгоритм обрабатывает полученные значения в формате Latex и записывает их в переменные. Затем интеграл решается Методом Симпсона с точностью проверки 0,01. Результат работы приложения уже хранится в скрытом блоке и пользователю остается только запросить ответ. Однако, если пользователю не нужно решение, а просто нужно получить Latex-код формулы, ему просто достаточно нажать на блок, где расположена формула и формат Latex будет скопирован в буфер обмена автоматически.

Структура работы приложения представлена на [рисунке 17](#рис17).

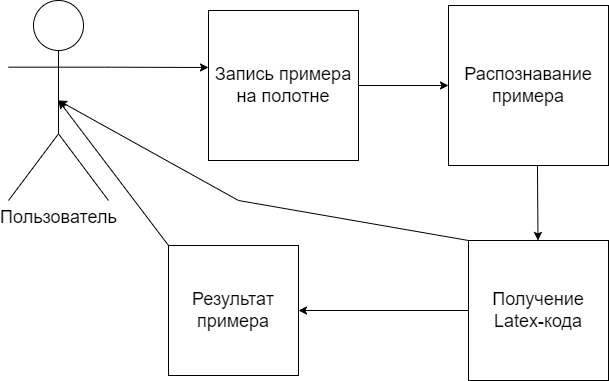


Рисунок 17 – Структура работы приложения

Реализация программной части для распознавания математических примеров и их решения велась в среде разработки Visual Studio Code 1.67 и на операционной системе Windows 10 Home.

Для реализации веб составляющей были использованы HTML, CCS3, JavaScript, jQuery 3.x.

Интерфейс приложения, согласно макету, рисунок 14, изображен ниже ([рисунок 18](#рис18)).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 18 – Интерфейс приложения

* 1. **Эксперимент**

В рамках работы был проведен эксперимент по качеству распознавания математического примера и решения.

Опыт 1

На вход программы подается корректно изображённый определенный интеграл ([рисунок 19](#рис19)).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 19 – Опыт 1

Алгоритм успешно распознал математический пример, выдал правильный Latex-код и верный результат ([рисунок 20](#рис20)).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 20 – Результат опыта 1

Опыт 2

На вход программы подается корректно изображённый определенный интеграл ([рисунок 21](#рис21)).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 21 – Опыт 2

Алгоритм успешно распознал математический пример, выдал правильный Latex-код и верный результат ([рисунок 22](#рис22)).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 22 – Результат опыта 2

Опыт 3

На вход программы подается некорректно изображенный определенный интеграл ([рисунок 23](#рис23)).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 23 – Результат опыта 3

Программа неправильно распознала символы и не смогла определить Latex формат, следовательно, решение примера не определенно.

Опыт 4

На вход программы подается корректно изображённый определенный интеграл ([рисунок 24](#рис24)).

Изображение выглядит как текст

Автоматически созданное описаниеРисунок 24 – Опыт 4

Алгоритм успешно распознал математический пример, выдал правильный Latex-код и верный результат ([рисунок 25](#рис25)).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 25 – Результат опыта 4

Опыт 5

На вход программы подается корректно изображённый определенный интеграл ([рисунок 26](#рис26)).

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 26 – Результат опыта 4

Алгоритм неправильно распознал математический пример, поскольку символ «1» имеет другое представление в базе рукописных цифр. Соответственно, Latex-код имеет неправильную запись, и программа выдает неправильно решение.

Итоги опытов приведены ниже ([таблица 7](#табл7)).

Таблица 7 – Итоги эксперимента

|  |  |  |
| --- | --- | --- |
| № Опыта | Входные данные | Результат |
| 1 | Корректное изображение | распознано |
| 2 | Корректное изображение | распознано |
| 3 | Некорректное изображение | не распознано |
| 4 | Корректное изображение | распознано |
| 5 | Корректное изображение | не распознано |

В ходе проведенных опытов, заметим, что программа не всегда точно распознает математический пример, зачастую верный результат можно получить, только с корректной изображенной задачи.

# **ЗАКЛЮЧЕНИЕ**

Цель курсовой работы – реализовать веб-сервис, распознающий математическую задачу и решающий ее – достигнута.

В ходе курсовой работы рассмотрены принципы построения нейронных сетей и изучены их области применения, а также направления применения цифр технологий в образовании. На основе академической литературы и веб-ресурсов проведен анализ приложений, которые распознают и решают математические примеры.

В практической части реализован веб-сервис, который решает задачу распознавания математических примеров и формул, заданных в рукописной форме. После того как приложение распознает текст с полотна, пользователю получает результат решения примера и его Latex формат.

Интерфейс веб-сервиса реализован на HTML, CCS3, JavaScript, jQuery.

В дальнейшем предполагается расширить область математических примеров, улучшение интерфейса и создания мобильной версии. Организовать хранение примеров и ответов в базе данных, добавление примеров для самостоятельного решения и отслеживания этапов решения.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Мобильное приложение Photomath : [[сайт](https://photomath.com/en/)]. – 2021. – URL: <https://photomath.com/en/> (дата обращения: 1.05.2022).
2. Detexify LaTeX handwritten symbol recognition : [[сайт](http://detexify.kirelabs.org/classify.html)]. – 2021. – URL: <http://detexify.kirelabs.org/> (дата обращения: 1.05.2022).
3. Веб приложение Mathway : [[сайт](https://www.mathway.com/ru/Algebra)]. – 2022. – URL: <https://www.mathway.com/ru/> (дата обращения: 1.05.2022)
4. Мобильное приложение Mathpix : [[сайт](https://mathpix.com/)]. – 2022 – URL: <https://mathpix.com/> (дата обращения: 1.12.2021).
5. Мобильное приложение MyScript Calculator : [сайт]. – 2022. – URL: <https://www.myscript.com/ru/calculator> (дата обращения: 1.05.2022).
6. Розенблатт, Ф. Принципы нейродинамики: Перцептроны и теория механизмов мозга / Ф. Розенблатт – М.: Мир, 1965. – 480 с.
7. Николенко, С.И. Глубокое обучение. Погружение в мир нейронных сетей / С.И. Николенко, А.А. Кадурин, Е.М. Архангельская – СПб.: Питер, 2020. – 480 с.
8. База данных MNIST: [[сайт](http://yann.lecun.com/exdb/mnist/)]. – 2022 – URL: <http://yann.lecun.com/exdb/mnist/> (дата обращения: 1.05.2022).
9. Документация LaTeX: [[сайт](http://www.latex‐project.org/help/documentation/)]. – 2022 – URL: <http://www.latex‐project.org/help/documentation/> (дата обращения: 1.05.2022).
10. Хайкин, С. Нейронные сети: полный курс/ С. Хайкин – М.: Издательский дом «Вильямс», 2006. – 1104c.
11. Гафаров, Ф.М. Искусственные нейронные сети и приложения: учеб. пособие/ Ф.М. Гафаров, А.Ф. Галимянов – Казань: Изд-во Казан. ун-та, 2018. – 121 с.