Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования
«Национальный исследовательский
Томский политехнический университет»

На правах рукописи

Фан Нгок Хоанг

АЛГОРИТМЫ ОБРАБОТКИ И АНАЛИЗА СИМВОЛОВ ВЕЙВЛЕТ-ПРЕОБРАЗОВАНИЕМ, МЕТОДОМ ГЛАВНЫХ КОМПОНЕНТ И НЕЙРОННЫМИ СЕТЯМИ

05.13.01 – Системный анализ, управление и обработка информации
(в отраслях информатики, вычислительной техники и автоматизации)

Диссертация
на соискание ученой степени
кандидата технических наук

Научный руководитель –
доктор технических наук,
профессор В.Г. Спицын

Томск 2014

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность работы. В области обработки изображений задача распознавания образов является одной из широко исследуемых задач в настоящее время. Решение задачи распознавания образов востребовано в различных сферах деятельности современного общества. Например, распознавание лиц используется в системах паспортного контроля аэропортов и вокзалов, распознавание радужной оболочки глаза – в системах контроля доступа, распознавание речи – для управления приборами, такими как компьютеры, телефоны, распознавание жестов – для взаимодействия с людьми с ограниченными возможностями и т.д.

Задача распознавания символов является одной из актуальных задач распознавания образов. Эту задачу можно использовать для решения других задач, таких как распознавание текстов, распознавание автомобильных номеров и т.п.

В настоящее время существует ряд программных средств и систем, использующих алгоритмы распознавания символов для решения задачи распознавания текстов. Широкое распространение получили такие программные средства как *ABBY FineReader*, *Tesseract OCR*, *CuneiForm*, *OmniPage*, *Readiris* и др. В каждом из перечисленных программных продуктов предложены свои алгоритмы и методы для распознавания символов и текстов. Однако большинство указанных программных средств являются коммерческими, поэтому алгоритмы и методы, применяемые в них для решения задач, известны только разработчикам.

Хотя перечисленные программы показывают высокую точность распознавания символов и текстов, но они не могут обеспечивать 100% точность распознавания для всех вариантов символов и текстов, а также в присутствии шума на изображениях. Указанные программные средства и системы продолжают развиваться в направлении повышения точности и скорости распознавания. Таким образом, можно сделать вывод, что разработка новых алгоритмов для распознавания символов и текстов

является актуальной задачей.

Главными преимуществами признакового подхода к распознаванию символов являются хорошая устойчивость к изменениям формы, размера и шрифта символов и высокая скорость распознавания. Кроме того, такой подход имеет другие преимущества, такие как низкое число отказов от распознавания и простота реализации. Благодаря указанным преимуществам признаковый подход выбран для дальнейшей разработки алгоритмов, предназначенных для распознавания символов разных шрифтов с высоким быстродействием.

Вейвлет-преобразование является эффективным методом для выделения признаков объектов при решении задач распознавания образов. При разложении изображения вейвлет-преобразование содержит в себе необходимую информацию об этом изображении. Например, субдиапазоны в пространственных и частотных доменах, в различных разрешениях, в горизонтальном, вертикальном и диагональном направлениях. Вейвлет-преобразование также достаточно быстро вычисляется. В алгоритмах ряда авторов *Mehdi L.*, *Weibao Z.*, *Chang T.*, *Daniel M.R.S.*, *Park S.B.* и *Gonzalez A.C.* используются вейвлет-преобразования для решения задачи классификации изображений. В указанных работах показано, что при использовании вейвлет-преобразования для выделения признаков изображения точность классификации изображений составляет 76–99,7%.

Кроме того, в области распознавания образов вейвлет-преобразования используются в алгоритмах ряда авторов *Lai J.H.*, *Kakarwal S.*, *Zhang B.*, *Gumus E.*, *Wadkar P.D.*, *Kumar S.V.P.* и *Mazloom M.* В указанных работах показано, что вейвлет-преобразования эффективно используются при решении задач распознавания образов, в особенности, задачи распознавания лиц. Точность распознавания лиц при этом составляет 90–98,5%.

Таким образом, применение вейвлет-преобразования является перспективным способом для разработки новых алгоритмов распознавания символов и текстов.

Целью диссертационной работы является разработка алгоритмов на основе вейвлет-преобразования, метода главных компонент и нейронных сетей, способных распознавать символы разных шрифтов и фрагменты текстов.

Для достижения поставленной цели необходимо решить следующие **основные задачи.**

1. Разработать алгоритм распознавания символов на основе вейвлет-преобразования, метода главных компонент и нейронных сетей.
2. Разработать способ построения классификатора для распознавания символов на основе нейронных сетей.
3. Создать алгоритм распознавания фрагментов печатных текстов на основе разработанного алгоритма распознавания символов.
4. Осуществить апробацию созданных в диссертационной работе алгоритмов на задачах распознавания символов и фрагментов печатных текстов на изображениях.

Апробация работы. Результаты диссертационной работы докладывались и обсуждались на следующих симпозиумах, конференциях и семинарах: VIII, IX Всероссийские научно-практические конференции студентов, аспирантов и молодых ученых «Молодежь и современные информационные технологии» (Томск, 2010, 2011); XVI, XVII Международные научно-практические конференции студентов, аспирантов и молодых ученых «Современные техника и технологии» (Томск, 2010, 2011); III Всероссийская научно-практическая конференция «Научная инициатива иностранных студентов и аспирантов российских вузов» (Томск, 2010); VI Международная научно-практическая конференция «Электронные средства и системы управления» (Томск, 2010); XIX Всероссийский семинар «Нейроинформатика, ее приложения и анализ данных» (Красноярск, 2011); XIV Всероссийский с международным участием научный симпозиум по теории и приложениям непараметрических и робастных статистических методов «НЕПАРАМЕТРИКА-XIV» (Томск, 2012); The 7th International Forum

on Strategic Technology IFOST (Томск, 2012).

Кратно изложим **основное содержание работы.**

В **первой главе** проведен анализ систем распознавания текстов и подходов к распознаванию символов. Приведен аналитический обзор основных методов и алгоритмов для выделения признаков изображения. Описывается принцип работы двумерного вейвлет-преобразования и определяются его преимущества при решении задач обработки изображений и распознавания образов. Проведен анализ основных методов и алгоритмов, основанных на вейвлет-преобразовании, предназначенных для решения задач обработки изображений и распознавания образов.

Во второй главе предложен и описан способ построения классификатора для распознавания символов на основе нейронных сетей. Разработан и описан алгоритм распознавания символов, основанный на вейвлет-преобразовании, методе главных компонент и нейронных сетях. Создан и описан алгоритм распознавания фрагментов печатных текстов, основанный на разработанном алгоритме распознавания символов и способе выделения символов из фрагмента текста.

В **третьей главе** рассматриваются средства для разработки программного обеспечения. Проведен анализ инструментальных библиотек по обработке изображений, по результатам которого сделан выбор объектно-ориентированного языка программирования C#, библиотек Emgu CV и OpenCV для дальнейшей разработки. В главе содержится описание библиотеки, предназначенной для реализации предложенного способа построения классификатора для распознавания символов, предложенных алгоритмов распознавания символов и фрагментов печатных текстов.

Указанная библиотека состоит из двух модулей. Первый модуль предназначен для выделения признаков и распознавания символов, а второй модуль осуществляет выделение символов из фрагмента текста. Подробно описываются классы этих модулей. В главе также описываются программные средства, разработанные для реализации библиотеки, имеющие два варианта

интерфейса для конечных пользователей. Первый вариант предназначен для исследователей, а второй вариант для обычных пользователей.

В **четвертой главе** приведены результаты апробации разработанных алгоритмов на задачах распознавания рукописных цифр, печатных символов и фрагментов печатных текстов. Представлены данные для обучения и тестирования алгоритмов, а также таблицы и диаграммы, в которых представлены результаты распознавания. Произведено сопоставление разработанных алгоритмов с другими современными алгоритмами распознавания.

Научную новизну полученных в диссертации результатов определяют следующие положения.

1. Впервые предложен способ построения классификатора для распознавания символов на основе нейронных сетей, отличающийся от других тем, что каждая нейронная сеть соответствует только одному символу обучающей выборки.

2. Предложен новый алгоритм, основанный на вейвлет-преобразовании, методе главных компонент и нейронных сетях, позволяющий распознавать символы разных шрифтов в присутствии шума на изображениях.

3. Разработан оригинальный алгоритм, основанный на предложенном алгоритме распознавания символов и способе выделения символов из фрагмента текста, позволяющий распознавать фрагменты печатных текстов.

Практическая ценность. Разработанные в ходе диссертационной работы алгоритмические и программные средства предназначены для использования в системах OCR, системах распознавания номеров автомобилей, при обработке изображений.

Реализованные в диссертации алгоритмы предназначены для распознавания рукописных цифр, печатных символов и фрагментов печатных текстов.

Апробация реализованных алгоритмов осуществлялась на задачах распознавания рукописных цифр и печатных символов, при распознавании фрагментов печатных текстов.

Методы исследования. Для решения поставленных задач используются вейвлет-преобразования, метод главных компонент, аппарат нейронных сетей, методы цифровой обработки изображений, методы вычислительной математики и численные компьютерные эксперименты для оценки надежности и эффективности разработанных алгоритмов.

Личный вклад автора. Постановка задач исследования по теме диссертации выполнена автором совместно с научным руководителем, д.т.н., профессором В.Г. Спицыным. Основные теоретические результаты, представленные в диссертации, получены лично автором.

Основные положения, выносимые на защиту.

1. Способ построения классификатора для распознавания символов на основе нейронных сетей, отличающийся от других тем, что каждая нейронная сеть соответствует только одному символу обучающей выборки.
2. Алгоритм распознавания символов, основанный на вейвлет-преобразовании, методе главных компонент и нейронных сетях.
3. Алгоритм распознавания фрагментов печатных текстов, основанный на предложенном алгоритме распознавания символов и способе выделения символов из фрагмента текста.
4. Разработанное программное обеспечение позволяет успешно распознавать рукописные цифры, печатные символы и фрагменты печатных текстов.

Автор выражает глубокую благодарность научному руководителю профессору, доктору технических наук В.Г. Спицыну за помощь в написании работы, ценные советы, замечания и доброжелательную критику. Автор также благодарит за ценные замечания и всестороннюю помощь кандидатов технических наук, доцентов Ю.Р. Цоя и Ю.А. Болотову. Автор благодарит заведующего кафедрой Вычислительной техники, профессора Н.Г. Маркова

за ценные замечания и обсуждение работы. Автор также благодарит за обсуждение работы доцентов Томского политехнического университета, кандидата технических наук Е.А. Мирошниченко и кандидата физико-математических наук Ю.Б. Буркатовскую.

Степень достоверности результатов проведенных исследований подтверждена результатами численных экспериментов на различных тестовых задачах и согласованностью результатов диссертационной работы с результатами, полученными другими авторами.

Внедрение работы. Результаты работы внедрены в Томском политехническом университете на кафедре вычислительной техники при подготовке специалистов по магистерской программе «Компьютерный анализ и интерпретация данных» по курсу «Методы интеллектуальной обработки и анализа изображений».

Структура и объем работы. Диссертация содержит введение, четыре главы, заключение, список использованной литературы, содержащий 117 наименований. Общий объем диссертации составляет 138 страниц машинописного текста, включающих 63 рисунка и 26 таблиц.

Глава 1. Аналитический обзор подходов к распознаванию символов

В данной главе проведен анализ систем распознавания текстов и подходов к распознаванию символов. Приведен аналитический обзор основных методов и алгоритмов для выделения признаков изображения. Описывается принцип работы двумерного вейвлет-преобразования и определяются его преимущества при решении задач обработки изображений и распознавания образов. Проведен анализ основных методов и алгоритмов, основанных на вейвлет-преобразовании, предназначенных для решения задач обработки изображений и распознавания образов.

1.1 Основные задачи обработки изображений

Методы цифровой обработки изображений можно разделить на две категории. Первая категория относится к методам, в которых на входе и на выходе имеются изображения. Во второй категории входом методов являются изображения, выходом – признаки и атрибуты этих изображений [10]. Основные стадии цифровой обработки изображений приведены в табл. 1.1.

Таблица 1.1. Основные стадии цифровой обработки изображений

	Первая категория	**Вторая категория**
Стадии	- Регистрация изображения - Улучшение изображения - Восстановление изображения - Кратномасштабная обработка - Сжатие изображения	- Сегментация изображения - Представление и описание - Распознавание образов

1. Регистрация изображения является предельно простой задачей, в случае, когда исходное изображение уже представлено в цифровой форме. Такая задача применяется во всех областях и в общем случае она включает

некоторые этапы предобработки, такие как масштабирование.

2. Методами улучшения изображения являются методы выявления плохо различимых деталей или просто подчеркивания интересующих характеристик на исходном изображении. Улучшение изображения может применяться на этапе предобработки изображения. Оно часто используется при обработке медицинских изображений и т.п. Методы улучшения изображений основаны на предпочтениях человеческого восприятия, которые оценивают, насколько хорош результат улучшения.

3. Задача восстановления изображения связана с повышением визуального качества изображения. Методы такой задачи основаны на математических или вероятностных моделях искажений изображения. К восстановлению изображения также относятся методы подавления шума. Известными методами подавления шума являются фильтры, такие как медианный фильтр, адаптивный медианный фильтр, винеровская фильтрация и др.

4. Кратномасштабная обработка позволяет представлять изображение с несколькими степенями разрешения. В области обработки изображений кратномасштабная обработка широко применяется для решения таких задач, как подавление шума изображений, сжатие изображений и т.п.

5. Задача сжатия изображения относится к методам, в результате которых уменьшается объем памяти, необходимого для хранения или для передачи изображения. Сжатие изображения подразделяют на две категории: сжатие с потерями и сжатие без потерь качества.

6. Цель задачи сегментации изображения заключается в разделении изображения на составные части или объекты, для того чтобы его было проще и легче анализировать. Сегментация изображения обычно используется для решения таких задач, как выделение объектов и выделение границ на изображениях. Примерами известных методов сегментации изображения являются такие методы, как метод водораздела (*Watershed*), текстурная сегментация, алгоритм К-средних и др.

7. Представление и описание часто следуют за этапом сегментации с целью преобразования данных в форму, которая удобна для компьютерной обработки.

8. Стадия распознавания образов относится к методам, которые классифицируют объекты по категориям или классам. Такая стадия является одной из самых изученных задач в таких областях как цифровая обработка изображений, компьютерное зрение, биометрия, создание интеллектуальных систем безопасности и контроля доступа и т.п. Тем не менее, в области распознавания образов продолжают представлять большой научный и практический интерес такие задачи как распознавание лиц, жестов, отпечатков пальцев, печатных и рукописных символов и текстов.

1.2 Подходы и системы распознавания символов и текстов

Задача распознавания символов является одной из актуальных задач распознавания образов в настоящее время. Эту задачу можно использовать для решения других задач, таких как распознавание текста, распознавание автомобильных номеров и т.п., являющихся востребованными в различных сферах деятельности современного общества.

1.2.1 Системы распознавания текста

В настоящее время существует ряд программных средств и систем, использующих алгоритмы распознавания символов для решения задачи распознавания текстов. Распространение получили такие программные средства как *ABBY FineReader, Tesseract OCR, CuneiForm, OmniPage, Readiris* и др. В каждом из перечисленных программных продуктов предложены свои алгоритмы и методы для распознавания текстов. Однако большинство указанных программных средств являются коммерческими, поэтому алгоритмы и методы, применяемые в них для решения задач, известны только разработчикам.

Хотя перечисленные программы показывают высокую точность распознавания символов и текстов, но они не могут обеспечивать точность распознавания 100% для всех вариантов символов и текстов, а также в присутствии шума на изображениях. Таким образом, можно сделать вывод, что разработка новых алгоритмов для распознавания символов и текстов является актуальной задачей.

a. ABBY FineReader

FineReader 11 в настоящее время является одной из наиболее распространенных программ распознавания текста, реализованной компанией **ABBYY**, основанной в 1989 г. Программа позволяет преобразовывать изображения документов и PDF-файлов в виде электронных редактируемых форматов. Она обеспечивает возможность сохранения результатов распознавания в различных форматах, таких как DOC, DOCX, DjVu, TXT и др.

Кроме того, FineReader 11 обеспечивает высокую точность распознавания текста и также сохраняет структуру документа при распознавании. Программа может распознавать тексты на 188 языках, основанных на кириллице, латинице, греческом, армянском алфавитах и на иероглифических символах. В том числе для 44 языков используется словарь и проверка орфографии при распознавании [2].

b. Tesseract OCR

Tesseract OCR представляет собой систему распознавания текста, обеспечивающую открытый доступ к своим исходным кодам. В 1995 г. она является одной из трех лучших программ по точности распознавания текста в конкурсе «*The Fourth Annual Test of OCR Accuracy*» [97]. Однако с 1995 г. по 2006 г. развитие системы замедлилось. В августе 2006 г. компания Google купила эту систему и обеспечила открытый доступ к исходным кодам под лицензией Apache 2.0 для дальнейшей разработки.

В настоящее время Tesseract OCR является одной из наиболее точных систем распознавания текста. Она позволяет на основе версии 3.0 преобразовывать большое количество форматов изображений в тексты на более чем 60 языках, в том числе на английском, русском, украинском и других. Однако система поддерживает сохранение результата распознавания только в формате текста, для сохранения в других форматах пользователям необходимо реализовать дополнительную программу [23].

c. CuneiForm

CuneiForm является программой, предназначенной для распознавания текста документов и преобразования его в редактируемый вид. Программа позволяет редактировать результаты распознавания в офисных средствах и текстовых редакторах. При распознавании она также обеспечивает возможность сохранения структуры и форматирования документа [1].

Многие результаты научных исследований и технологические ноу-хау, основанные на CuneiForm, успешно применяются в коммерческих продуктах компании Cognitive Technologies, таких как Cognitive Forms, Cognitive Forms Bank, Cognitive Passport и Cognitive ScanPack. Следует отметить, что компанией Cognitive Technologies принято решение сделать CuneiForm бесплатной программой и обеспечить доступ к ее исходным кодам.

CuneiForm позволяет распознавать текст документов на более чем 20 языках, в том числе на английском, русском, немецком и других. В программе используется словарная проверка для повышения точности распознавания. При этом программа позволяет расширить использованный словарь с помощью импорта новых слов из текстовых файлов.

1.2.2 Подходы к распознаванию символов

В данном разделе проводится анализ подходов, предназначенных для решения задачи распознавания символов. В настоящее время подходы для решения этой задачи можно разделить на три основных подхода: шаблонный, структурный и признаковый [3, 9, 17].

а. Шаблонный подход

Алгоритмы шаблонного подхода обычно требуют предварительную обработку для обеспечения приемлемой точности распознавания символов. Основной предварительной обработкой данного подхода является представление изображения символов в виде растрового изображения, нормализация размера, наклона и толщины штриха символов. Принцип работы алгоритмов шаблонного подхода основан на прямом сравнении изображения распознаваемого символа со всеми шаблонами, хранящимися в базе шаблонов. Наиболее подходящим является шаблон, который имеет наименьшее количество несовпадающих пикселей, отличающих этот шаблон от изображения распознаваемого символа.

При этом большинство систем, использующих алгоритмы данного подхода, имеет шаблоны, созданные для различных начертаний. После того как осуществлено распознавание нескольких слов, система определяет основный используемый шрифт и изображения символов будут сравниваться с шаблонами только этого шрифта. В некоторых случаях система использует численные значения частей символа для определения нового шрифта. Это может улучшить эффективность распознавания символов.

К преимуществам шаблонного подхода относятся простота реализации, высокая скорость распознавания и хорошая устойчивость к дефектам изображений символов.

Недостаток данного подхода заключается в том, что при данном подходе невозможно распознавать шрифты, которые немного отличаются от шрифтов в базе шаблонов по размеру и начертанию. При этом требуется необходимость настройки системы на типы и размеры шрифтов. Алгоритмы шаблонного подхода должны заранее знать шрифты, которые эти алгоритмы будут распознавать. Однако все шрифты и их модификации невозможно охватить при обучении алгоритмов. При этом алгоритмы шаблонного подхода не являются универсальным.

b. Структурный подход

Алгоритмы данного подхода распознают символы на основе представления изображений символов в виде топологии, содержащей информацию о взаимном расположении отдельных составных частей символов. Каждый из этих алгоритмов ищет особые характеристики символов, такие как острова, полуострова, точки, прямые оттиски и дуги. Такие особые характеристики могут быть представлены в виде формы графа. Алгоритмы данного подхода также требуют предварительную обработку, которой является процесс скелетизации, предназначенный для утоньшения символов.

Преимущество данного подхода заключается в том, что данный подход обеспечивает инвариантность относительно типов и размеров шрифтов символов. Таким образом, точность распознавания символов алгоритмами данного подхода не зависит от шрифтов символов.

К недостаткам структурного подхода относятся низкая скорость распознавания и трудность распознавания дефектных символов. Дефектный символ может стать специфической проблемой для данного подхода, так как отсутствующий пиксель может разбивать длинный штрих или кривую, а дополнительное пятно грязи может закрывать петлю.

c. Признаковый подход

Алгоритмы признакового подхода работают на основе представления изображений символов в виде вектора признаков. Процесс распознавания символов заключается в сравнении вектора признаков изображений символов с набором векторов изображений символов обучающей выборки той же размерности. В алгоритмах данного подхода чаще всего используется классификатор, основанный на определении расстояния Евклида между вектором признаков изображения распознаваемого символа и векторами признаков изображений символов обучающей выборки. Процесс формирования вектора, представляющего изображение символа, называется процессом выделения признаков. Признаки каждого изображения

распознаваемого символа получают путем аналоговой обработки изображений символов обучающей выборки.

Основными преимуществами признакового подхода являются простота реализации, хорошая обобщающая способность, хорошая устойчивость к изменениям формы, размеры и шрифта символов, низкое число отказов от распознавания и высокая скорость распознавания.

Недостаток данного подхода заключается в неустойчивости к различным дефектам изображений и потере части информации о символе на этапе выделения признаков. Выделение признаков ведется независимо, поэтому информация о взаимном расположении элементов символа утрачивается.

Благодаря своим преимуществам признаковый подход выбран для дальнейшей разработки алгоритмов, предназначенных для распознавания символов разных шрифтов с высоким быстродействием.

1.2.3 Выделение признаков

Выделение признаков традиционно является предметом исследования в области распознавания образов. Оно представляет собой один из шагов, необходимых для решения задачи распознавания. В работах [33, 38] перечислены следующие основные методы выделения признаков.

a. Моменты изображений

Во многих приложениях моменты изображений используются в качестве признаков образов, предназначенных для решения задачи их распознавания. Эти признаки содержат глобальные свойства изображения, такие как область формы, центр массы и т.п.

Геометрические моменты являются распространенными типами моментов. Геометрический момент цифрового изображения можно вычислить по следующей форме:

$$m_{p,q} = \sum_{-\infty}^{+\infty}\sum_{-\infty}^{+\infty} x^p y^p f(x,y), \qquad (1.1)$$

где p и q – целые числа (0, ∞), представляющие уровень момента; x и y – координаты пикселя изображения; $f(x, y)$ – интенсивность изображения в этом пикселе. На основе этих моментов *Hu M.K.* предложил способ выделения признаков изображения, являющихся инвариантными при преобразовании изображения [56].

Моменты Zernike являются проекциями интенсивности изображения $f(x, y)$ на ортогональный полином Zernike, определяющий полное ортогональное множество сложных полиномов по координатам (x, y), расположенным в круге $x^2 + y^2 = 1$ [62].

$$V_{p,q}(x,y) = V_{p,q}(\rho,\theta) = R_{p,q}(\rho)e^{jq\theta}, \qquad (1.2)$$

где p и q – целые числа, представляющие уровень момента, $j = \sqrt{-1}$, $p \geq 0$, $p \geq |q|$, $p-|q|$ – четное число, ρ – наименьшее расстояние пикселя (x, y) до центра круга, θ – угол между вектором ρ и осью X, $R_{p,q}(\rho)$ – ортогональный радиальный полином, вычисляющийся по формуле:

$$R_{p,q}(\rho) = \sum_{s=0}^{(p-|q|)/2} \frac{(-1)^s (p-s)!}{s!(\frac{p+|q|}{2}-s)!(\frac{p-|q|}{2}-s)!}\rho^{p-2s}. \qquad (1.3)$$

Момент Zernike цифрового изображения вычисляется следующим образом:

$$Z_{p,q} = \frac{p+1}{\pi}\sum_x\sum_y f(x,y)V^*_{p,q}(\rho,\theta),\ x^2+y^2 \leq 1. \qquad (1.4)$$

Чтобы вычислить моменты Zernike используются координаты центра изображения в качестве координат центра круга. Пиксели, расположенные вне круга, не учтены при вычислении моментов.

Изображение $f(x, y)$ можно представить с использованием моментов Zernike в следующем виде:

$$f_Z(x,y) = \sum_{p=0}^{N} \sum_{q=-N}^{N} Z_{p,q} R_{p,q}(\rho) e^{jq\theta}, \quad (1.5)$$

где p–$|q|$ – четное число; N – максимальный уровень моментов Zernike.

На рис. 1.1 приведены исходное изображение буквы «B» и его представление на основе применения моментов Zernike с максимальным уровнем момента N: 0, 2, 4, 6, 8, 10, 12, 14, 16 и 18 («zm0»–«zm18» соответственно).

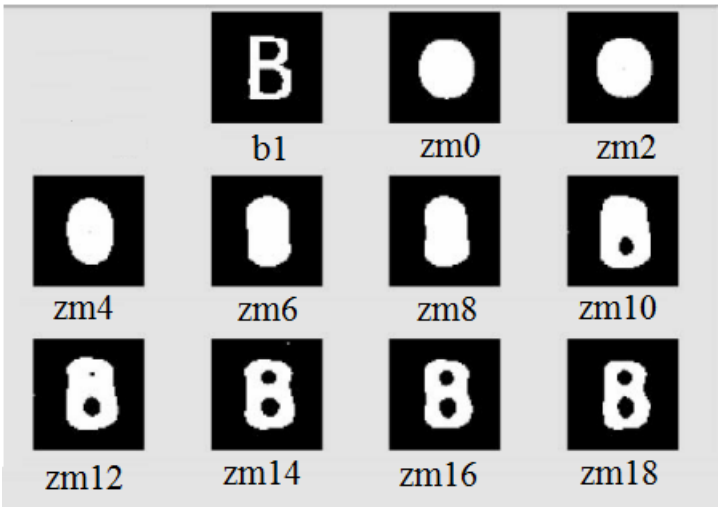


Рисунок 1.1. Преставление изображения буквы «B» на основе применения моментов Zernike [60]

Моменты Legendre цифрового изображения с размером $N \times N$ пикселей можно вычислить по следующей формуле:

$$L_{p,q} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} P_p(x_N) P_q(y_N) f(x,y), \quad (1.6)$$

где p и q – целые числа $(0, \infty)$, представляющие уровень момента;

$$x_N = \frac{2x - N + 1}{N - 1},$$

$$P_p(x) = \frac{1}{2^p p!} \frac{d^p}{dx^p} (x^2 - 1)^p \text{ и } x \in [-1, 1],$$

$$P_q(y) = \frac{1}{2^q q!} \frac{d^q}{dy^q} (y^2 - 1)^q \text{ и } y \in [-1, 1].$$

В результате представления изображения $f(x, y)$ с использованием моментов Legendre получается новое изображение, значение пикселя (x, y) которого вычисляется по следующей формуле:

$$f_L(x,y) = \sum_{p=0}^{M} \sum_{q=0}^{p} L_{p,q} P_p(x) P_q(y), \tag{1.7}$$

где M – максимальный уровень моментов Legendre.

Пример представления изображения символа с использованием моментов Legendre приведен на рис. 1.2. Слева-направо на рис. 1.2 приведены исходные полутоновое (в верхней строке) и бинарное (в нижней строке) изображения символов и их представление на основе применения моментов Legendre с максимальным уровнем момента M: 20, 40 и 60.

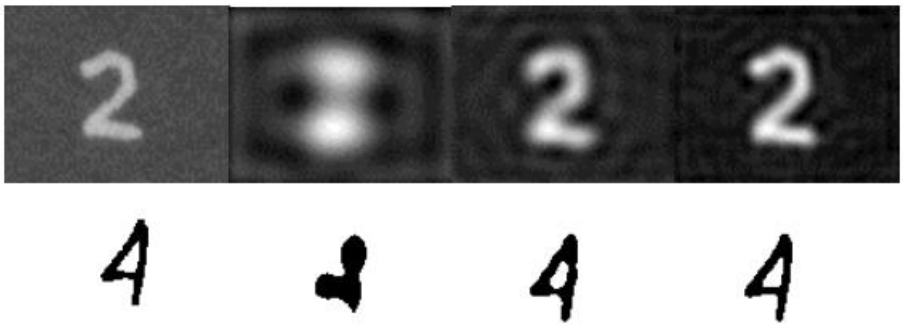


Рисунок 1.2. Представление изображения символа на основе применения моментов Legendre [105]

b. Гистограмма изображения

Гистограммы представляют распределение пикселей полутонового изображения по их интенсивности $f(x, y)$. Амплитуда гистограммы полутонового изображения является таблицей $H(k)$, показывающей количество пикселей, имеющих значение интенсивности k. Амплитуда гистограммы полутонового изображения размером $N \times N$ пикселей вычисляется следующим образом:

$$\begin{aligned} H(k) &= count\{ f(x,y) \mid f(x,y) = k \}, \\ k &\in [0, k_{\max}], (x,y) \in [(0,0), (N-1, N-1)]. \end{aligned} \tag{1.8}$$

c. Направленные признаки

Символы включают штрихи (*stroke*), ориентация которых играет важную роль в дифференциации между различными символами. При распознавании символов векторы из статистики ориентаций штрихов используются в качестве векторов признаков символов. Для выделения признаков символа угол ориентации штриха разбивается на фиксированное число диапазонов и количество сегментов штрихов в каждом диапазоне используется в качестве значения признака. Множество количеств сегментов штрихов, которое формулирует гистограмму, называется гистограммой ориентации [48].

Чтобы увеличить возможность дифференциации гистограмма обычно вычисляется для локальных диапазонов изображения символов и называется гистограммой локальной ориентации. Локальные ориентации штрихов символов можно определить по-разному: ориентация скелетона, сегмент штриха, цепной код (*chaincode*) контура, направление градиента и т.п. [65, 72, 99, 111].

На рис. 1.3 приведен пример представления изображения символа с использованием направленных признаков. На рис. 1.3(а) представлено исходное изображение символа, а на рис. 1.3(б–г) приведены его представления на основе применения контура, направления градиента и гистограммы локальной ориентации.

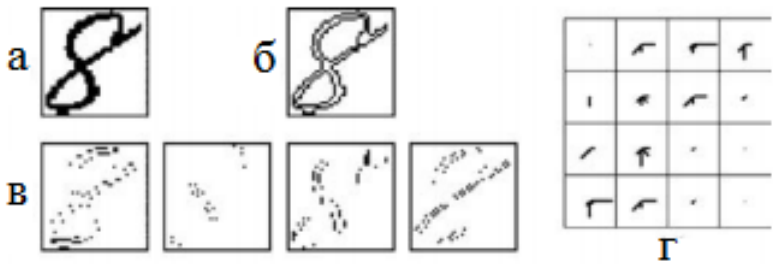


Рисунок 1.3. Представление изображения символа на основе применения направленных признаков [38]

d. Преобразование Хафа

Распространенным методом выделения признаков в области обработки изображений является преобразование Хафа. Оно позволяет обнаружить прямые линии, кривые или любую конкретную форму, которые можно определить с использованием параметрических уравнений. Преобразование Хафа отображает точки из пространства изображения в пространстве параметров и после этого выделяет признаки изображения. Преобразования Хафа можно разделить на два типа: стандартное и случайное (*randomized*) преобразование Хафа.

В стандартном преобразовании Хафа все точки $\{(x_1, y_1\},\ldots,(x_n, y_n)\}$ пространства изображения преобразуются в пространство параметров [46]. Например, линия в пространстве изображения, $y = mx + b$, определяется в пространстве параметров по следующей формуле:

$$\rho = x\cos\theta + y\sin\theta, \tag{1.9}$$

где ρ – расстояние от центра пространства до линии; θ – угол между ρ и осью X (рис. 1.4). Таким образом, прямая линия в пространстве изображения отражается как уникальная точка в пространстве параметров.

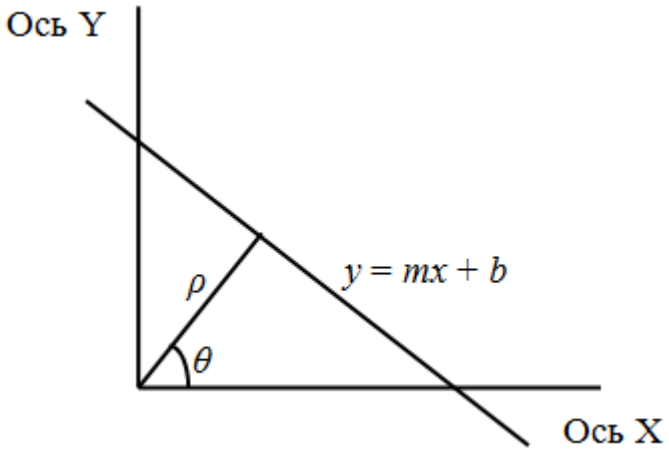


Рисунок 1.4. Представление прямой линии в пространстве параметров

Случайным преобразованием Хафа является расширение стандартного преобразования, предназначенное для выделения признаков бинарных изображений. Оно учитывает недостатки стандартного преобразования Хафа, а именно, длительное время вычислений и большие требования к памяти.

е. Представление на основе линии

Штрихи на изображениях символов обычно толще, чем один пиксель, поэтому признаки символов, основанные на линии, можно выделить с использованием контура штриха или скелета.

Край (контур) определяется как граница между двумя областями изображения, имеющими различные значения серого уровня [10]. Значения серого уровня пикселей изображения, расположенных на крае, сильно изменяются. Это изменение проводит к неоднородности серого уровня. Таким образом, край на изображениях можно выделить на основе использования этого свойства.

В большинстве методов выделения края используются первая и вторая производные. Сильное изменение в величине первой производной показывает присутствие края на изображениях. Точно так же вторая производная используется, чтобы определить находится ли пиксель края на темной или светлой стороне. Если значение второй производной положительно, то пиксель края принадлежит к темной стороне. В обратном случае пиксель края принадлежит к светлой стороне.

Утоньшением является преобразование изображения, после которого все линии на изображении имеют толщину в один пиксель. Такая линия обычно называется скелетоном. Большинство алгоритмов утоньшения повторяет удаление последовательных слоев пикселей, расположенных на крае символа, пока не оставляют только один скелетон [115]. Алгоритмы утоньшения можно разделить на две категории: последовательные и параллельные.

В последовательном алгоритме необходимо предопределить последовательность пикселей, в которой пиксели будут проверены на удаление. При этом работа n-ой итерации зависит от результатов всех предыдущих итераций (1, 2,…, n–1). В параллельном алгоритме работа n-ой

итерации зависит от результата только предыдущей итерации (n–1).

Примеры результатов утоньшения изображений букв «М» и «К» представлены на рис. 1.5. Слева-направо на рис. 1.5 представлены исходные изображения символов и результаты их утоньшения.

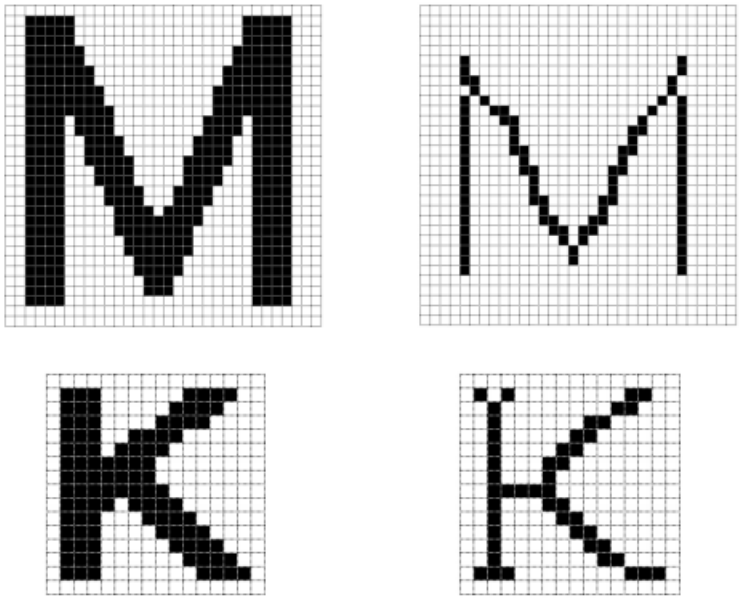


Рисунок 1.5. Результаты утоньшения изображений символов [85]

f. Дескрипторы Фурье

В настоящее время дескрипторы Фурье являются одним из распространенных и эффективных методов, предназначенных для выделения формы на изображениях. В принципе эти дескрипторы представляют форму объекта в частотном домене. Преимущество дескрипторов Фурье заключается в том, что они обеспечивают возможность дифференциации между различными символами, низкую шумовую чувствительность и сохранение информации о символе [115].

Дескрипторы Фурье можно разделить на два типа: стандартные и эллиптические. Они используются для описания формы (закрытая кривая) любого объекта, найденного на входном изображении. Для вычисления дескрипторов Фурье вначале необходимо выделить контуры символа на изображении, затем полученные данные преобразуются в виде одномерного представления.

g. Линейное преобразование

Методы преобразования пространства признаков обычно используются в области распознавания образов. Они изменяют представление признаков образов с целью улучшения точности классификации.

Метод главных компонент (МГК) является статистическим методом, основная цель которого заключается в уменьшении множества признаков путем определения подмножества преобразованных признаков, которое содержит ту же необходимую информацию, что и оригинальное множество признаков.

МГК является линейным, поэтому каждый новый преобразованный признак представляет собой линейную комбинацию оригинального множества признаков. МГК также включает в себя концепцию регрессии, целью которой является нахождение простого способа для объяснения отношения между зависимой переменной Y и независимой переменной X. Простым линейным отношением является то, которое минимизирует сумму квадратов ошибок регрессии. Примеры собственных символов, полученных на основе применения МГК, представлены на рис. 1.6.

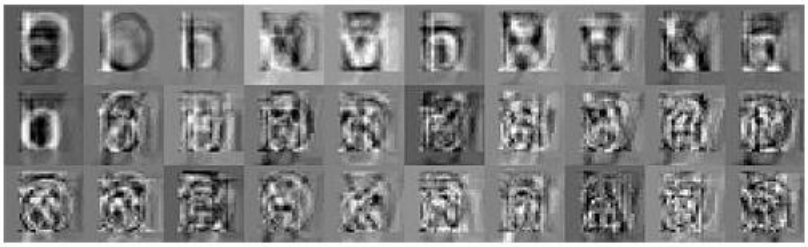


Рисунок 1.6. Собственные символы при использовании МГК [42]

Линейный дискриминантный анализ (LDA) является распространенным методом, применяющимся для нахождения линейного подпространства признаков, которое максимизирует критерий Фишера [49].

LDA представляет собой параметрический метод выделения признаков, который вычисляет Гауссову плотность распределения вероятности с равной ковариацией для всех классов. Выделенное подпространство работает

довольно хорошо даже в ситуациях, в которых классы описываются негауссовскими распределениями или ковариационные матрицы не равны. Полученные с использованием методом LDA признаки могут быть в дальнейшем распознаны линейными или нелинейными классификаторами.

h. Вейвлет-преобразование

Вейвлет-преобразование впервые предложено математиком *Alfred Haar* в 1910 г. и было названо вейвлетом Хаара. Однако в то время концепции вейвлета еще не существовало. В 1982 г. концепция вейвлета была предложена геофизиком *Jean Morlet* [84] и вейвлет-преобразования сразу были признаны в качестве нового метода для анализа сейсмических сигналов. В это же время обратное преобразование вейвлета было исследовано физиком *Alex Grossmann*. В 1984 г. *Morlet* и *Grossmann* совместно провели подробное математическое исследование такого типа вейвлета, называемого интегральным (непрерывным) вейвлет-преобразованием [52]. В 1985 г. *Yves Meyer* разработал новое преобразование вейвлета, названное вейвлетом Meyer. В 1988 г. *Meyer* и *Stephane Mallat* совместно предложили концепцию кратномасштабного разложения (*multiresolution*) [77]. В этом же году *Ingrid Daubechies* разработала метод, предназначенный для конструкции полного ортогонального вейвлет-преобразования [44]. В 1989 г. *Mallat* предложил так называемое быстрое вейвлет-преобразование, являющееся надежным методом, предназначенным для выполнения вычислений дискретного вейвлет-преобразования [79]. В это время, благодаря приемлемой скорости вычислений, быстрое вейвлет-преобразование широко использовалось для решения задач области обработки сигналов.

Все эти типы вейвлет-преобразований являются одномерными вейвлет-преобразованиями и они часто используются для решения задач обработки непрерывных сигналов, например, для подавления шумов на музыкальных записях [40, 78]. Однако сигнал можно разделить на два типа: непрерывный и

дискретный. Примером дискретного сигнала является изображение, описываемое функцией $f(x, y)$, где x и y – координаты этого пикселя в пространстве, а значение f указывает интенсивность изображения в этом пикселе. Для анализа и обработки дискретного сигнала целесообразно использовать двумерное вейвлет-преобразование, которое позволяет упростить работу с дискретными сигналами по сравнению с одномерным вейвлет-преобразованием.

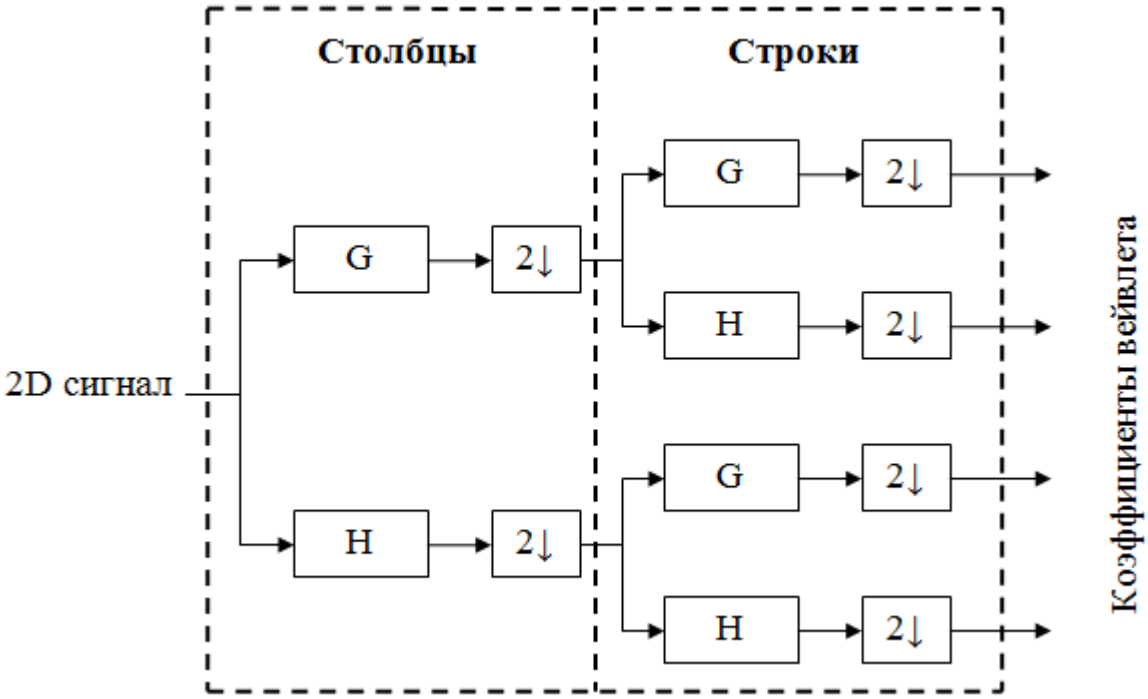


Рисунок 1.7. Схема работы двумерного вейвлет-преобразования

Схема работы первого разложения 2D сигнала, основанного на использовании двумерного вейвлет-преобразования, представлена на рис. 1.7. Вначале сигнал преобразуют по столбцам с использованием низкочастотного фильтра *G* и высокочастотного фильтра *H*. Затем два полученных субдиапазона аналогично преобразуют по строкам. В результате получаются четыре субдианазона входного сигнала. На рис. 1.8 представлен пример первого разложения изображения двумерным вейвлет-преобразованием Хаара.

Рисунок 1.8. Пример разложения изображения двумерным вейвлет-преобразованием Хаара: а) – исходное изображение; б) – преобразование по столбцам; в) – преобразование по строкам изображения (б)

Использование двумерного вейвлет-преобразования для обработки изображений обладает следующими преимуществами.

- Вейвлет-преобразование разделяет изображение на субдиапазоны, представляющиеся как в пространственных, так и в частотных доменах.
- При разложении изображения на субдиапазоны, вейвлет-преобразование позволяет анализировать это изображение в различных разрешениях.
- Вейвлет-преобразование дает возможность анализа изображений по разным направлениям, таким как горизонтальное (вдоль столбцов), вертикальное (вдоль строк) и диагональное направление.
- Использование вейвлет-преобразований позволяет определить маленькие детали на изображениях. Маленькие вейвлеты можно использовать для выделения маленьких деталей, а большие вейвлеты – для выделения больших деталей изображений.
- Вейвлеты могут вычисляться с высокой скоростью при разложении изображений.

Таким образом, вейвлет-преобразование является перспективным методом для выделения признаков символов. Далее проводится обзор и

анализ методов и алгоритмов обработки изображений и распознавания образов, основанных на вейвлет-преобразованиях.

1.3 Методы обработки изображений и распознавания образов с использованием вейвлет-преобразования

В области обработки изображений вейвлет-преобразование является одним из распространенных и эффективных методов, предназначенных для решения задач этой области. В работах [30, 34, 47, 54, 57, 76, 87–88, 92–94] показано эффективное использование вейвлет-преобразования для подавления шума на изображениях, а в работах [25, 32, 45, 50, 61, 71, 73, 100–101] представлены способности вейвлетов при сжатии изображений. В [26–27, 35–37, 63, 86, 104, 116] представлено применение вейвлетов для решения задачи сегментации изображений. Однако количество работ, посвященных применению вейвлет-преобразований для распознавания образов, еще невелико. Далее проводится подробный аналитический обзор методов и алгоритмов распознавания образов на основе применения вейвлет-преобразований.

1.3.1 Построение дескриптора фигуры

В области распознавания образов фигуры являются самыми главными признаками образов. Дескрипторы, которые представляют эти фигуры, играют особенную роль при решении задачи распознавания образов. Эти дескрипторы можно получить с помощью применения вейвлет-преобразования к контурам образов.

В работе [41] разработан дескриптор для дуги, полученный с использованием вейвлет-преобразования. Этот дескриптор позволяет разложить дуги на компоненты в различных масштабах, в которых содержится глобальная и локальная информация об этих дугах. Показано, что полученный дескриптор имеет важные свойства, такие как единственность,

инвариантность, устойчивость, кратномасштабное представление и локальность.

В работе [110] предложен новый дескриптор для представления фигур образов на основе применении вейвлет-преобразования к контурам образов. Этот дескриптор был протестирован на базе из 6000 рукописных символов и в результате численных экспериментов показано, что дескрипторы, полученные на основе применения вейвлет-преобразования, являются эффективными представлениями для решения задачи распознавания образов.

1.3.2 Классификация изображений

Цель задачи классификации изображений заключается в том, чтобы определить существует ли на изображениях объект заданного класса, такого как «тигр», «лев», «волк» и т.п., либо установить относятся ли изображения к заданному классу, такому как изображения «на горах», «на море», «в небе» и т.п.

В работе [82] признаки изображений получаются на основе применения вейвлет-преобразования Добеши и моментов цветов. Полученные признаки в дальнейшем подаются на вход многослойной нейронной сети для классификации изображений самолетов.

В работе [35] разработан алгоритм, в котором используется пакет вейвлет-преобразования для извлечения признаков изображений текстур. В результате тестирования на различных базах изображений текстур показано, что вейвлеты успешно применяются для решения задачи классификации изображений текстур.

В работе [43] используется комбинация вейвлет-преобразования и нейронной сети для классификации изображений сцены войны. Проведено сопоставление результатов работы вейвлет-преобразований Хаара и Добеши и показано, что вейвлет Хаара эффективнее.

В работе [89] комбинация вейвлет-преобразования и нейронной сети также используется для классификации изображений объектов, таких как

яблоко, танк, мотоцикл и т.п.

В работе [51] предложен алгоритм, использующий комбинацию вейвлет-преобразования Добеши и многослойной нейронной сети для классификации изображений самолетов.

В работе [109] используется вейвлет-преобразование для разложения изображений. Проекции коэффициентов вейвлета низкочастотного субдиапазона на оси X и Y вычисляются для составления вектора признаков, использующихся для классификации изображений. Показано, что алгоритм способен работать для нахождения изображений в больших базах данных. Примеры изображений, использующихся для тестирования алгоритма классификации изображений в этой работе, представлены на рис. 1.9.

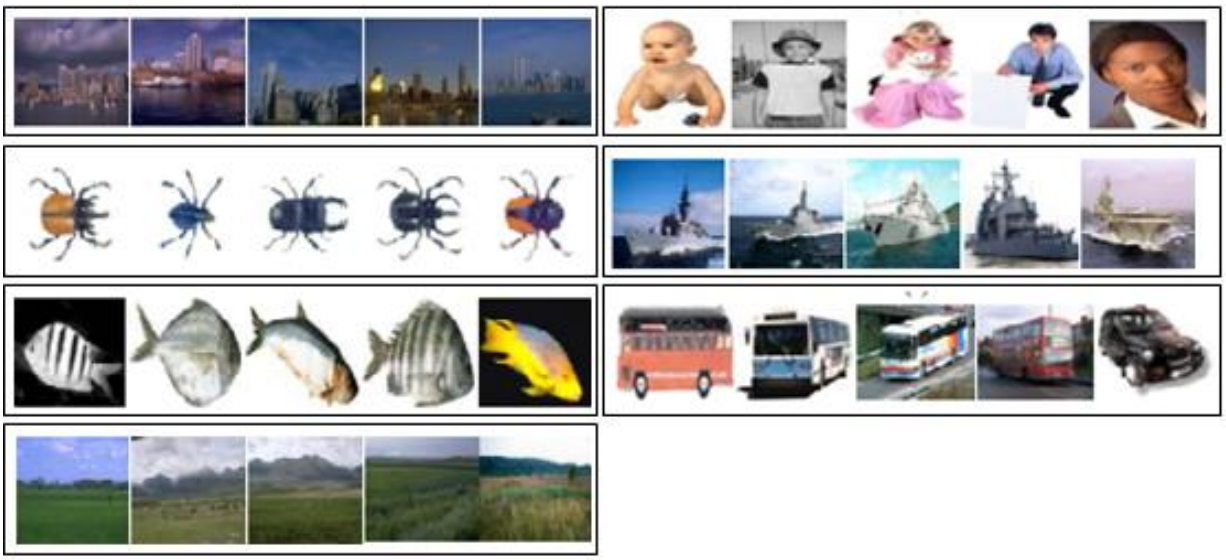


Рисунок 1.9. Примеры изображений для экспериментов работы [109]

Результаты численных экспериментов для алгоритмов вышеперечисленных работ приведены в табл. 1.2. Показано, что использование вейвлет-преобразования для выделения признаков изображения обеспечивает возможность эффективно классификации изображений. Точность классификации изображений при этом составляет 76–99,7%.

Таблица 1.2. Результаты классификации изображений различных алгоритмов

Работа	**База изображений**			**Точность**	
	число классов	**обучающая выборка**	**тестовая выборка**	**обучающая выборка (%)**	**тестовая выборка (%)**
[82]	6	150	200	98	90
[109]	7	1470	1470	96	90
[35]	30	–	–	–	99,6
	8	–	–	–	97
[43]	2	200	200	–	82,5
[89]	30	300	300	81,7	76,7
[51]	6	120	120	–	88

В работе [108] предложен алгоритм, предназначенный для быстрого нахождения нужных изображений из больших баз данных. В этом алгоритме используется вейвлет-преобразование Добеши для разложения каждой из трех цветовых компонент изображений. Вейвлет-коэффициенты низкочастотного субдиапазона являются вектором признаков изображений. В результате экспериментов показано, что алгоритм позволяет найти 100 наилучших подобных изображений из базы 10000 изображений за 3,3 секунды.

В работе [39] приведен анализ эффективности вейвлет-преобразования для извлечения признаков изображений текстур, использующихся для решения задачи их классификации. Проведены эксперименты с различными классификаторами, такими как *Naive Bayse*, *Feed Forward Neural Network* (FFNN), *K-Neareast Neighbor* (kNN), *Cascaded Forward Neural Network* (CFNN). В результате численных экспериментов показано, что вейвлет-преобразование эффективно используется для классификации изображений текстур. Точность наилучшего результата составляет 91,3%.

В работе [90] используется вейвлет-преобразование Хаара для извлечения признаков изображений, на основе которых составляются векторы из 10 главных признаков. Затем к этим векторам применяется алгоритм *K-means* для классификации изображений текстур. При тестировании алгоритм используется для нахождения изображений в базе данных по информации о заданном изображении. Показано, что алгоритм позволяет найти нужные изображения из базы данных. На рис. 1.10 представлен пример работы алгоритма.

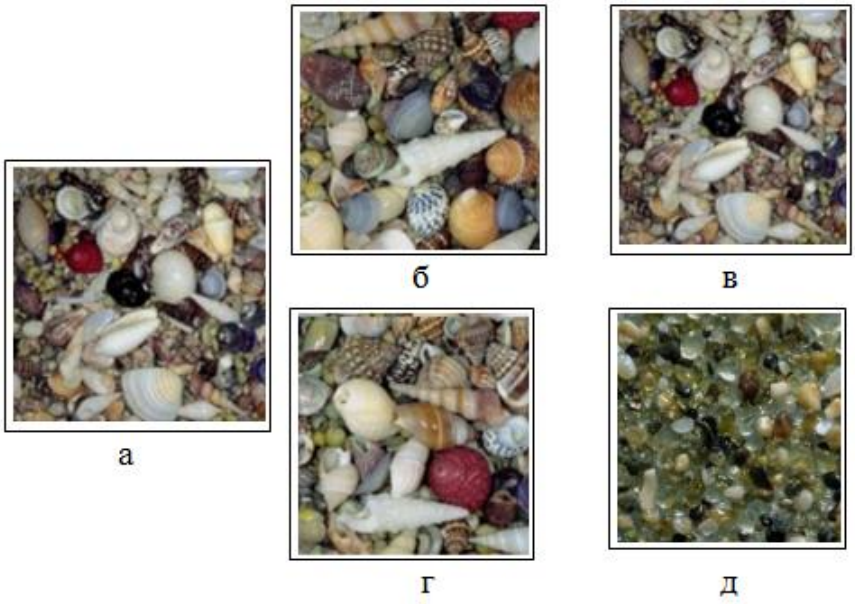


Рисунок 1.10. Пример работы алгоритма [90]: а) – исходное изображение запроса; б–д) – результат нахождения в базе данных

1.3.3 Распознавание лиц

Задача распознавания лиц на изображениях является одной из самых актуальных задач области распознавания образов. Она играет важную роль в нашей жизни и имеет широкое применение, например, может использоваться в системах паспортного контроля аэропортов и вокзалов, в иммиграционных службах, в системах видеоконтроля, расположенные в метрополитене, местах проведения зрелищных, спортивных мероприятий, вокзалах, автовокзалах, аэропортах и т.д.

В работе [68] предложен алгоритм распознавания лиц, основанный на применении комбинации преобразования Фурье и вейвлет-преобразования. В этом алгоритме к исходным изображениям лиц применяется вейвлет-преобразование для их разложения. Затем к низкочастотным изображениям применяется преобразование Фурье для получения представления изображений лиц.

В работе [59] предложено использование двумерного вейвлет-преобразования Добеши для выделения признаков лиц. Полученные признаки в дальнейшем используются в процессе распознавании лиц.

В работе [114] разработан алгоритм распознавания лиц, состоящий из двух шагов. Вначале разлагаются изображения лиц на основе применения двумерного вейвлет-преобразования, в результате которого изображения лиц представляются в виде коэффициентов разложения. Затем используется метод классификации лиц, основанный на модели *Kernel Associative Memory* (KAM), полученной на основе реконструкции изображений лиц. В этом алгоритме каждый человек имеет свою модель. Процесс распознавания лица выполняется на основе определении ошибки реконструкции изображения лица в каждой модели.

В работе [53] предложен алгоритм распознавания лиц, в котором вейвлет-преобразование Хаара используется для извлечения признаков лиц, а метод SVM – для их классификации.

В работе [107] используется двумерное вейвлет-преобразование Хаара для извлечения признаков лиц. Полученные векторы признаков применяются для классификации лиц на основе вычисления расстояния Евклида между этими векторами.

Метод *Locality Preserving Projection* (LPP) используется для представления локальной структуры изображений. Метод *Locally Discriminating Projection* (LDP) является расширенным вариантом метода LPP, который эффективно используется для извлечения признаков изображений лиц. Использование таких методов позволяет получить только

линейные признаки. В работе [106] предложен метод извлечения нелинейных признаков изображений, называемый *Wavelet based Kernel Locally Discriminating Projection* (WKLDP), который является расширенным вариантом метода LDP. Эти признаки используются далее для классификации изображений лиц.

В работе [81] разработан алгоритм распознавания лиц, в котором используется комбинация вейвлет-преобразования и МГК для извлечения признаков лиц. Полученные признаки подаются на вход многослойной нейронной сети. По результатам сопоставления с другими методами извлечения признаков лиц, применяемыми в отдельности (вейвлет-преобразованием, МГК, LDA), показано, что наилучший результат распознавания лиц получается при использовании комбинации вейвлет-преобразования и МГК.

Алгоритмы вышеперечисленных работ протестированы на различных базах лиц и частях этих баз лиц, таких как *Yale database* [24], *ORL database* [22], *FERET Databases (Release2)* [91], *XM2VTS Database* [74]. На рис. 1.11 и 1.12 представлены примеры изображений из баз данных *Yale* и *ORL*.

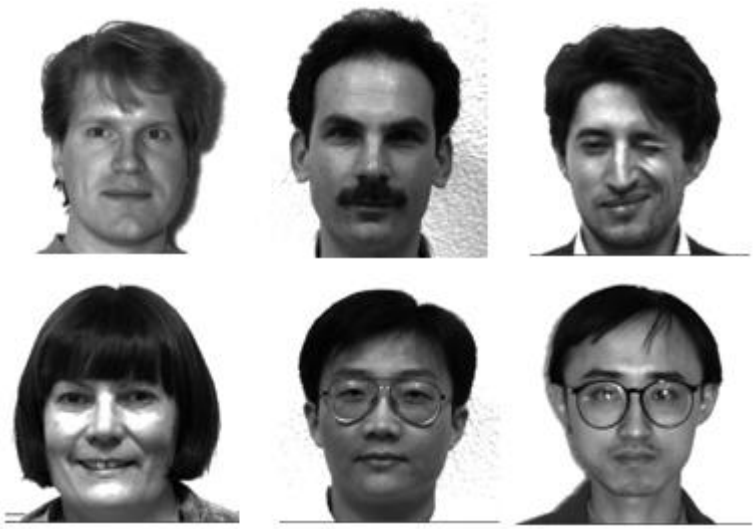


Рисунок 1.11. Примеры изображений в базе лиц Yale

Рисунок 1.12. Примеры изображений в базе лиц ORL

Результаты численных экспериментов из вышеперечисленных работ приведены в табл. 1.3. В результате показано, что точность распознавания лиц составляет 90–98,5% и доказано что вейвлет-преобразование надежно и эффективно используется для извлечения признаков, предназначенных для распознавания лиц.

Таблица 1.3. Точность распознавания лиц различными алгоритмами

Работа	**База изображений лиц**				**Точность (%)**
	название	**объекты**	**обучающая выборка**	**тестовая выборка**	
[59]	FERET	10	150	50	90
[114]	FERET	119	357	570	84,7
			476	451	91,6
	XM2VTS	295	855	295	82,2
	ORL	40	120	280	89,4
[53]	ORL	40	40	360	66,9
			80	320	86,2
			120	280	93,9
			160	240	95,4
			200	200	97,5
			240	160	98,1

Работа	База изображений лиц				Точность (%)
	название	объекты	обучающая выборка	тестовая выборка	
[107]	ORL	40	–	–	89,4
[106]	ORL	40	240	160	98,5
[81]	Yale	15	164	1	90,3

Показано, что вейвлет-преобразования эффективно применяются в различных алгоритмах для решения задач обработки изображений, таких как восстановление изображений, сжатие изображений, сегментация изображений, классификация изображений и распознавание образов. Доказано, что использование вейвлет-преобразований пригодно для извлечения признаков образов и дает возможность надежного и эффективного решения задач области распознавания образов.

Большой научный и практический интерес в настоящее время представляют задачи распознавания лиц, жестов, печатных и рукописных символов, текстов и т.д. Распознавание символов может использоваться в системах OCR, в системах чтения номеров автомобилей и т.п. Программные средства и системы, использующие распознавание символов, продолжают развиваться в направлении повышения точности и скорости распознавания.

Таким образом, применение вейвлет-преобразования является перспективным способом для решения задачи разработки новых алгоритмов, предназначенных для распознавания символов и текстов.

1.4 Цель и задачи исследования

Целью диссертационной работы является разработка алгоритмов на основе вейвлет-преобразования, метода главных компонент и нейронных сетей, способных распознавать символы разных шрифтов и фрагменты текстов.

Для достижения поставленной цели необходимо решить следующие **основные задачи.**

1. Разработать алгоритм распознавания символов на основе вейвлет-преобразования, метода главных компонент и нейронных сетей.

2. Разработать способ построения классификатора для распознавания символов на основе нейронных сетей.

3. Создать алгоритм распознавания фрагментов печатных текстов на основе разработанного алгоритма распознавания символов.

4. Осуществить апробацию созданных в диссертационной работе алгоритмов на задачах распознавания символов и фрагментов печатных текстов на изображениях.

1.5 Основные результаты и выводы по главе 1

В данной главе проведен анализ задач цифровой обработки изображений и их областей применения. Сделан аналитический обзор систем распознавания текста и подходов к распознаванию символов. Показаны преимущества и недостатки каждого подхода. Благодаря своим преимуществам признаковый подход выбран для дальнейшей разработки алгоритмов, предназначенных для распознавания символов разных шрифтов с высоким быстродействием. Проведен анализ методов для выделения признаков изображения. На основе проведенного анализа вейвлет-преобразование было выбрано для выделения признаков изображения.

Кроме того, в данной главе сделан аналитический обзор методов и алгоритмов для решения задач цифровой обработки изображений и распознавания образов, основанных на вейвлет-преобразовании. Показано, что при использовании вейвлет-преобразования для выделения признаков изображений, точность алгоритмов классификации изображений составляет 76–99,7%, а точность алгоритмов распознавания лиц 90–98,5%.

Было принято решение об актуальности исследования возможности применения вейвлет-преобразования при разработке алгоритмов, предназначенных для распознавания символов и фрагментов печатных текстов.

Глава 2. Применение вейвлет-преобразования, метода главных компонент и нейронных сетей для распознавания символов и фрагментов печатных текстов

В данной главе предложен и описан способ построения классификатора для распознавания символов, основанный на нейронных сетях. Разработан и описан алгоритм распознавания символов, основанный на вейвлет-преобразовании, методе главных компонент и нейронных сетях. Создан и описан алгоритм распознавания фрагментов текстов, основанный на разработанном алгоритме распознавания символов и способе выделения символов из фрагмента текста.

2.1 Предложенный алгоритм распознавания символов

Распознавание образов является одной из самых изученных задач в таких областях как цифровая обработка изображений, компьютерное зрение, биометрия, создание интеллектуальных систем безопасности и контроля доступа и т.п. Тем не менее, в области распознавания образов продолжают представлять большой научный и практический интерес такие задачи как распознавание лиц, жестов, отпечатков пальцев, печатных и рукописных символов, печатных текстов и т.д.

В данной главе предложен новый **алгоритм распознавания символов,** основанный на вейвлет-преобразовании, методе главных компонент и многослойных нейронных сетях. Алгоритм распознавания символов работает следующим образом.

Шаг 1. Обучение нейронных сетей.

Шаг 1.1. Выделение характерных признаков символов из обучающей выборки на основе применения вейвлет-преобразования.

Шаг 1.2. Уменьшение размерности векторов выделенных признаков методом главных компонент.

Шаг 1.3. Обучение нейронных сетей полученными векторами выделенных признаков символов.

Шаг 2. Распознавание символа.

Шаг 2.1. Выделение характерных признаков распознаваемого символа из тестовой выборки на основе применения вейвлет-преобразования.

Шаг 2.2. Уменьшение размерности вектора выделенных признаков методом главных компонент.

Шаг 2.3. Распознавание символа обученными нейронными сетями.

2.1.1 Выделение признаков изображений символов

Главной задачей в каждом виде обработки изображения является нахождение эффективного представления, позволяющего отобразить изображение в компактной форме. В современной теории и практике обработки сигналов активно используются преобразования специального вида – вейвлеты, показавшие свою эффективность в спектральном анализе сигналов [10, 95].

Вейвлет-преобразования успешно применяются для извлечения признаков изображений при решении таких задач как классификация изображений текстур [28, 35, 67, 75, 96], классификация изображений объектов, таких как: здание, рыба, трава, люди, автомобиль, самолет, помидор, яблоко и т.д. [5, 14, 16, 51, 82, 90, 109, 113].

Кроме этого, комбинация вейвлет-преобразования и МГК показала свою эффективность для решения задач распознавания образов. В работах [7, 8] предложен алгоритм, основанный на методе Виолы-Джонса, вейвлет-преобразовании и МГК для распознавания множества лиц на изображениях и видеопоследовательностях. Результаты проведенных экспериментов на основе этого алгоритма показали, что точность распознавания составляет ~98,4%.

В работе [15] предложен алгоритм, основанный на алгоритме *CAMShift*,

методе Виолы-Джонса, вейвлет-преобразовании и МГК для распознавания жестов на видеопоследовательности. Точность результата распознавания этого алгоритма составляет ~94,6%.

Скорость обработки указанных двух алгоритмов достаточно быстрая, 7–14 кадров в секунду для алгоритма распознавания множества лиц, 25–30 кадров в секунду для алгоритма распознавания жестов. Эта скорость позволяет программам, реализованным на основе разработанных алгоритмов, работать в режиме реального времени.

В работах [7, 8, 15] проводилось исследование зависимости эффективности результатов распознавания лиц и жестов от типа вейвлетов и показано, что при использовании комбинации вейвлет-преобразования Хаара и МГК получены наилучшие результаты распознавания. Кроме того, для вейвлет-преобразования Хаара характерны простота реализации и высокая скорость вычисления [12, 13].

Таким образом, можно сделать вывод о целесообразности применения вейвлет-преобразования Хаара и МГК для решения задач распознавания символов и фрагментов печатных текстов с приемлемой скоростью обработки.

Вейвлет Хаара является базисным вейвлет-преобразованием, с помощью которого можно разложить одномерный дискретный сигнал $f = (f_1, f_2, ..., f_N)$ на два компонента равного размера. Первый компонент называется средним или аппроксимацией (*approximation*), а второй – различием (*difference*) или детализацией (*detail*). Значения среднего подсигнала $a^1 = (a_1, a_2, ..., a_{N/2})$ на первом уровне разложения вычисляются по следующей формуле [6]:

$$a_n = \frac{f_{2n-1} + f_{2n}}{\sqrt{2}}, n = 1, 2, 3, ..., N / 2, \qquad (2.1)$$

а значения детализирующего подсигнала $d^1 = (d_1, d_2, ..., d_{N/2})$ на этом же уровне разложения определяются следующим образом:

$$d_n = \frac{f_{2n-1} - f_{2n}}{\sqrt{2}}, n = 1,2,3,..., N/2. \quad (2.2)$$

В результате получаются два новых сигнала: $a = \{a_n\}$ и $d = \{d_n\}, n = 1,..., N/2$. Первый сигнал является огрубленной версией исходного дискретного сигнала. Во втором сигнале содержится информация, необходимая для восстановления исходного дискретного сигнала.

$$f_{2n-1} = \frac{a_n + d_n}{\sqrt{2}}, \quad (2.3)$$

$$f_{2n} = \frac{a_n - d_n}{\sqrt{2}}. \quad (2.4)$$

Рассмотрим простой пример для того, чтобы понять как работает двумерное вейвлет-преобразование Хаара. Пусть имеется следующий двумерный сигнал

$$I = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \\ 5 & 6 & 7 & 8 \\ 8 & 7 & 6 & 5 \end{bmatrix}.$$

При применении одномерного вейвлет-преобразования Хаара к первой строке, значения коэффициентов аппроксимации вычисляются по формуле (2.1)

$$a_1 = \frac{1+2}{\sqrt{2}} \text{ и } a_2 = \frac{3+4}{\sqrt{2}},$$

а значения коэффициентов различия вычисляются по формуле (2.2)

$$d_1 = \frac{1-2}{\sqrt{2}} \text{ и } d_2 = \frac{3-4}{\sqrt{2}}.$$

Аналогично к остальным строкам матрицы I применяется одномерное вейвлет-преобразование Хаара. В результате получается новая матрица, первые два столбца которой содержат значения коэффициентов аппроксимации каждой строки, а последующие два столбца – значения коэффициентов различия.

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \\ 5 & 6 & 7 & 8 \\ 8 & 7 & 6 & 5 \end{bmatrix} \xrightarrow{\text{одномерное преобразование Хаара на строках}} \frac{1}{\sqrt{2}} \left[\begin{array}{ccccc} 3 & 7 & : & -1 & -1 \\ 7 & 3 & : & 1 & 1 \\ 11 & 15 & : & -1 & -1 \\ 15 & 11 & : & 1 & 1 \end{array}\right].$$

Затем к столбцам полученной матрицы применяется одномерное вейвлет-преобразование Хаара. В результате получается преобразованная матрица первого уровня.

$$\frac{1}{\sqrt{2}} \left[\begin{array}{ccccc} 3 & 7 & : & -1 & -1 \\ 7 & 3 & : & 1 & 1 \\ 11 & 15 & : & -1 & -1 \\ 15 & 11 & : & 1 & 1 \end{array}\right] \xrightarrow{\text{одномерное преобразование Хаара на столбцах}} \left[\begin{array}{ccccc} 5 & 5 & : & 0 & 0 \\ 13 & 13 & : & 0 & 0 \\ .. & .. & .. & .. & .. \\ -2 & 2 & : & -1 & -1 \\ -2 & 2 & : & -1 & -1 \end{array}\right].$$

Преобразованная матрица делится на четыре подматрицы

$$LL = \begin{bmatrix} 5 & 5 \\ 13 & 13 \end{bmatrix}, HL = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, LH = \begin{bmatrix} -2 & 2 \\ -2 & 2 \end{bmatrix} \text{ и } HH = \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix}.$$

Таким образом, двумерное вейвлет-преобразование Хаара состоит из поочередного применения одномерного вейвлет-преобразования Хаара к строкам и столбцам этой матрицы. В результате исходное изображение делится на четыре подматрицы (квадранта) равного размера. На рис. 2.1 представлены четыре квадранта преобразованного изображения со стандартными обозначениями: LL, LH, HL и HH. Квадрант LL содержит низкочастотные вейвлет-коэффициенты, HH – высокочастотные вейвлет-коэффициенты (буква L означает *Low* и буква H – *High*) [18].

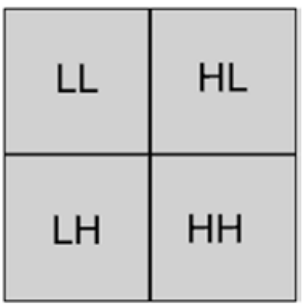


Рисунок 2.1. Полученные квадранты при применении вейвлет-преобразования

Процесс **выделения признаков символа** происходит следующим образом. Вначале область, содержащая символ, приводится к размеру 64×64 пикселя. Затем к полученному изображению применяется вейвлет-преобразование и извлекаются низкочастотные вейвлет-коэффициенты. В результате получается матрица, состоящая из 32×32 низкочастотных вейвлет-коэффициентов.

Для того чтобы выделить локальные характерные признаки символа, его изображение делится на 12 частей с одинаковым размером 32×32 пикселя (рис. 2.2). Затем к каждой части применяется вейвлет-преобразование и извлекаются низкочастотные вейвлет-коэффициенты. В результате получаются 12 матриц, каждая из которых состоит из 16×16 низкочастотных вейвлет-коэффициентов.

После этого составляется вектор характерных признаков символа, элементами которого являются все низкочастотные вейвлет-коэффициенты, полученные на предыдущих шагах. В результате получается вектор характерных признаков символа $\vec{I}$, состоящий из 32×32 + 12×16×16 = 4096 элементов (рис. 2.2).

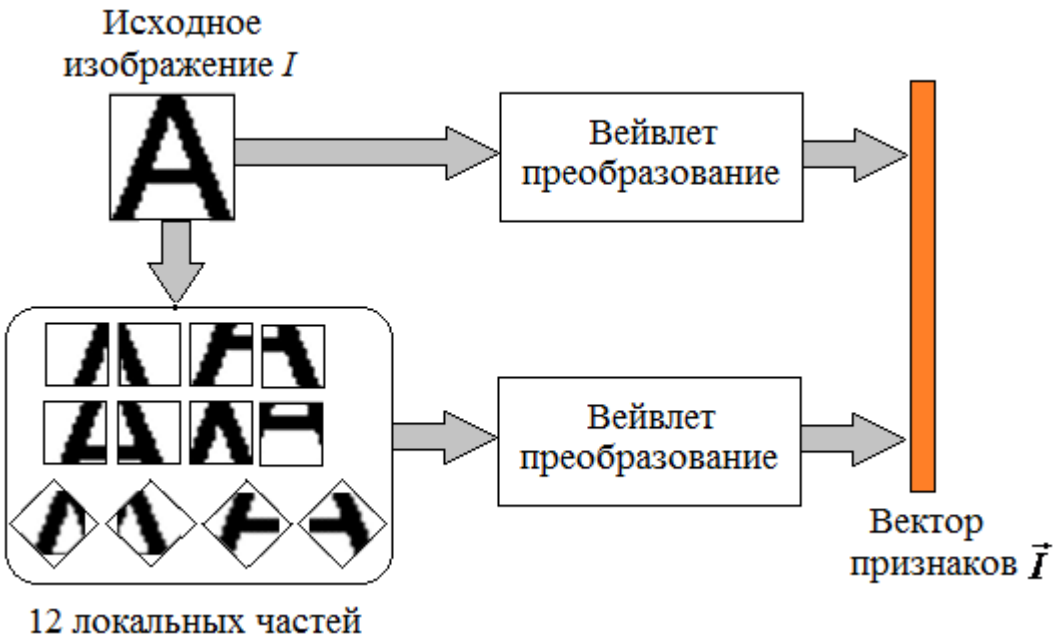


Рисунок 2.2. Выделение характерных признаков символа «А»

2.1.2 Уменьшение размерности вектора признаков

Перед подачей на входы нейронных сетей **размерность вектора признаков уменьшается**. Для решения этой задачи предлагается использование метода главных компонент (МГК).

МГК был создан с целью уменьшения размерности векторов признаков, которые используется для решения задачи распознавания образов. При решении задачи распознавания лиц, используются главные компоненты распределения лиц, представляющие собой собственные векторы ковариационной матрицы набора соответствующих изображений. Эти собственные векторы можно рассматривать как набор характерных признаков изображений лиц, их можно представить как некое приближение к изображению лица [11]. Эти векторы принято называются собственными лицами (*eigenfaces*).

МГК также является одним из распространенных и эффективных методов выделения признаков, предназначенных для решения задач распознавания образов. В работах [29, 31, 64, 83, 102, 103, 112, 117] МГК показал свои работоспособность, эффективность и надежность при решении задачи распознавания лиц. В работе [81] показано, что при распознавании лиц использование комбинации вейвлет-преобразования и МГК для извлечения признаков изображений эффективнее, чем использование только вейвлет-преобразования или использование только МГК.

Уменьшение размерности вектора признаков символа на основе МГК происходит следующим образом. Вначале создается пространство собственных символов, используя набор M изображений символов. При этом M намного меньше 4096. Создание пространства собственных символов выполняется следующим образом.

К каждому из M изображений применяется разработанный способ выделения характерных признаков. В результате получается набор векторов признаков $\vec{I}_1,...,\vec{I}_M$. Затем составляется средний вектор, значение каждого элемента которого по всем M векторам признаков вычисляется по формуле:

$$\vec{I}_{\text{ср}} = \frac{1}{M}\sum_{n=1}^{M}\vec{I}_n. \qquad (2.5)$$

Далее из каждого вектора признаков вычитается средний вектор

$$\vec{\Phi}_n = \vec{I}_n - \vec{I}_{\text{ср}},\ n = 1,...,M. \qquad (2.6)$$

После этого создается пространство, состоящее из K собственных векторов $\vec{u}_k$ ковариационной матрицы C, которые наилучшим образом описывают распределение значений M векторов признаков ($K<M$)

$$C = \frac{1}{M}\sum_{n=1}^{M}\vec{\Phi}_n\vec{\Phi}_n^T = AA^T, \quad A = \{\vec{\Phi}_1,...,\vec{\Phi}_M\}. \qquad (2.7)$$

При этом k-ый вектор $\vec{u}_k$ удовлетворяет условию максимизации следующего выражения:

$$\lambda_k = \frac{1}{M}\sum_{n=1}^{M}(\vec{u}_k^T\vec{\Phi}_n)^2 \qquad (2.8)$$

и условию ортогональности:

$$\vec{u}_l^T\vec{u}_k = \begin{cases}1, & l = k\\ 0, & \text{иначе}\end{cases}. \qquad (2.9)$$

Векторы $\vec{u}_k$ и величины λ_k представляют собственные векторы и собственные значения ковариационной матрицы C.

Для создания этого пространства вначале нужно вычислить M собственных векторов $\vec{u}_l$ матрицы C с использованием векторов $\vec{v}_i$, являющихся собственными векторами матрицы $L = A^T A$. Каждый вектор $\vec{u}_l$ вычисляется по следующей формуле:

$$\vec{u}_l = \frac{1}{M}\sum_{k=1}^{M}v_{lk}\Phi_k,\ l = 1,\ldots,M. \qquad (2.10)$$

Затем из M полученных векторов выбираются K собственных векторов, имеющих наибольшие собственные значения. Пространством собственных символов является набор выбранных K собственных векторов (рис. 2.3). Значение K определяется эмпирическим способом.

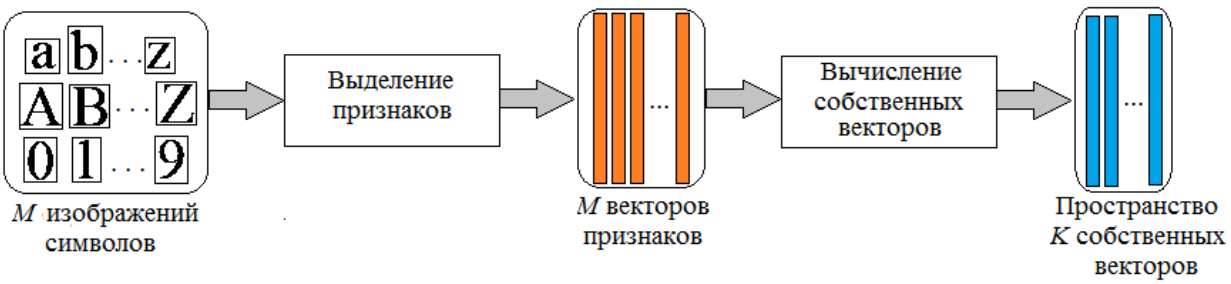


Рисунок 2.3. Создание пространства собственных символов

После того как создано пространство собственных символов, уменьшение размерности вектора характерных признаков символа $\vec{I}_{\text{вх}}$ осуществляется следующим образом.

Вначале вектор признаков символа разлагается по K имеющимся собственным символам $\vec{u}_i$ и вычисляются соответствующие коэффициенты разложения, определяющиеся по формуле:

$$w_i = \vec{u}_i^T (\vec{I}_{\text{вх}} - \vec{I}_{\text{ср}}), i = 1,...,K. \quad (2.11)$$

Затем составляется вектор, описывающий вклад каждого собственного символа в представление входного вектора признаков символа:

$$\vec{\Omega}^T = \{w_1,...,w_K\}. \quad (2.12)$$

В результате уменьшения размерности получается новый вектор признаков символа $\vec{\Omega}$, состоящий из K элементов. При этом K намного меньше 4096 (рис. 2.4).

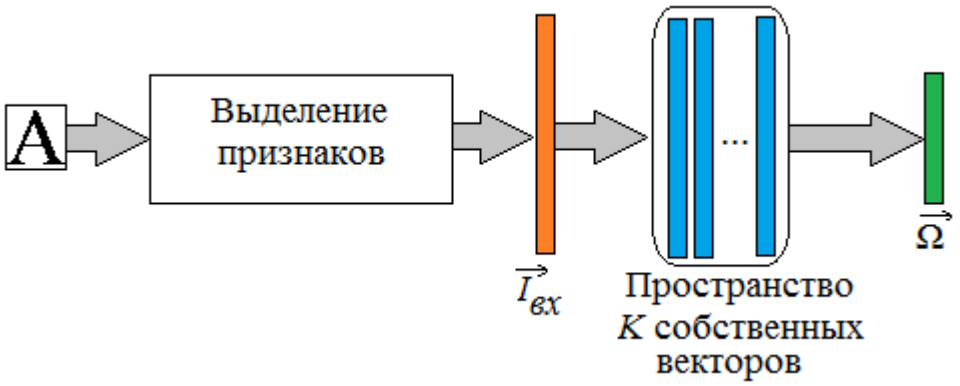


Рисунок 2.4. Уменьшение размерности вектора признаков символа

2.1.3 Распознавание символов нейронными сетями

В качестве классификатора предлагается использование многослойных нейронных сетей, поскольку они обеспечивают высокую скорость и точность распознавания [55]. В работе [58] предложен алгоритм распознавания лиц, основанный на многослойных нейронных сетях. На основе идеи этого алгоритма в данной диссертации предложен **способ построения классификатора** для распознавания символов на основе многослойных нейронных сетей.

Особенностью данного подхода является создание для каждого символа обучающей выборки специальной нейронной сети, обучаемой алгоритмом с обратным распространением ошибки. Входом каждой нейронной сети является вектор характерных признаков $\vec{\Omega}$ (K элементов). Количество нейронов скрытого слоя равно 70% количества нейронов входного слоя. Выходной слой имеет только один нейрон, возвращающий значение в пределах от 0 до 1. Использование специальной нейронной сети для каждого символа обучающей выборки позволяет ускорить процесс обучения нейронной сети. Структура специальной нейронной сети представлена на рис. 2.5.

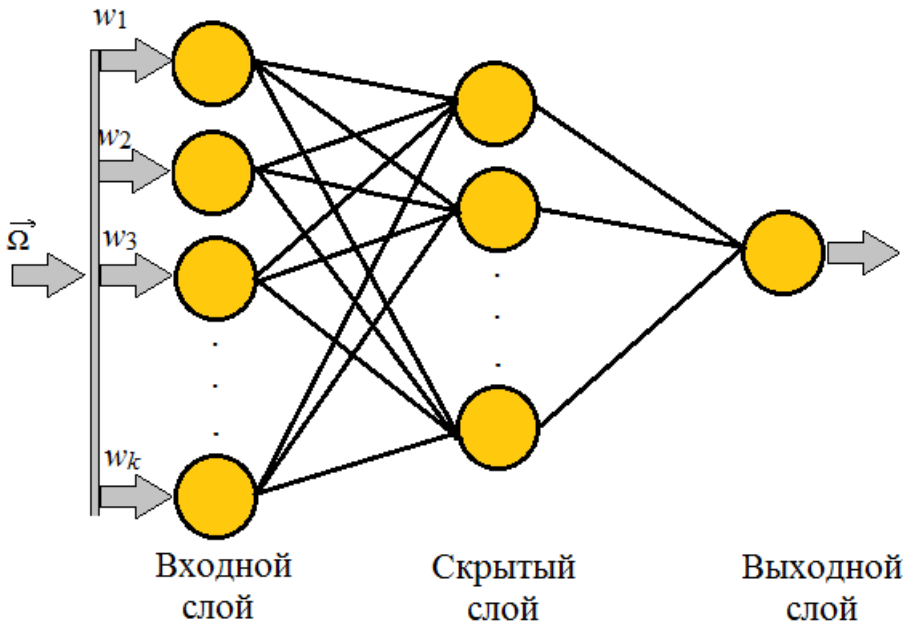


Рисунок 2.5. Структура специальной многослойной нейронной сети

Каждая нейронная сеть определяет степень близости распознаваемого символа к только одному из символов обучающей выборки. Распознавание входного символа нейронными сетями происходит следующим образом. Вначале извлекается вектор признаков символа и уменьшается его размерность. Затем полученный вектор признаков подается на входы всех обученных нейронных сетей. Входной символ распознается как символ обучающей выборки, нейронная сеть которого возвращает наибольшее значение (рис. 2.6).

Кроме того, использование специальной нейронной сети для каждого символа обучающей выборки обеспечивает возможность рассмотрения второго варианта результата распознавания. Вторым предположением является символ обучающей выборки, нейронная сеть которого возвращает второе наибольшее значение. Преимущество рассмотрения двух вариантов распознавания проявляется в случае распознавания похожих по написанию символов, таких как {c, C}, {o, O}, {p, P}, {s, S}, {u, U}, {v, V}, {w, W}, {x, X} и {z, Z} (рис. 2.6).

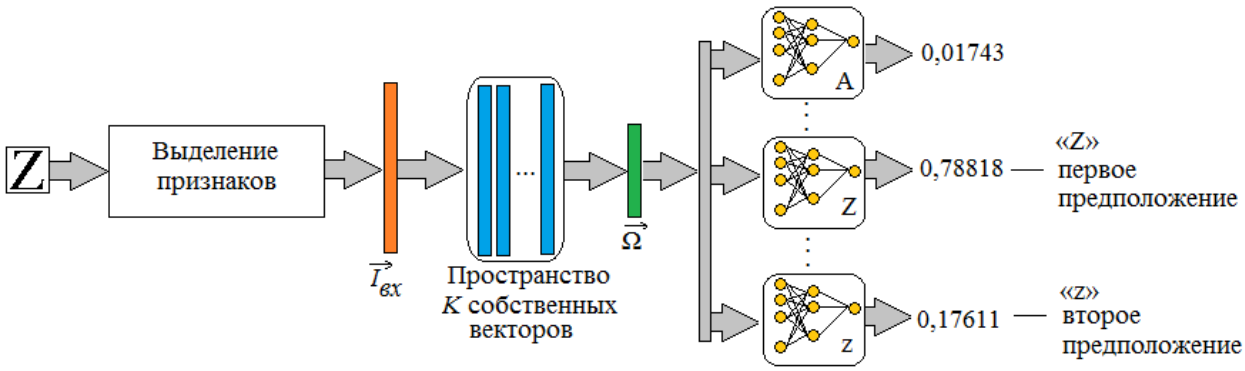


Рисунок 2.6. Распознавание символа нейронными сетями

2.2 Предложенный алгоритм распознавания фрагментов печатных текстов

Для выполнения процесса распознавания фрагмента печатного текста необходимо решить задачу выделения символов из блока текста. Эти выделенные символы являются входами процесса распознавания символов.

На основе разработанного алгоритма распознавания символов, в данной диссертации предложен новый **алгоритм распознавания фрагментов печатных текстов**. Алгоритм состоит из следующих шагов.

Шаг 1. Выделение символов из фрагмента текста.

Шаг 1.1. Поворот наклонного изображения фрагмента текста.

Шаг 1.2. Выделение строк из фрагмента текста.

Шаг 1.3. Выделение слов из строк.

Шаг 1.4. Выделение символов из слов.

Шаг 2. Распознавание выделенных символов.

Шаг 2.1. Извлечение характерных признаков выделенных символов на основе применения вейвлет-преобразования.

Шаг 2.2. Уменьшение размерности векторов извлеченных признаков методом главных компонент.

Шаг 2.3. Распознавание символов обученными нейронными сетями.

2.2.1 Выделение символов из фрагмента текста

В рамках данной диссертационной работы, главной задачей является эффективность распознавания символов фрагмента печатного текста. Таким образом, в данной работе не рассматриваются задача выделения таблиц, блоков текста и изображений из содержания документа, а также задача выделения абзацев из блока текста. В работе [98] проведен обзор различных методов сегментации символов, таких как *White Space and Pitch* (WSP), *Projection Analysis* (PA), *Connected Component Analysis* (CCA) и других и показано, что метод PA является эффективным, простым и быстрым для выделения печатных символов. Предполагается, что имеется бинарное изображение фрагмента текста. Процесс выделения символов из фрагмента текста осуществляется следующим образом.

Шаг 1. Поворот наклонного изображения фрагмента текста: Символы, использующиеся при обучении, расположены вертикально в горизонтальных строках. Однако, при сканировании строки фрагмента текста могут быть расположены не горизонтально. Таким образом, для того чтобы

горизонтально расположить строки изображение фрагмента текста необходимо повернуть. Для решения этой задачи используется проекция изображения на ось Y (рис. 2.7).

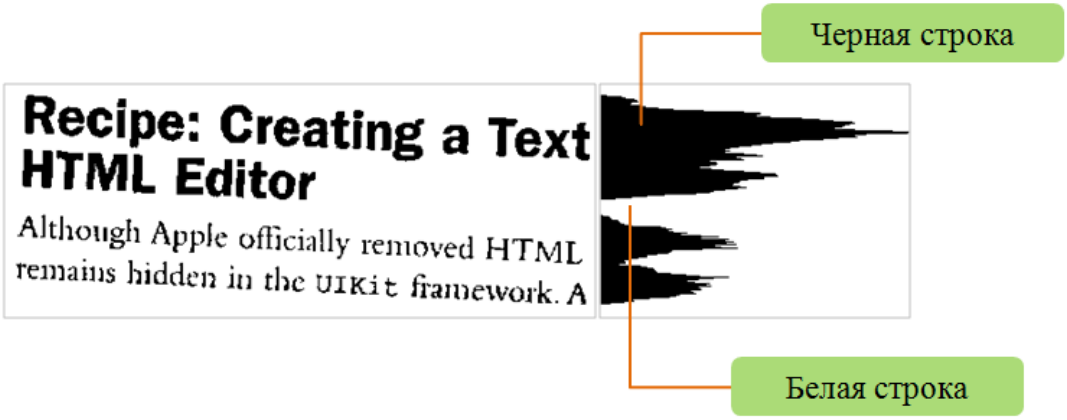


Рисунок 2.7. Пример наклонного текста и его проекция на ось Y

Вводятся следующие термины для горизонтальной проекции: «белая строка» – строка, на которой нет ни одной черной точки; «черная строка» – строка, на которой расположена хотя бы одна черная точка. Поворот изображения фрагмента текста заключается в нахождении повернутого изображения, проекция которого имеет наибольшее количество «белых строк». Пример результата поворота наклонного изображения фрагмента текста представлен на рис. 2.8.

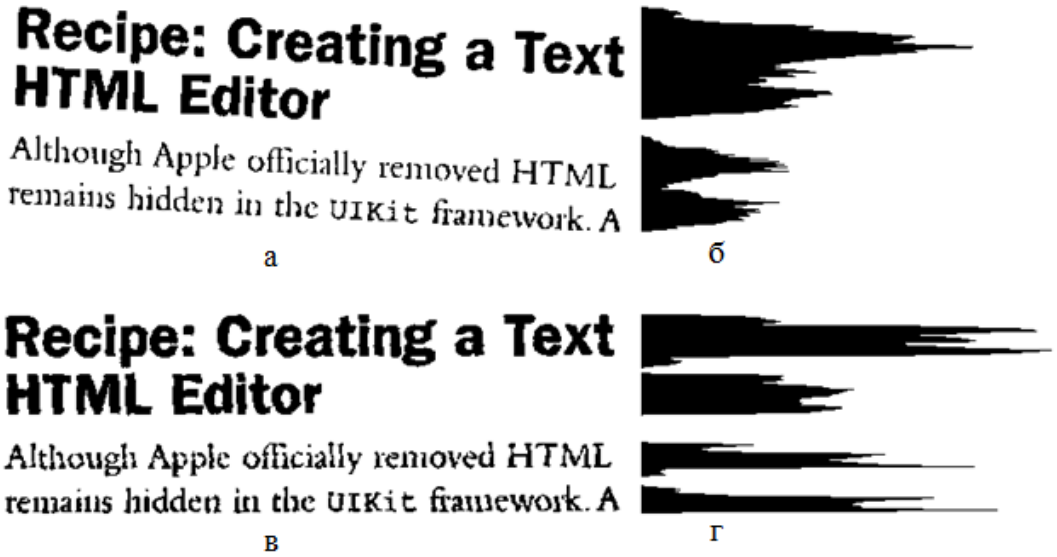


Рисунок 2.8. Пример результата поворота наклонного текста: а, б) исходное изображение текста и его проекция на ось Y; в, г) повернутое изображение и его проекция на ось Y

Шаг 2. Выделение строк: На этом шаге строки фрагмента текста выделяются на основе использования проекции повернутого изображения на ось *Y*, полученного на первом шаге. Область строки определяется последовательностью только «черных строк». Верхняя и нижняя границы строки определяются относительно этой области. Пример выделения строк фрагмента текста представлен на рис. 2.9.

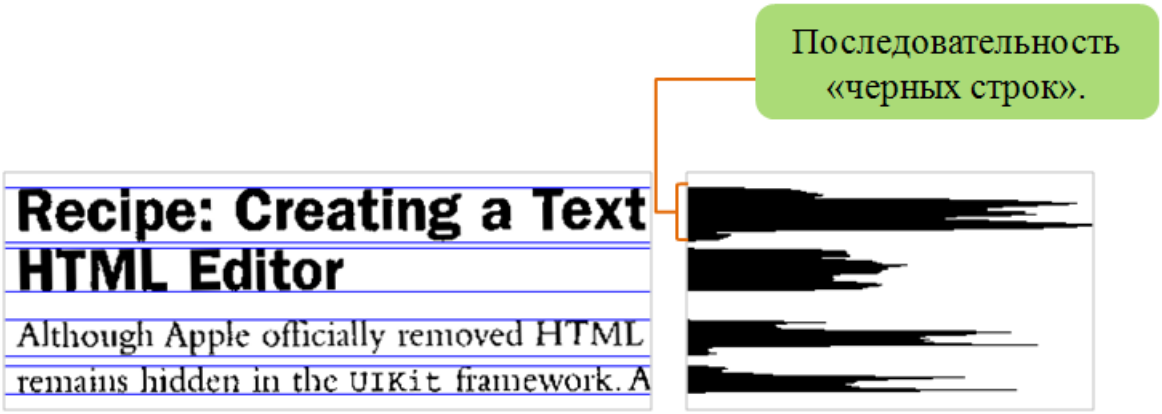


Рисунок 2.9. Результат выделения строк проекцией на ось Y

Шаг 3. Выделение слов: На этом шаге слова каждой из строк, полученных во втором шаге, выделяются на основе применения проекции этой строки на ось *X* (рис. 2.10).

Вводятся следующие термины для вертикальной проекции: «белый столбец» – столбец, на котором нет ни одной черной точки; «черный столбец» – столбец, на котором расположена хотя бы одна черная точка. На основе результатов эмпирического способа «пробел» в строке определяется последовательностью только «белых столбцов», количество которых больше чем 8. Области слов выделяются на основе информации о полученных пробелах. Левая и правая границы слов определяются в соответствии с этими областями. Пример результата выделения слов в строке представлен на рис. 2.10.

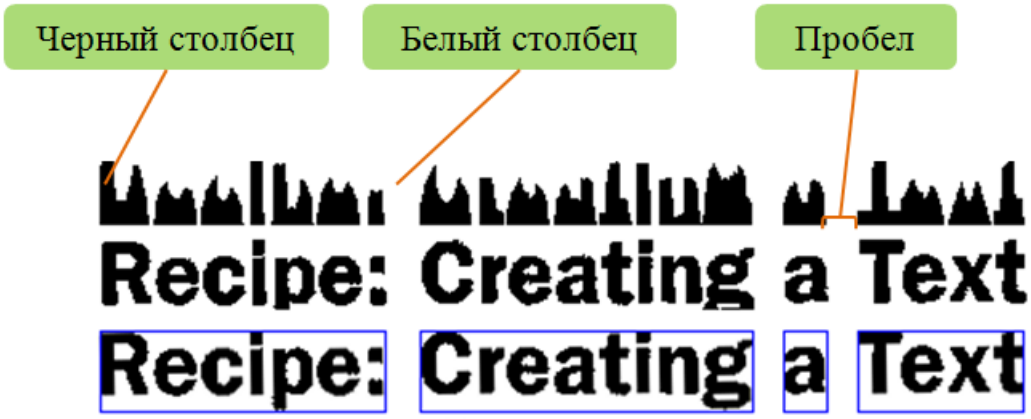


Рисунок 2.10. Результат выделения слов строки проекцией на ось X

Шаг 4. Выделение символов: На этом шаге выделяются символы каждого из слов, полученных на третьем шаге. Для описания алгоритма, вводится следующий термин: «соединяющийся столбец» – столбец, на котором расположена последовательность черных точек, количество которых больше или равно 1, соединяющаяся с черными точками соседних столбцов. Примеры этих «соединяющихся столбцов» представлены на рис. 2.11

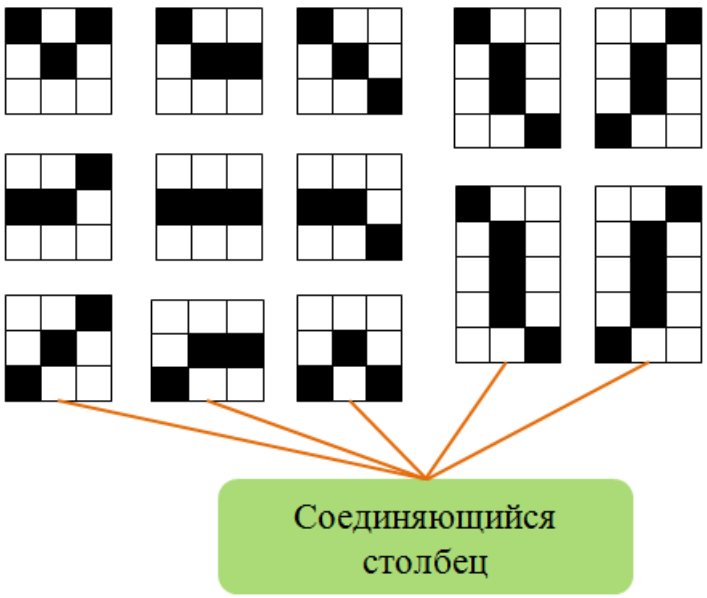


Рисунок 2.11. Примеры «соединяющихся столбцов»

Выделение символов из слова осуществляется следующим образом. Вначале слово сканируется слева-направо и обнаруживается первая

последовательность «соединяющихся столбцов». Затем определяются левая и правая границы возможной области первого символа. Полученная возможная область сканируется сверху-вниз и Y-координата первой черной точки является верхней границей области символа. Затем полученная область сканируется снизу-вверх и Y-координата первой черной точки является нижней границей области символа. Сканирование слова слева-направо продолжается, пока все содержащиеся в нем символы не будут выделены. Пример выделения символов слов «*Recipe:*» представлен на рис. 2.12.

Рисунок 2.12. Пример выделения символов слова «Recipe:»

2.2.2 Распознавание фрагмента текста

Таким образом, процесс распознавания фрагмента текста состоит из двух шагов. Вначале выделяются символы из фрагмента текста (раздел 2.2.1) и в результате получается набор областей выделенных символов. Затем каждый из выделенных символов распознается с использованием процесса распознавания символов (раздел 2.1).

Распознавание каждого выделенного символа происходит следующим образом. Вначале извлекается вектор характерных признаков символа. Затем осуществляется сокращение размерности извлеченного вектора признаков. Полученный вектор с меньшей размерностью подается на входы обученных

нейронных сетей для распознавания выделенного символа. Схема процесса распознавания фрагмента печатного текста представлена на рис. 2.13.

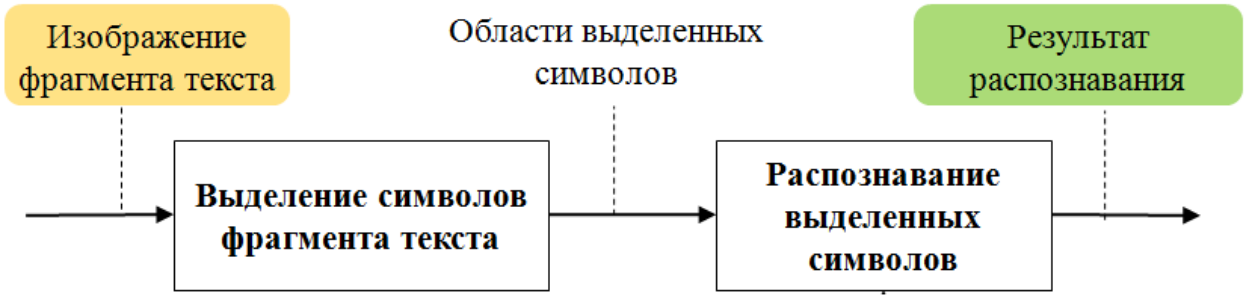


Рисунок 2.13. Схема процесса распознавания печатных текстов

2.2.3 Распознавание похожих по написанию символов

Существуют похожие по написанию символы, такие как {c, C}, {o, O}, {p, P}, {s, S}, {u, U}, {v, V}, {w, W}, {x, X} и {z, Z}. Эти символы зачастую неправильно распознаются. На рис. 2.14 приведен пример представления букв «s», «S» после того как размеры их изображений увеличиваются до одинакового размера 64×64 пикселя. Таким образом, при распознавании фрагмента печатного текста необходимо выполнить дополнительный процесс отдельного распознавания похожих по написанию символов.

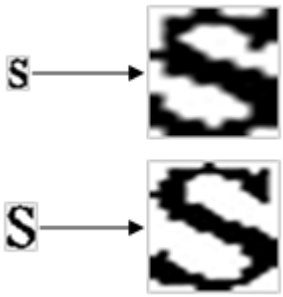


Рисунок 2.14. Изображения области букв «s», «S» и их представления после увеличения размера до 64×64 пикселя

Строки текста обычно содержат следующие характеристики: верхняя, главная, основная и нижняя границы; верхняя, средняя и нижняя области

(рис. 2.15).

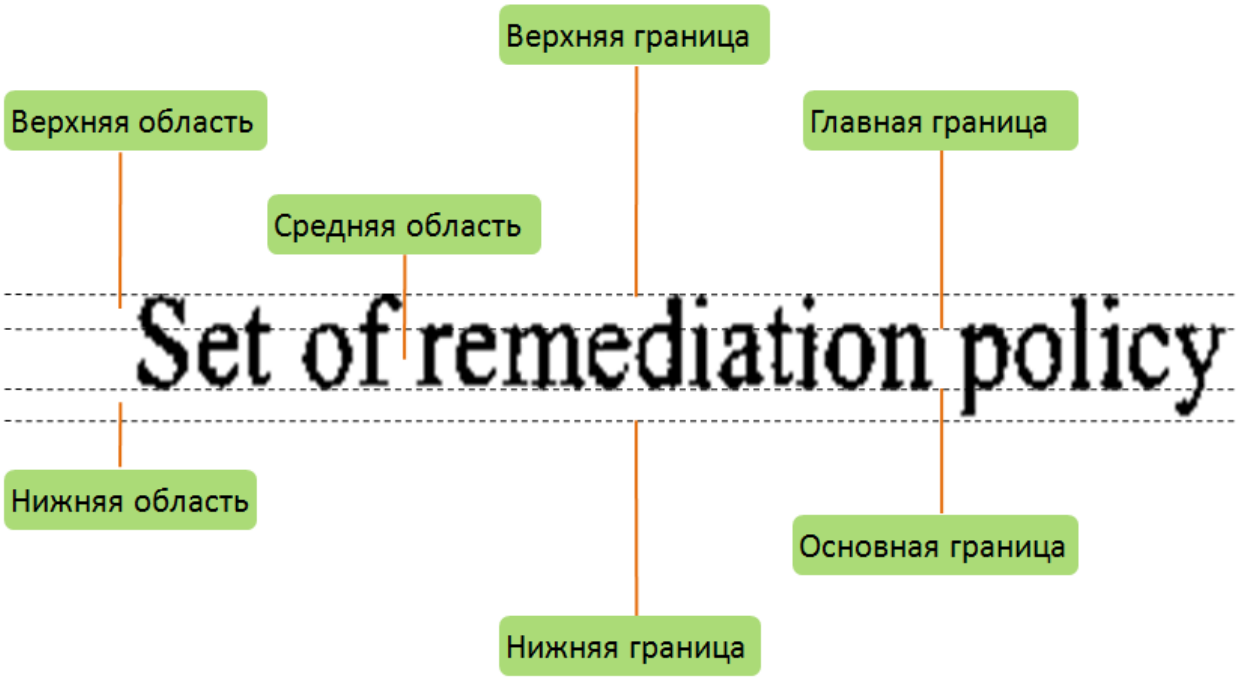


Рисунок 2.15. Характеристики строки текста

Таким образом, области символов можно разделить на четыре типа, примеры которых представлены на рис. 2.16. Первым типом является область символа, имеющая верхнее и нижнее свободное пространство. Область символа второго типа содержит только верхнее свободное пространство, а область символа третьего типа имеет только нижнее свободное пространство. Четвертым типом является область символа, которая не имеет ни верхнего, ни нижнего свободного пространства.

Для вышеперечисленных похожих по написанию символов создается дополнительное пространство собственных символов и соответствующие им нейронные сети. Процесс распознавания фрагмента печатного текста с учетом отдельного распознавания похожих по написанию символов осуществляется следующим образом.

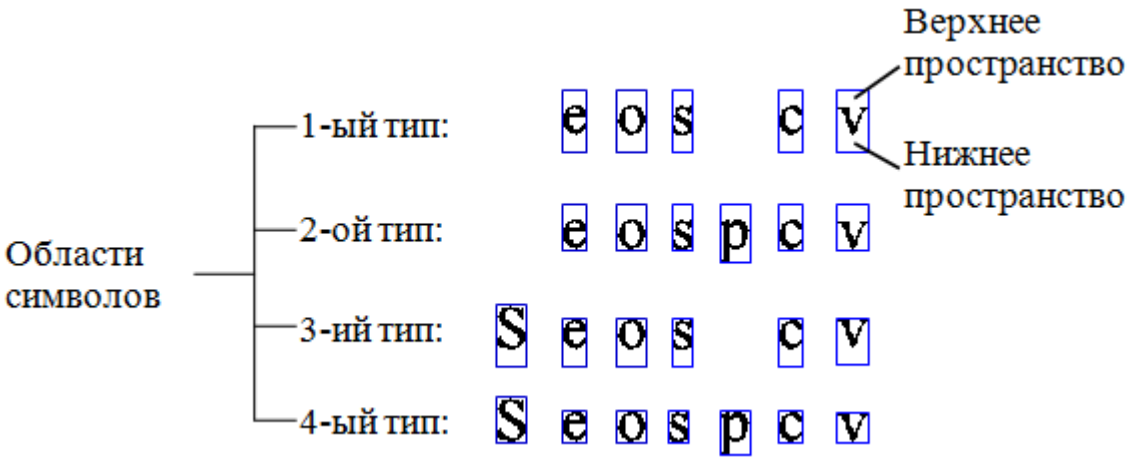


Рисунок 2.16. Примеры четырех типов областей символов

Вначале выполняются все шаги распознавания фрагмента печатного текста. Если в результате распознавания первое и второе предположения составляют пару из вышеперечисленных похожих символов, то выделяется область символа со свободным пространством. Затем к выделенной области символа применяется вейвлет-преобразование для извлечения признаков символа. После этого происходит уменьшение размерности полученного вектора признаков символа и он подается на входы обученных нейронных сетей для распознавания. Схема процесса отдельного распознавания похожих по написанию символов представлена на рис. 2.17

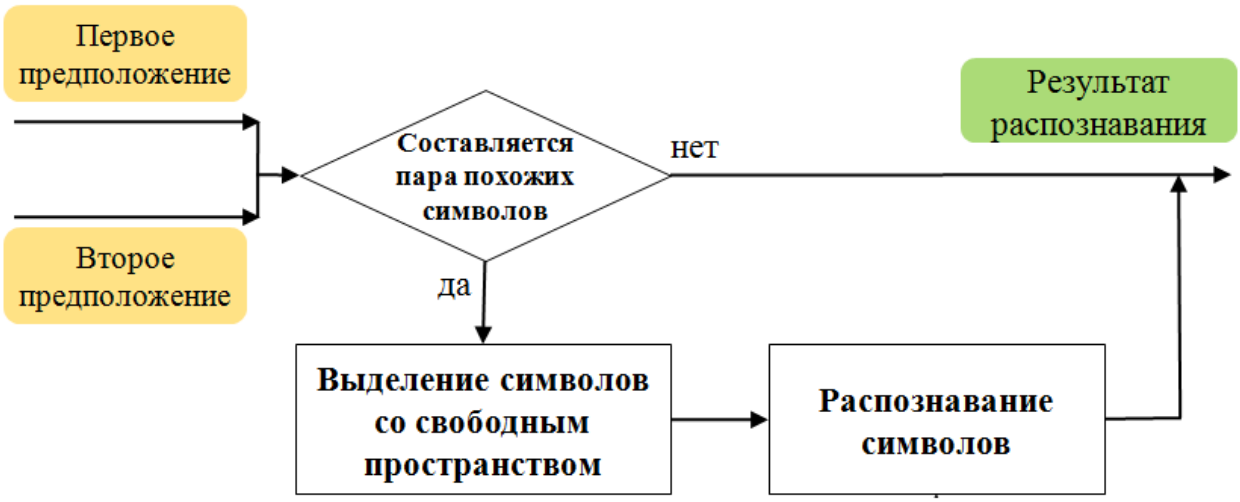


Рисунок 2.17. Схема процесса отдельного распознавания похожих по написанию символов

2.3 Основные результаты и выводы по главе 2

В данной главе приведено подробное описание предложенного способа построения классификатора для распознавания символов, реализованного алгоритма распознавания символов и разработанного алгоритма распознавания фрагментов печатных текстов.

Впервые предложен способ построения классификатора для распознавания символов, основанный на многослойных нейронных сетях. Особенностью данного подхода является создание для каждого символа обучающей выборки специальной нейронной сети, обучаемой алгоритмом с обратным распространением ошибки. Каждая нейронная сеть определяет степень близости распознаваемого символа к только одному из символов обучающей выборки.

Разработан новый алгоритм распознавания символов, основанный на вейвлет-преобразовании, методе главных компонент и нейронных сетях. В разработанном алгоритме вейвлет-преобразование применяется для выделения признаков символов, метод главных компонент осуществляет сокращение размерности векторов признаков, а нейронные сети используются в качестве классификатора.

Создан оригинальный алгоритм распознавания фрагментов печатных текстов, основанный на способе выделения символов из фрагмента текста и разработанном алгоритме распознавания символов.

Глава 3. Разработанное программное обеспечение для распознавания символов и фрагментов печатных текстов

В данной главе описываются программные средства, разработанные для реализации алгоритмов распознавания символов и фрагментов печатных текстов, описанных в главе 2. Проводится анализ инструментальных библиотек по обработке изображений и осуществляется выбор средств для разработки ПО.

Приведены основные переменные и методы реализующих классов для распознавания символов и фрагментов печатных текстов. Для конечных пользователей предусмотрено два варианта интерфейса. Первый вариант предназначен для исследователей, а второй вариант – для обычных пользователей.

3.1 Выбор средств разработки

В настоящее время существует ряд платформ, использующихся для реализации ПО. Широкое распространение получили такие платформы как .Net и Java. При этом программы на платформе Java разработаны на языке программирования Java, а программы на платформе .Net могут реализовываться на различных языках программирования, таких как Visual C++, Visual C#, Visual Basic .Net и другие.

Таким образом, представляет интерес рассмотреть следующие языки программирования: Java, Visual C++ и Visual C#. Эти перечисленные языки являются объектно-ориентированными языками программирования.

В работе [66] проведено сравнение между этими языками программирования и показано, что у языков Java и Visual C# имеются некоторые преимущества перед языком Visual C++, например в языке Visual C++, когда используется ключевое слово *new*, данные нового экземпляра класса создаются в стеке. Если позже забыли использовать ключевое слово *delete*, то это приводит к возникновению проблемы «утечки памяти». У

языков Visual C# и Java нет такой проблемы из-за того, что оба языка имеют встроенный сборщик мусора.

В работе [66] также анализируются сходство и различие языков Visual C# и Java. Сделан вывод о том, что если время разработки ограничено и программные средства будут работать на различных операционных системах, то язык Java является идеальным. А если время разработки не ограничено и программные средства будут работать только на Windows, то использование языка C# является превосходным решением.

В настоящее время существуют многие открытые платформы и библиотеки по обработке изображений, наиболее распространенными из них являются платформы AForge.NET и Accord.NET, библиотеки OpenCV и Emgu CV.

AForge.NET является общедоступной платформой (на языке C#), разработанным для разработчиков и исследователей в областях компьютерного зрения и искусственного интеллекта. Платформа состоит из набора библиотек, которые имеют следующие особенности [20].

- AForge.Imaging – наибольшая библиотека платформы, которая содержит различные процедуры обработки изображений, предназначенные для решения задач улучшения качества изображения, обработки изображений и компьютерного зрения.
- AForge.Vision – библиотека, которая состоит из различных процедур обнаружения и обработки движения.
- AForge.Math – библиотека, которая содержит различные алгоритмы, связанные с математикой.
- AForge.Video – библиотека, которая состоит из различных классов, обеспечивающих доступ к видео данным и их обработку.
- AForge.Robotics – библиотека, которая содержит классы, предназначенные для манипулирования робототехническими комплексами (Robotics kits).

- AForge.Neuro – библиотека, которая состоит из классов, предназначенных для создания нейронных сетей и их обучения.
- AForge.Genetic – библиотека, которая состоит из классов, направленных на решение различных задач в областях генетических алгоритмов (*Genetic Algorithms*), генетического программирования (*Genetic Programming*).
- AForge.Fuzzy – библиотека, которая содержит классы, предназначенные для выполнения различных нечетких вычислений.
- AForge.MachineLearning – библиотека, которая состоит из некоторых классов для области машинного обучения.

Accord.NET (на языке C#) является расширением платформы AForge.NET с новыми инструментами и библиотеками. Эта платформа предназначена для научных вычислений и состоит из следующих главных библиотек [19].

- Accord.Math – содержит дополнительную библиотеку операций над матрицами, наряду с набором численных методов декомпозиции матриц, численные алгоритмы оптимизации, специальные функции и другие инструменты для научных приложений.
- Accord.Statistics – состоит из библиотек распределений вероятностей, статистических моделей и методов, таких как линейные и логистические регрессии, скрытых моделей Маркова, МГК, дискриминантного анализа и др.
- Accord.MachineLearning – содержит SVM, деревья решений, сети Байеса, алгоритм K-средних, алгоритм RANSAC и д.р.
- Accord.Neuro – содержит алгоритмы обучения нейронных сетей, такие как Levenberg-Marquardt, Parallel Resilient Backpropagation, процедуры инициализации, такие как Nguyen-Windrow и другие методы, связанные с нейронными сетями.
- Accord.Imaging – содержит алгоритмы выделения главных точек,

такие как Harris, SURF и FAST, методы для сшивания изображений (*image stitching*) и сравнения изображений (*image matching*).

- Accord.Audio – содержит методы преобразования, фильтрации и обработки аудио сигналов для машинного обучения и статистических приложений.
- Accord.Vision – содержит методы для распознавания и отслеживания лиц в режиме реального времени, а также методы для обнаружения, отслеживания и трансформации объектов в потоке изображений.

OpenCV (*Open Source Computer Vision Library*) является наиболее полной и распространенной открытой библиотекой по компьютерному зрению и машинному обучению [21]. В библиотеке имеется более 2500 оптимизированных алгоритмов, который включает в себя полный набор классических и современных алгоритмов компьютерного зрения и машинного обучения. Эти алгоритмы могут быть использованы для того чтобы обнаруживать и распознавать лица, идентифицировать объекты, классифицировать действия человека на видеопоследовательностях, отслеживать движущиеся объекты, извлекать 3D-модели объектов, находить похожие изображения из базы изображений, удалять эффекты красных глаз от снимков и т.д.

Библиотека OpenCV широко используется в известных компаниях, таких как Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota и во многих других компаниях, таких как Applied Minds, VideoSurf и Zeitera. Библиотека OpenCV также используется в больших проектах для решения таких задач, как: обнаружение вторжения в системах видеонаблюдения в Израиле, контроль шахтного (*mine*) оборудования в Китае, помощь роботам в ориентировании, обнаружение утопших в бассейне в Европе, проверка состояния взлетно-посадочных полос в Турции.

Указанная библиотека имеет C++, C, Python и Java интерфейсы и

поддерживает операционные системы Windows, Linux, Android и Mac OS. Для этой библиотеки также существует привязка (*cross platform*) Emgu CV, которая позволяет другим языкам программирования, таким как C#, VB, VC++, IronPython и др. вызывать функции библиотеки OpenCV.

Исходя из вышеперечисленных особенностей, язык C# с использованием библиотеки OpenCV через привязку Emgu CV был выбран для дальнейшей реализации программного средства распознавания символов и фрагментов печатных текстов.

3.2 Реализованные классы для распознавания символов и фрагментов текстов

3.2.1 Классы для распознавания символов

В соответствии с описанием разработанного алгоритма распознавания символов, приведенным в разделе 2.1, модуль для распознавания символов должен решать следующие задачи: выделение признаков символов, уменьшение размерности вектора признаков символов и создание классификатора.

а. Классы для выделения признаков символов

Для **выделения признаков символов** созданы следующие классы: *Wavelet*, *WaveletTransform* и *ImageFeature* (рис. 3.1).

Класс Wavelet

Класс `Wavelet` предназначен для сохранения информации о вейвлете. В классе содержатся коэффициенты для преобразования и указывается количество этих коэффициентов. Класс `Wavelet` имеет конструктор, который загружает коэффициенты для преобразования из соответствующего текстового файла. Основные переменные и методы класса `Wavelet` перечислены в табл. 3.1.

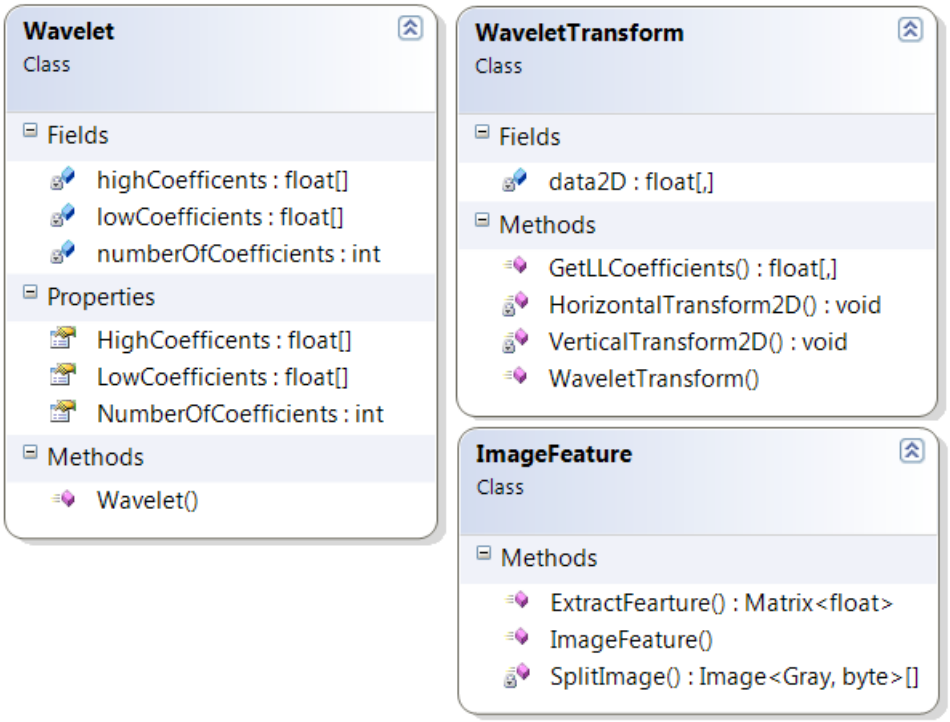


Рисунок 3.1. Классы для выделения признаков символов

Таблица 3.1. Основные переменные и методы класса `Wavelet`

Название переменной/метода	Описание
`public double[] HighCoefficicents`	Высокочастотные коэффициенты для преобразования.
`public double[] LowCoefficients`	Низкочастотные коэффициенты для преобразования.
`public int NumberOfCofficients`	Количество коэффициентов.
`public Wavelet()`	Загрузка высокочастотных и низкочастотных коэффициентов вейвлет-преобразования из текстового файла.

Класс WaveletTransform

Класс `WaveletTransform` предназначен для сохранения информации, полученной на основе использования вейвлет-преобразования,

выполняющегося с помощью класса `Wavelet`. В классе `WaveletTransform` содержатся полученные вейвлет-коэффициенты `data2D` путем применения вейвлет-преобразования к входному изображению. Класс `WaveletTransform` имеет конструктор, который выполняет вейвлет-преобразование на основе использования двух методов `HorizontalTransform2D` и `VerticalTransform2D`, которые горизонтально и вертикально преобразуют изображение для извлечения вейвлет-коэффициентов. В классе `WaveletTransform` имеется метод `GetLLCoefficients`, предназначенный для возвращения полученных низкочастотных вейвлет-коэффициентов, использованных в качестве признаков символов. Основные переменные и методы класса `WaveletTransform` перечислены в табл. 3.2.

Таблица 3.2. Основные переменные и методы класса `WaveletTransform`

Название переменной/метода	**Описание**
`private double[,] data2D`	Полученные коэффициенты на основе применения вейвлет-преобразования к входному изображению.
`public WaveletTransform()`	Применение вейвлет-преобразования к входному изображению.
`private void HorizontalTransform2D()`	Применение вейвлет-преобразования к столбцам входного изображения.
`private void VerticalTransform2D()`	Применение вейвлет-преобразования к строкам входного изображения.
`public float[,] GetLLCoefficients()`	Возвращение низкочастотных вейвлет-коэффициентов.

Класс ImageFeature

Класс `ImageFeature` предназначен для выделения признаков символов. Класс `ImageFeature` имеет конструктор, который создает новые экземпляры этого класса. Для выделения признаков символов в классе созданы следующие методы: `SplitImage` и `ExtractFeature`. Метод `SplitImage` предназначен для разделения входного изображения на 12 локальных частей. Метод `SplitImage` также включает шаг изменения размера изображения символа до 64×64 пикселей. Метод `ExtractFeature` осуществляет выделение признаков символа на основе применения вейвлет-преобразования `WaveletTransform` к изображению символа размером 64×64 пикселей и к каждой из 12 полученных локальных частей изображения символа. Основные методы класса `ImageFeature` перечислены в табл. 3.3.

Таблица 3.3. Основные методы класса `ImageFeature`

Название метода	Описание
`public ImageFeature ()`	Создание нового экземпляра класса.
`private void SplitImage()`	Разложение изображения символа на локальные части.
`public Matrix<float> ExtractFeature()`	Выделение характерных признаков символа.

b. Классы для уменьшения размерности вектора признаков символов

Для **уменьшения размерности вектора признаков символов** созданы следующие классы *EigenSpace* и *EigenSpaceManager* (рис. 3.2).

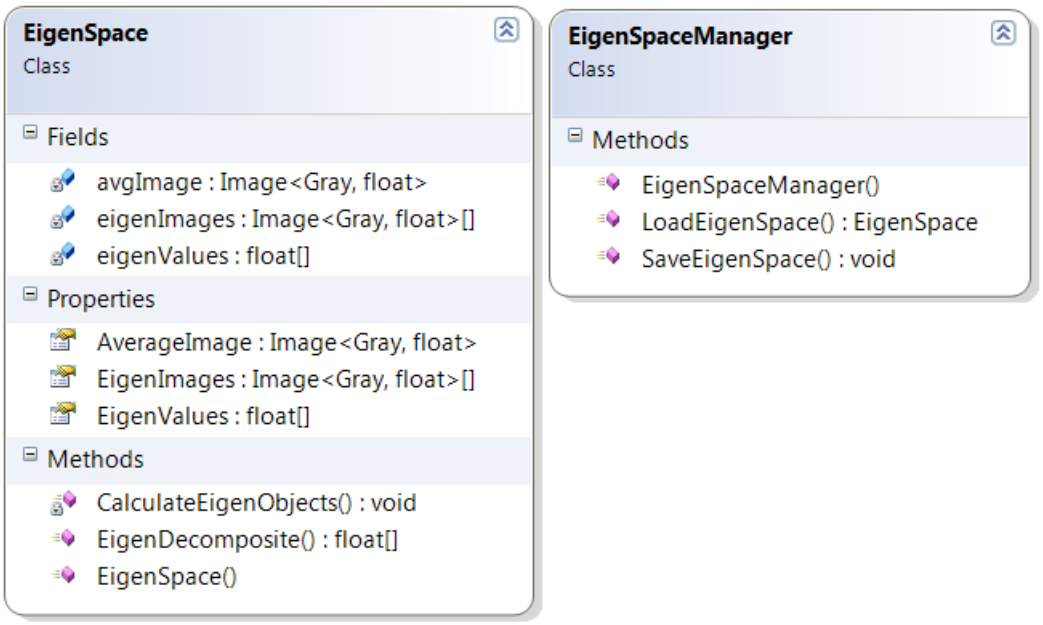


Рисунок 3.2. Классы для уменьшения размерности вектора признаков символов

Класс EigenSpace

Класс `EigenSpace` предназначен для создания пространства собственных символов, применяемого для уменьшения размерности вектора признаков символов. В классе содержатся среднее изображение из набора M изображений, множество собственных символов и их собственных значений. Класс имеет конструктор, который создает собственные символы с использованием метода `CalculateEigeinObjects`, предназначенного для вычисления собственных векторов ковариационной матрицы. В классе `EigenSpace` содержатся метод `EigenDecomposite`, использующийся для разложения входного изображения по собственным символам, в результате которого получается вектор признаков с меньшей размерности. Основные методы класса `EigenSpace` перечислены в табл. 3.4.

Таблица 3.4. Основные переменные и методы класса `EigenSpace`

Название переменной/метода	Описание
`private Image<Gray, Single>[] EigenImages`	Множество собственных изображений символов.

Название переменной/метода	Описание
`private Matrix<float>[] EigenValues`	Множество собственных значений, соответствующих этим собственным изображениям.
`private Image<Gray, Single> AverageImage`	Среднее изображение набора M изображений символов.
`public EigenSpace()`	Создание пространства собственных символов.
`public static void CalculateEigenObjects()`	Вычисление собственных векторов, их собственных значений и среднего изображения.
`public static float[] EigenDecomposite()`	Разложение входного изображения по полученным собственным векторам.

Класс EigenSpaceManager

Класс `EigenSpaceManager` имеет конструктор, который создает новые экземпляры этого класса. В классе `EigenSpaceManager` содержатся следующие методы: `SaveEigenSpace` и `LoadEigenSpace`. Метод `SaveEigenSpace` предназначен для сохранения созданного пространства собственных символов `EigenSpace` в файле для его последующего использования. Метод `LoadEigenSpace` загрузит пространство собственных символов `EigenSpace` из файла. Основные методы класса `EigenSpaceManager` перечислены в табл. 3.5.

Таблица 3.5. Основные методы класса `EigenSpaceManager`

Название метода	Описание
`public void SaveEigenSpace()`	Сохранение созданного пространства собственных символов в файле.

Название метода	Описание
`public EigenSpace LoadEigenSpace()`	Загрузка пространства собственных символов из файла.
`public EigenSpaceManager()`	Создание нового экземпляра класса.

c. Классы для создания классификатора на основе нейронных сетей

Для **построения классификатора** созданы следующие классы *NeuralNetwork*, *Classificator* и *ClassificatorManager* (рис. 3.3).

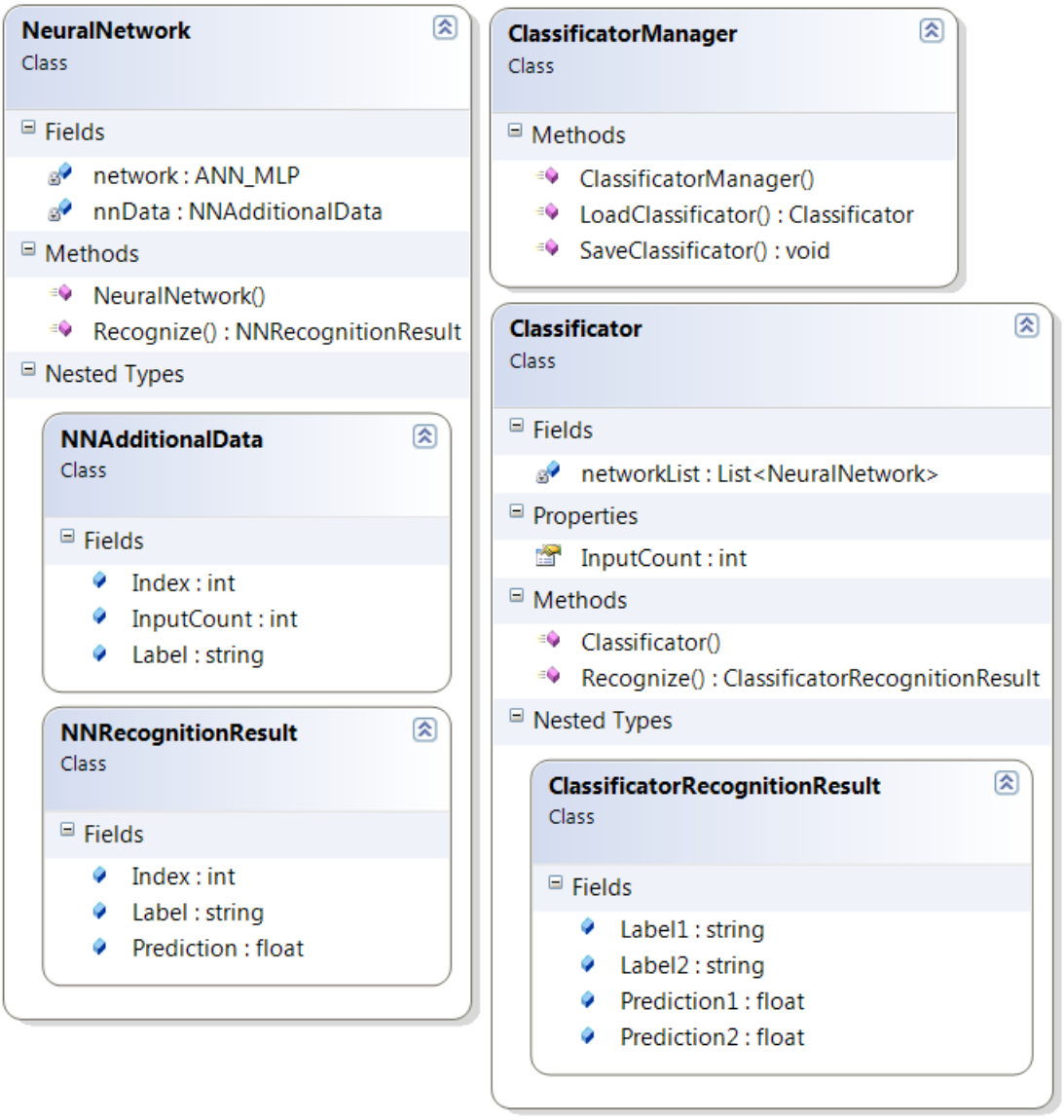


Рисунок 3.3. Классы для создания классификатора

Класс NeuralNetwork

Класс `NeuralNetwork` предназначен для создания специальной нейронной сети, соответствующей только одному символу обучающей выборки. В классе `NeuralNetwork` содержатся многослойная нейронная сеть типа класса `ANN_MLP` и соответствующая этой сети дополнительная информация типа класса `NNAdditionalData`. Класс `NNAdditionalData` включает в себя название символа обучающей выборки, соответствующего нейронной сети, количество нейронов входного слоя и номер сети.

Класс `NeuralNetwork` имеет конструктор, предназначенный для создания многослойной нейронной сети, которая обучается заданными изображениями символов обучающей выборки. В классе `NeuralNetwork` содержится метод `Recognize`, который возвращает результат распознавания символов в виде класса `NNRecognitionResult`. Класс `NNRecognitionResult` включает в себя название символа обучающей выборки, соответствующего нейронной сети, степень близости к символу и номер сети. Основные переменные и методы классов `NeuralNetwork`, `NNAdditionalData` и `NNRecognitionResult` перечислены в табл. 3.6–3.8.

Таблица 3.6. Основные переменные и методы класса `NeuralNetwork`

Название переменной/метода	Описание
`private ANN_MLP network`	Многослойная нейронная сеть типа класса `ANN_MLP` (`ANN_MLP` является классом нейронной сети библиотеки Emgu CV).
`private NNAdditionalData nnAddData`	Дополнительная информация о нейронной сети.
`public NeuralNetwork()`	Создание многослойной нейронной сети, обученной заданными изображениями символов.

Название переменной/метода	Описание
`public NNRecognitionResult Recognize()`	Возвращение результата распознавания символов нейронной сетью в виде класса `NNRecognitionResult.`

Таблица 3.7. Основные переменные класса `NNAdditionalData`

Название переменной	Описание
`public string Label`	Название символа обучающей выборки, соответствующего нейронной сети.
`public int InputCount`	Количество нейронов входного слоя нейронной сети.
`public int Index`	Номер нейронной сети.

Таблица 3.8. Основные переменные класса `NNRecognitionResult`

Название переменной	Описание
`public string Label`	Название символа обучающей выборки, соответствующего нейронной сети.
`public float Prediction`	Степень близости к символу обучающей выборки, находящаяся в пределах от 0 до 1.
`public int Index`	Номер нейронной сети.

Класс Classificator

Класс `Classificator` предназначен для создания классификатора на основе специальных нейронных сетей `NeuralNetwork`. В классе `Classificator` содержится множество специальных многослойных нейронных сетей и указывается количество нейронов их входного слоя. Класс `Classificator`

имеет конструктор, который создает множество специальных многослойных нейронных сетей, количество которых соответствует количеству классов обучающих символов. В классе `Classificator` содержится метод `Recognize`, который возвращает результат распознавания символов в виде класса `ClassificatorRecognitionResult`. Класс `ClassificatorRecognitionResult` включает в себя название и степень близости символа первого предположения, а также название и степень близости символа второго предположения. Основные переменные и методы классов `Classificator` и `ClassificatorRecognitionResult` перечислены в табл. 3.9–3.10.

Таблица 3.9. Основные переменные и методы класса `Classificator`

Название переменной/метода	**Описание**
`private List<NeuralNetwork> networkList`	Множество специальных многослойных нейронных сетей.
`public int InputCount`	Количество нейронов входного слоя нейронных сетей.
`public Classificator ()`	Создание классификатора для распознавания символов, состоящего из специальных многослойных нейронных сетей.
`public ClassificatorRecognitionResult Recognize()`	Возвращение результата распознавания символов нейронными сетями в виде класса `ClassificatorRecognitionResult.`

Таблица 3.10. Основные переменные класса `ClassificatorRecognitionResult`

Название переменной	**Описание**
`public string Label1`	Название символа обучающей выборки первого предположения.

Название переменной	Описание
`public float Prediction1`	Степень близости к символу обучающей выборки первого предположения.
`public string Label2`	Название символа обучающей выборки второго предположения.
`public float Prediction2`	Степень близости к символу обучающей выборки второго предположения.

Класс ClassificatorManager

Класс `ClassificatorManager` имеет конструктор, который создает новые экземпляры этого класса. В классе `ClassificatorManager` содержатся следующие методы: `SaveClassificator` и `LoadClassificator`. Метод `SaveClassificator` предназначен для сохранения созданного классификатора `Classificator`, основанного на специальных многослойных нейронных сетях, в файле для его последующего использования. Метод `LoadClassificator` осуществляет загрузку классификатора `Classificator` для распознавания символов из файла. Основные методы класса `ClassificatorManager` перечислены в табл. 3.11.

Таблица 3.11. Основные методы класса `ClassificatorManager`

Название метода	Описание
`public void SaveClassificator ()`	Сохранение созданного классификатора для распознавания символов в файле.
`public Classificator LoadClassificator ()`	Загрузка классификатора для распознавания символов из файла.
`public ClassificatorManager ()`	Создание нового экземпляра класса.

3.2.2 Классы для распознавания фрагментов печатных текстов

В соответствии с описанием алгоритма распознавания фрагмента печатного текста, приведенным в разделе 2.2, модуль для распознавания фрагментов печатных символов должен решать следующие задачи: поворот изображения фрагмента текста, выделение строк из фрагмента текста, выделение слов из строк, выделение символов из слов и распознавание выделенных символов. Для **распознавания выделенных символов** используются классы для распознавания символов, представленные в разделе 3.2.1.

a. Классы для поворота изображения фрагмента текста

Для **поворота изображения фрагмента текста** созданы следующие классы: *HorizontalHistogram* и *ImageProc* (рис. 3.4).

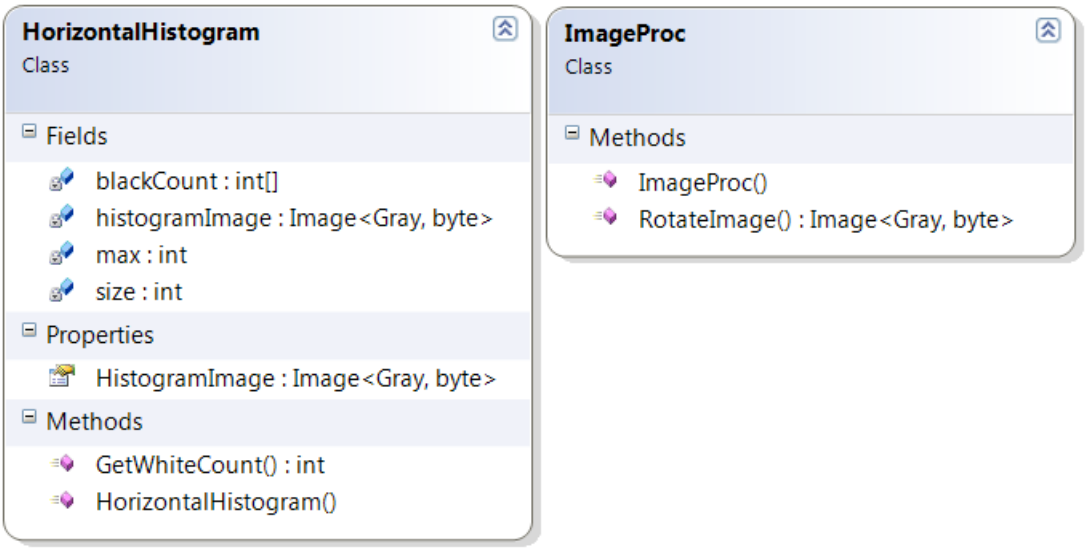


Рисунок 3.4. Классы для поворота изображения фрагмента текста

Класс HorizontalHistogram

Класс `HorizontalHistogram` предназначен для построения проекции бинарного изображения фрагмента печатного текста на ось Y. В классе содержатся количество строк (высота) изображения; количество черных пикселей каждой строки и их максимальное значение; изображение, которое используется для представления данных проекции. Класс `HorizontalHistogram` имеет конструктор, с помощью которого проводится

построение проекции изображения фрагмента текста на ось Y, использующейся для его поворота. В классе `HorizontalHistogram` также имеется метод `GetWhiteCount`, который вычисляет количество «белых точек». Основные переменные и методы класса `HorizontalHistogram` перечислены в табл. 3.12.

Таблица 3.12. Основные переменные и методы класса `HorizontalHistogram`

Название переменной/метода	Описание
`private int[] blackCount`	Матрица, значение каждого элемента которой равно количеству черных пикселей, расположенных на одной строке изображения фрагмента текста.
`private int size`	Количество строк (высота) изображения текста.
`private int max`	Максимальное количество черных пикселей.
`public Image<Gray, byte> HistogramImage`	Изображение, представляющее данные проекции.
`public HorizontalHistogram()`	Создание проекции изображения фрагмента текста на ось Y.
`public int GetWhiteCount()`	Вычисление количества «белых строк» проекции изображения фрагмента текста.

Класс ImageProc

Класс `ImageProc` предназначен для обработки изображения фрагмента текста. Класс `ImageProc` имеет конструктор, который создает новые экземпляры этого класса. В классе `ImageProc` содержится метод `RotateImage`,

предназначенный для поворота изображения фрагмента текста на основе использовании проекции изображения на ось *Y* `HorizontalHistogram`. Поворот изображения фрагмента текста заключается в нахождении повернутого изображения текста, на котором расположено наибольшее количество «белых строк». Основные методы класса `ImageProc` перечислены в табл. 3.13.

Таблица 3.13. Основные методы класса `ImageProc`

Название метода	Описание
`public Image<Gray, byte> RotateImage ()`	Поворот изображения фрагмента печатного текста.
`public ImageProc()`	Создание нового экземпляра класса.

b. Классы для выделения строк фрагмента текста

Для **выделения строк фрагмента текста** созданы следующие классы *TextImageLayout* и *Line* (рис. 3.5).

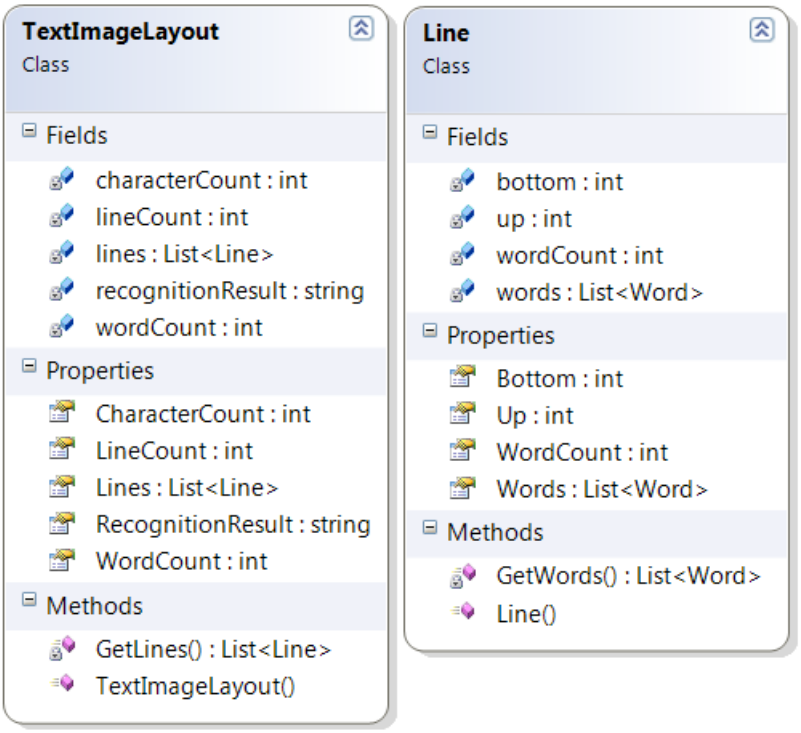


Рисунок 3.5. Классы для выделения строк фрагмента текста

Класс TextImageLayout

Класс `TextImageLayout` предназначен для сохранения информации об изображении фрагмента печатного текста. Класс содержит список выделенных строк; количество выделенных строк, слов и символов текста; результат распознавания текста. Класс `TextImageLayout` имеет конструктор, с помощью которого класс извлекает информацию о бинарном изображении текста. В классе содержится метод `GetLines`, который выполняет выделение строк `Line` из фрагмента текста на основе применения проекции изображения текста на ось Y, построенной с использованием класса `HorizontalHistogram`. Основные переменные и методы класса `TextImageLayout` перечислены в табл. 3.14.

Таблица 3.14. Основные переменные и методы класса `TextImageLayout`

Название переменной/метода	**Описание**
`public List<Line> Lines`	Список выделенных строк из изображения фрагмента текста.
`public int LineCount`	Количество выделенных строк из изображения фрагмента текста.
`public int WordCount`	Количество выделенных слов из изображения фрагмента текста.
`public int CharacterCount`	Количество выделенных символов из изображения фрагмента текста.
`public string RecognitionResult`	Результат распознавания фрагмента текста.
`public TextImageLayout()`	Создание нового экземпляра класса.
`private List<Line> GetLines()`	Выделение строк из изображения фрагмента текста.

Класс Line

Класс `Line` предназначен для сохранения информации о выделенной строке текста. В классе содержатся координаты верхней и нижней границ выделенной строки; список слов в строке и их количество. Класс `Line` имеет конструктор, который извлекает информацию о выделенной строке на основе использования метода `GetWords`, который выделяет слова в строке на основе использования проекции интенсивности строки на ось X. Основные переменные и методы класса `Line` перечислены в табл. 3.15.

Таблица 3.15. Основные переменные и методы класса `Line`

Название переменной/метода	Описание
`public int Up`	Координата верхней границы строки.
`public int Bottom`	Координата нижней границы строки.
`public List<Word> Words`	Список слов строки.
`public int WordCount`	Количество слов строки.
`public Line()`	Создание нового экземпляра класса.
`private List<Word> GetWords()`	Выделение слов строки.

c. Классы для выделения слов из строк

Для **выделения слов** из строк созданы следующие классы *VerticalHistogram* и *Word* (рис. 3.6).

Класс VerticalHistogram

Класс `VerticalHistogram` предназначен для построения проекции интенсивности изображения текста на ось X. В классе содержатся количество столбцов (ширина) изображения; количество черных пикселей каждого столбца и их максимальное значение; изображение, которое используется для

представления данных проекции. Класс `VerticalHistogram` имеет конструктор, с помощью которого строится проекция интенсивности изображения на ось X, использующаяся для выделения слов в строке текста. Основные переменные и методы класса `VerticalHistogram` перечислены в табл. 3.16.

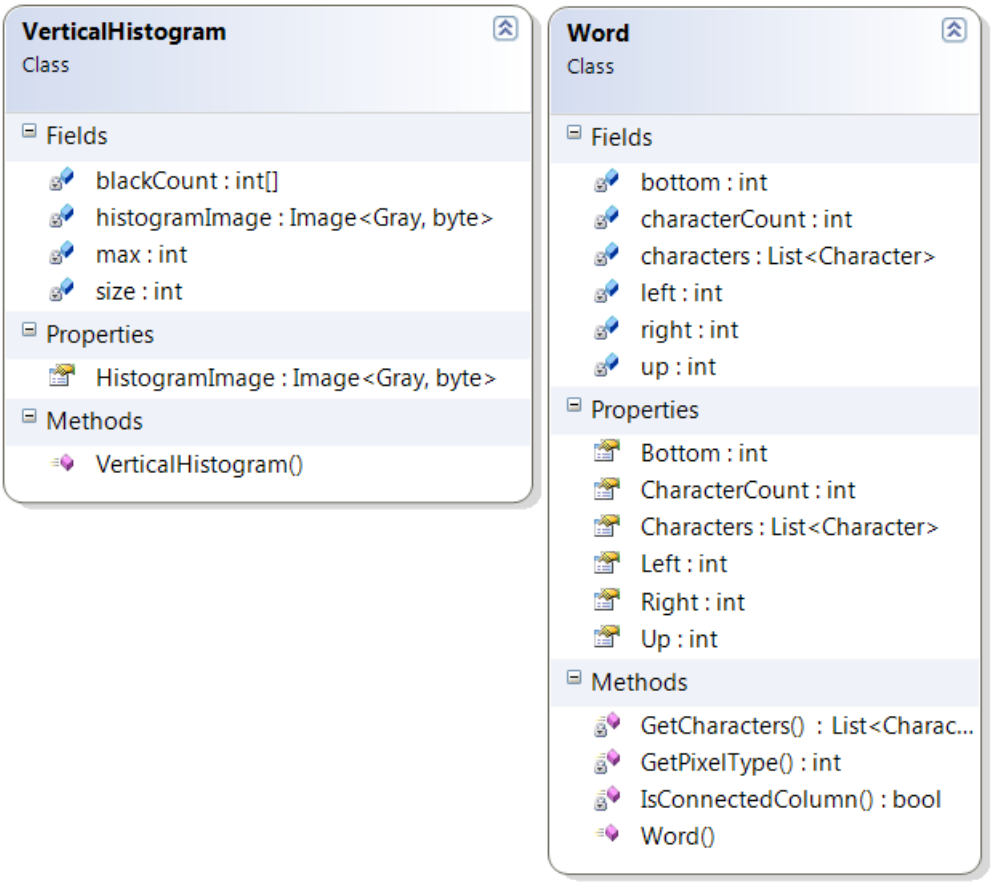


Рисунок 3.6. Классы для выделения слов из строк

Таблица 3.16. Основные переменные и методы класса `VerticalHistogram`

Название переменной/метода	Описание
`private int[] blackCount`	Матрица, значение каждого элемента которой равно количеству черных пикселей, расположенных на одном столбце изображения фрагмента текста.

Название переменной/метода	Описание
`private int size`	Количество столбцов изображения фрагмента текста.
`private int max`	Максимальное количество черных пикселей.
`public Image<Gray, byte> HistogramImage`	Изображение, представляющее данные проекции.
`public VerticalHistogram()`	Создание нового экземпляра класса.

Класс Word

Класс `Word` предназначен для сохранения информации о выделенном слове в строке текста. Слова из строк выделяются с использованием метода `GetWords` созданного класса `Line`. В классе содержатся координаты верхней, нижней, левой и правой границ выделенного слова; список символов в слове и их количество. Класс `Word` имеет конструктор, который извлекает информацию о слове на основе использования метода `GetCharacters`, который выделяет символы в слове с использованием «соединяющихся столбцов», определяющихся методами `IsConnectedColumn` и `GetPixelType`. Основные переменные и методы класса `Word` перечислены в табл. 3.17.

Таблица 3.17. Основные переменные и методы класса `Word`

Название переменной/метода	Описание
`public int Up`	Координата верхней границы выделенного слова.
`public int Bottom`	Координата нижней границы выделенного слова.
`public int Left`	Координата левой границы выделенного слова.

Название переменной/метода	Описание
`public int Right`	Координата правой границы выделенного слова.
`public List<Character> Characters`	Список символов в выделенном слове.
`public int CharacterCount`	Количество символов в выделенном слове.
`public Word()`	Создание нового экземпляра класса.
`private bool IsConnectedColumn()`	Определение, является ли столбец изображения выделенного слова «соединяющимся столбцом».
`private int GetPixelType()`	Определение типа пикселя, использованного для определения «соединяющегося столбца».
`private int GetCharacters()`	Выделение символов слова.

d. Классы для выделения символов из слов

Для **выделения символов** из слов созданы следующие классы *Character* и *Rect* (рис. 3.7).

Класс Character

Класс `Character` предназначен для сохранения информации о выделенном символе в словах текста. Символы из слов выделяются с использованием метода `GetCharacters` созданного класса `Word`. В классе содержатся координаты верхней, нижней, левой и правой границ возможной области выделенного символа; список четырех областей символа и тип символа, использованного для отдельного распознавания похожих по написанию символов (см. раздел 2.2.3). Класс `Character` имеет конструктор, который извлекает информацию о выделенном символе на основе

использования метода `GetRects`, который выделяет четыре области символа и возвращается тип символа. Основные переменные и методы класса `Character` перечислены в табл. 3.18.

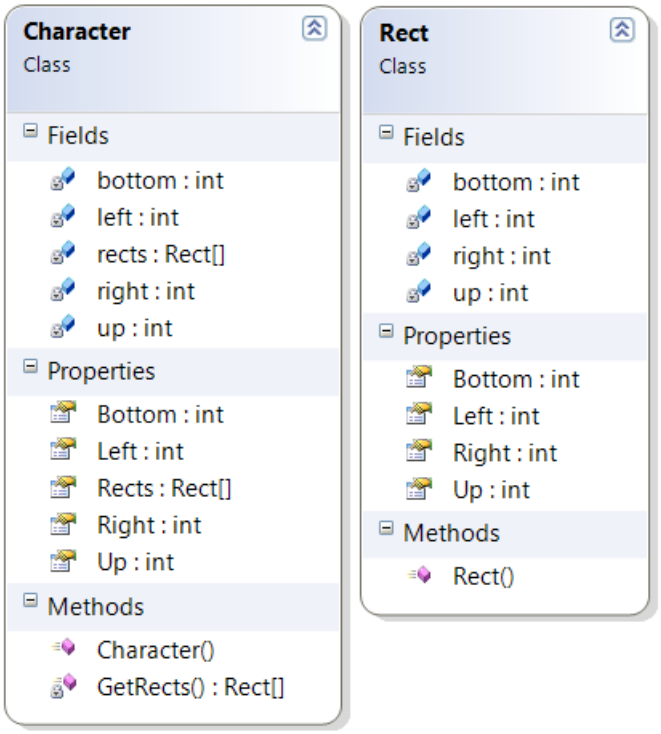


Рисунок 3.7. Классы для выделения символов из слов

Таблица 3.18. Основные переменные и методы класса `Character`

Название переменной/метода	Описание
`public int Up`	Координата верхней границы возможной области выделенного символа.
`public int Bottom`	Координата нижней границы возможной области выделенного символа.
`public int Left`	Координата левой границы возможной области выделенного символа.

Название переменной/метода	Описание
`public int Right`	Координата правой границы возможной области выделенного символа.
`public Rect[] Rects`	Список четырех областей выделенного символа (раздел 2.2.3).
`public Character()`	Создание нового экземпляра класса.
`private Rect[] GetRects()`	Выделение четырех областей символа (раздел 2.2.3).

Класс Rect

Класс `Rect` предназначен для сохранения информации об области выделенного символа `Character`. В классе содержатся координаты верхней, нижней, левой и правой границ области выделенного символа; ширина и высота области символа. Класс `Rect` имеет конструктор, который получает информацию об области символа. Основные переменные и методы класса `Rect` перечислены в табл. 3.19.

Таблица 3.19. Основные переменные и методы класса `Rect`

Название переменной/метода	Описание
`public int Up`	Координата верхней границы области выделенного символа.
`public int Bottom`	Координата нижней границы области выделенного символа.
`public int Left`	Координата левой границы области выделенного символа.
`public int Right`	Координата правой границы области выделенного символа.

Название переменной/метода	Описание
`public Rect()`	Создание нового экземпляра класса.

3.3 Разработанные программные средства

При реализации разработанных алгоритмов распознавания символов и фрагментов печатных текстов были созданы программы, использующие классы, описанные в разделе 3.2. Данные классы предназначены как исследователям, так и обычным пользователям. Таким образом, были реализованы две программы, предназначенные для указанных категорий пользователей. Первая программа предназначена для исследования различных методов обработки изображений. Эта программа позволяет исследователям наблюдать результаты шагов разработанных алгоритмов. Вторая программа предназначена для обычных пользователей, которые заинтересованы в первую очередь в удобстве использования, а не в гибкости.

3.3.1 Приложение для исследователей

Для исследования свойств разработанных алгоритмов распознавания символов и фрагментов печатных текстов было разработано приложение, позволяющее исследователям наблюдать и анализировать результаты обработки следующих задач.

- Создание пространства собственных символов.
- Уменьшение размерности вектора признаков символов.
- Создание классификатора на основе нейронных сетей.
- Распознавание рукописных цифр нейронными сетями.
- Распознавание печатных символов нейронными сетями.
- Поворот наклонного изображения фрагмента текста.
- Выделение строк из фрагмента текста.
- Выделение слов в строке.

- Выделение символов в слове.
- Распознавание печатных текстов нейронными сетями.

Для разработки приложения и классов, использовался ориентированный язык программирования Visual C#, библиотека OpenCV через привязку Emgu CV и среда Microsoft Visual Studio 2010. Интерфейс пользователя приложения для исследователей представлен на рис. 3.8. Приложение состоит из вкладок, выполняющих вышеперечисленные задачи.

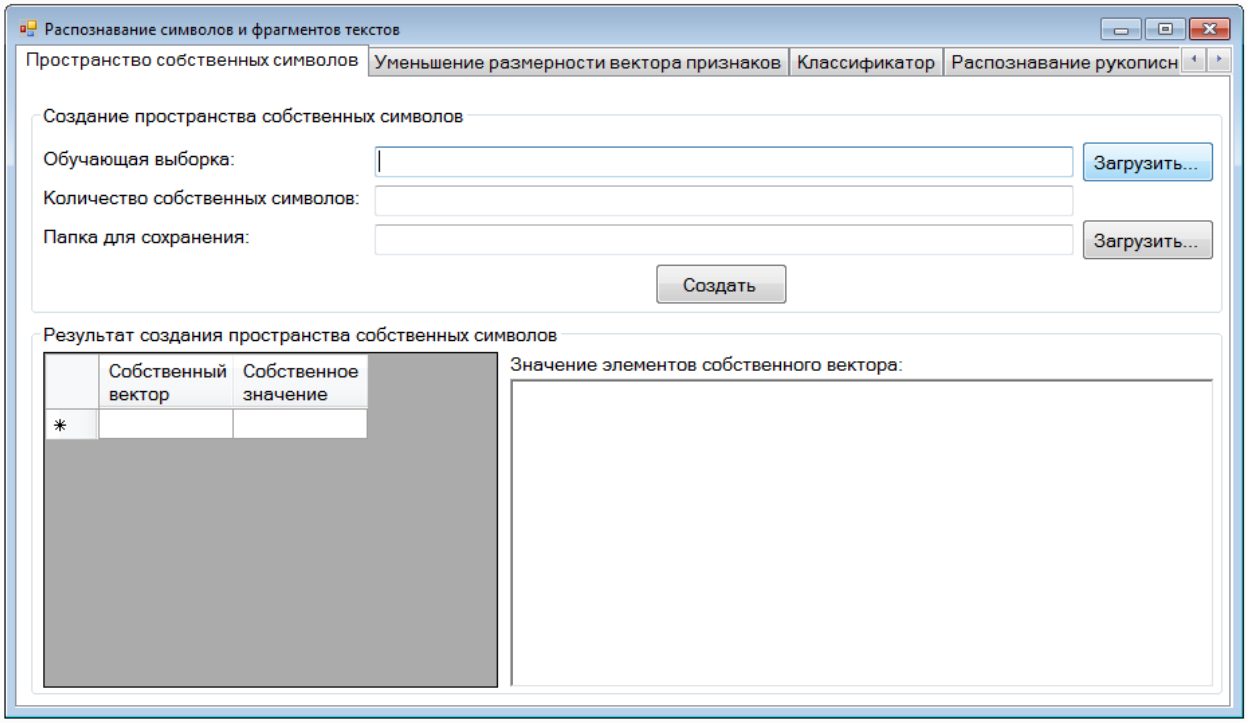


Рисунок 3.8. Интерфейс пользователя приложения для исследователя

a. Создание пространства собственных символов

Создание пространства собственных символов осуществляется с помощью вкладки «*Пространство собственных символов*» (рис. 3.9). Вначале пользователю нужно вводить необходимые данные для создания собственных символов. На поле «*Обучающая выборка*» представлена папка, в которой содержатся изображения символов, используемые для создания собственных символов. Поле «*Количество собственных символов*» обозначает число собственных векторов ковариационной матрицы, которое соответствует числу использованных признаков при уменьшении

размерности вектора признаков. На поле «*Папка для сохранения*» представлено место для сохранения пространства собственных символов в файле. Кнопки «*Загрузить…*» открывают стандартный диалог для выбора папки изображений символов или места для сохранения пространства собственных символов. После заполнения всех полей пользователю необходимо нажать кнопку «*Создать*» для создания пространства собственных символов.

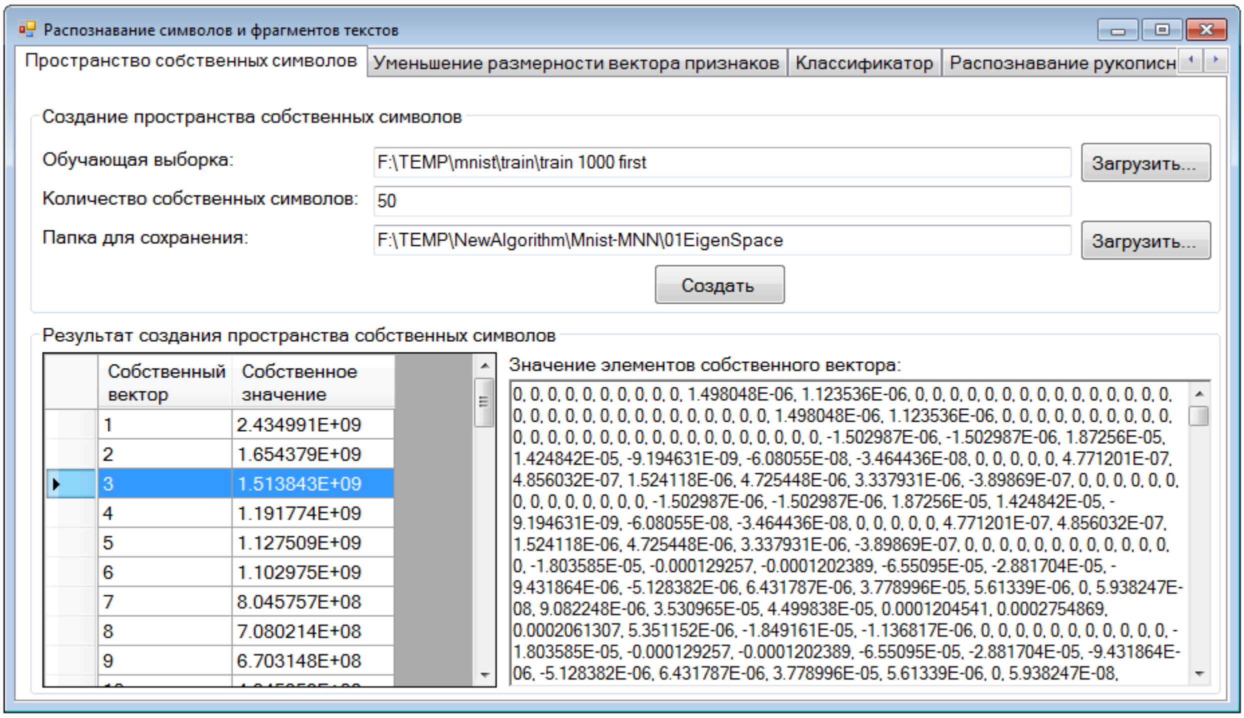


Рисунок 3.9.   Вкладка «Пространство собственных символов»

После того как создано пространство собственных символов, данные полученных собственных векторов представляются в виде таблицы, имеющей два столбца: «*Собственный вектор*» и «*Собственное значение*» (рис. 3.9). На этих столбцах показываются номер собственного вектора и соответствующее ему собственное значение. Собственные векторы отсортированы по убыванию соответствующих им собственных значений. Кроме того, поле «*Значение элементов собственного вектора*» позволяет исследователю наблюдать значения элементов выбранного собственного вектора.

Папка с изображениями символов состоит из папок, в каждой из которых содержатся изображения только одного класса символа. На рис. 3.10 представлен пример папки с изображениями символов, используемые для создания пространства собственных символов.

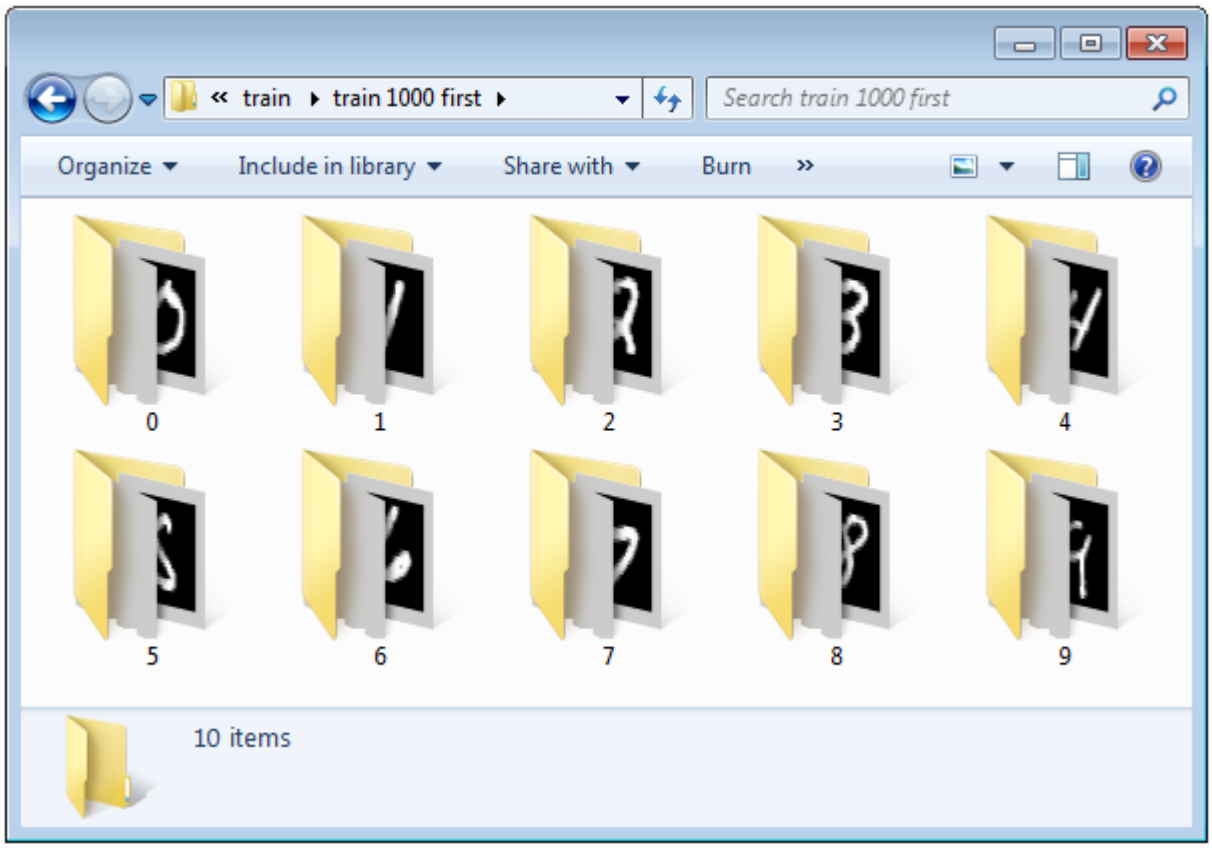


Рисунок 3.10. Пример папки с изображениями символов

b. Уменьшение размерности вектора признаков символов

Уменьшение размерности вектора признаков символов выполняется с помощью вкладки «*Уменьшение размерности вектора признаков*» (рис. 3.11). Вначале пользователю необходимо указать используемые изображения символов и пространство собственных символов, предназначенное для уменьшения размерности вектора признаков символов. Данные изображений символов представляются в виде таблицы, состоящей из трех столбцов: «*Номер*», «*Символ*» и «*Изображение*». На этих столбцах показываются номер изображения, название символа и ссылка на изображение. На поле «*Пространство собственных символов*» представлена ссылка на файл

пространства собственных символов. Кнопки «*Загрузить…*» открывают стандартный диалог для выбора изображений символов или файла пространства собственных символов.

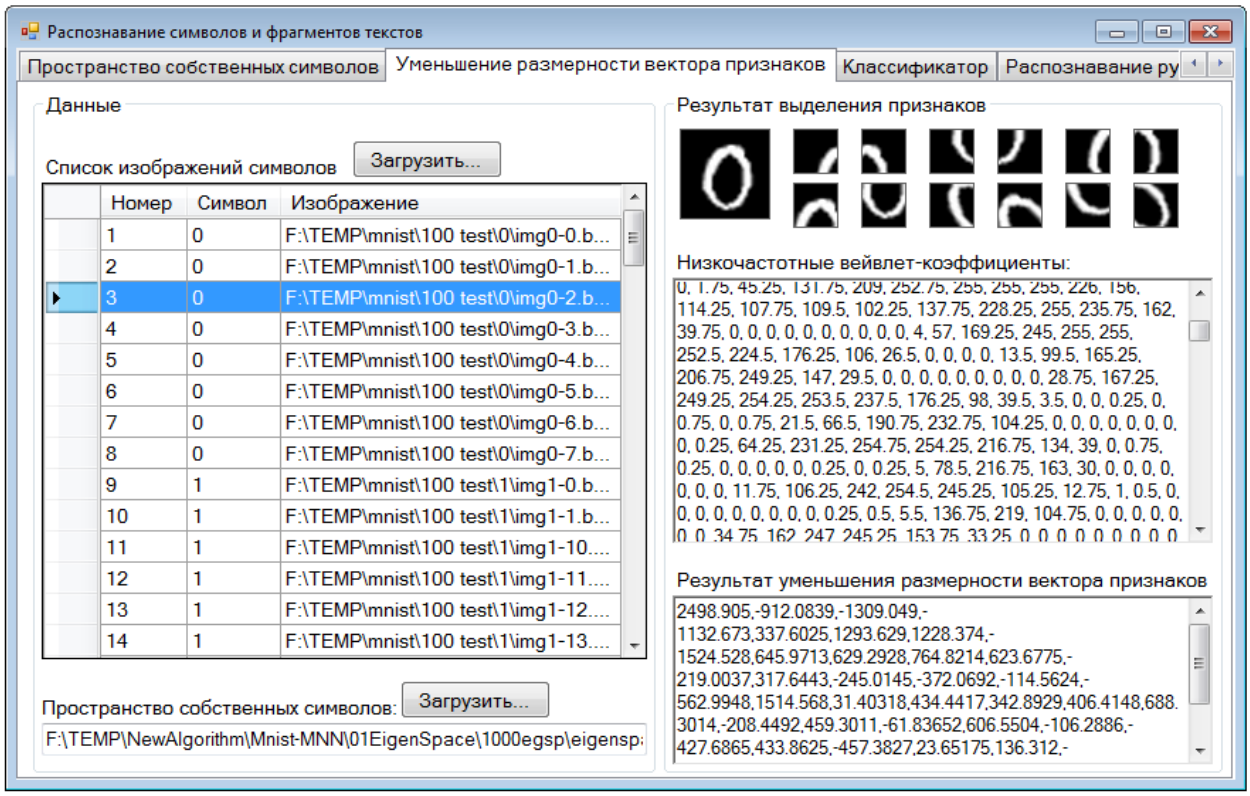


Рисунок 3.11. Вкладка «Уменьшение размерности вектора признаков»

После того как указаны необходимые данные, при выборе изображения символа программа автоматически выделяет вектор признаков выбранного символа и уменьшает его размерность. В результате этого на экране показываются изображение символов размером 64×64 пикселя и его 12 локальных частей. На поле «*Низкочастотные вейвлет-коэффициенты*» представлены значения полученных низкочастотных коэффициентов с использованием вейвлет-преобразования. На поле «*Результат уменьшения размерности вектора признаков*» приведены значения элементов вектора с меньшей размерностью.

c. Создание классификатора на основе нейронных сетей

Для создания классификатора для распознавания символов на основе нейронных сетей используется вкладка «*Классификатор*» (рис. 3.12).

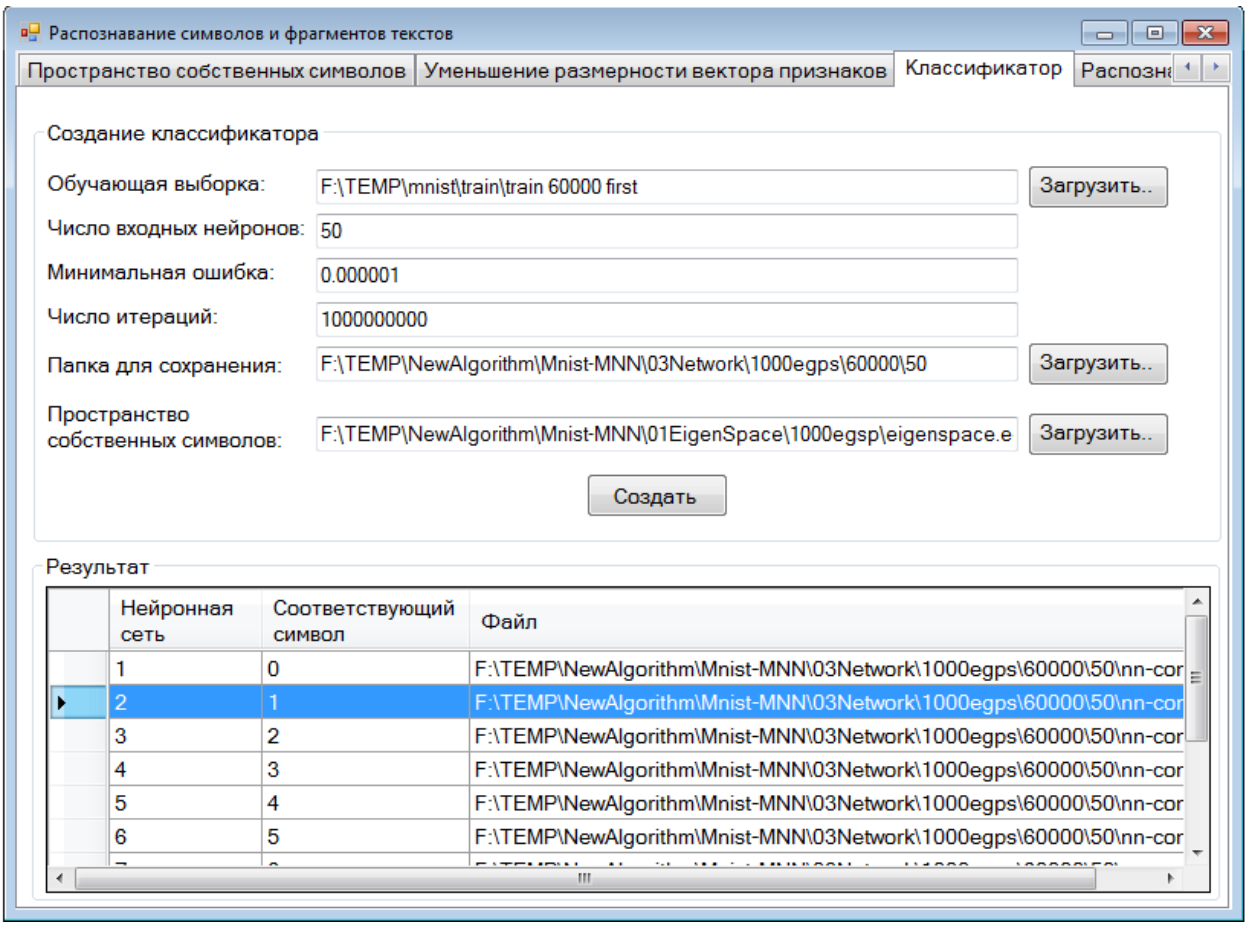


Рисунок 3.12. Вкладка «Классификатор»

Вначале пользователю нужно заполнить необходимые данные для решения этой задачи. На поле «*Обучающая выборка*» указана папка с изображениями символов, используемыми для обучения нейронных сетей. Поле «*Число входных нейронов*» показывает количество нейронов входного слоя нейронных сетей. Минимальная ошибка и максимальное число итераций, использованные в качестве условия завершения обучения нейронных сетей, указаны на полях «*Минимальная ошибка*» и «*Число итераций*». На поле «*Пространство собственных символов*» указана ссылка на файл пространства собственных символов, а на поле «*Пака для сохранения*» – папка, в которой сохранятся обученные нейронные сети. Кнопки «*Загрузить…*» открывают стандартный диалог для выбора папки с обучающими изображениями или файла пространства собственных символов или места для сохранения обученных нейронных сетей. После заполнения необходимых данных пользователь нажимает кнопку «*Создать*» для запуска

процесса создания классификатора.

В результате данные обученных сетей представлены на экране в виде таблицы, состоящей из трех столбцов: «*Нейронная сеть*», «*Соответствующий символ*» и «*Файл*». На этих столбцах приведены номер сети, соответствующий ей символ и ссылка на ее файл.

d. Распознавание рукописных цифр

Для наблюдения результатов распознавания рукописных цифр используется вкладка «*Распознавание рукописных цифр*» (рис. 3.13).

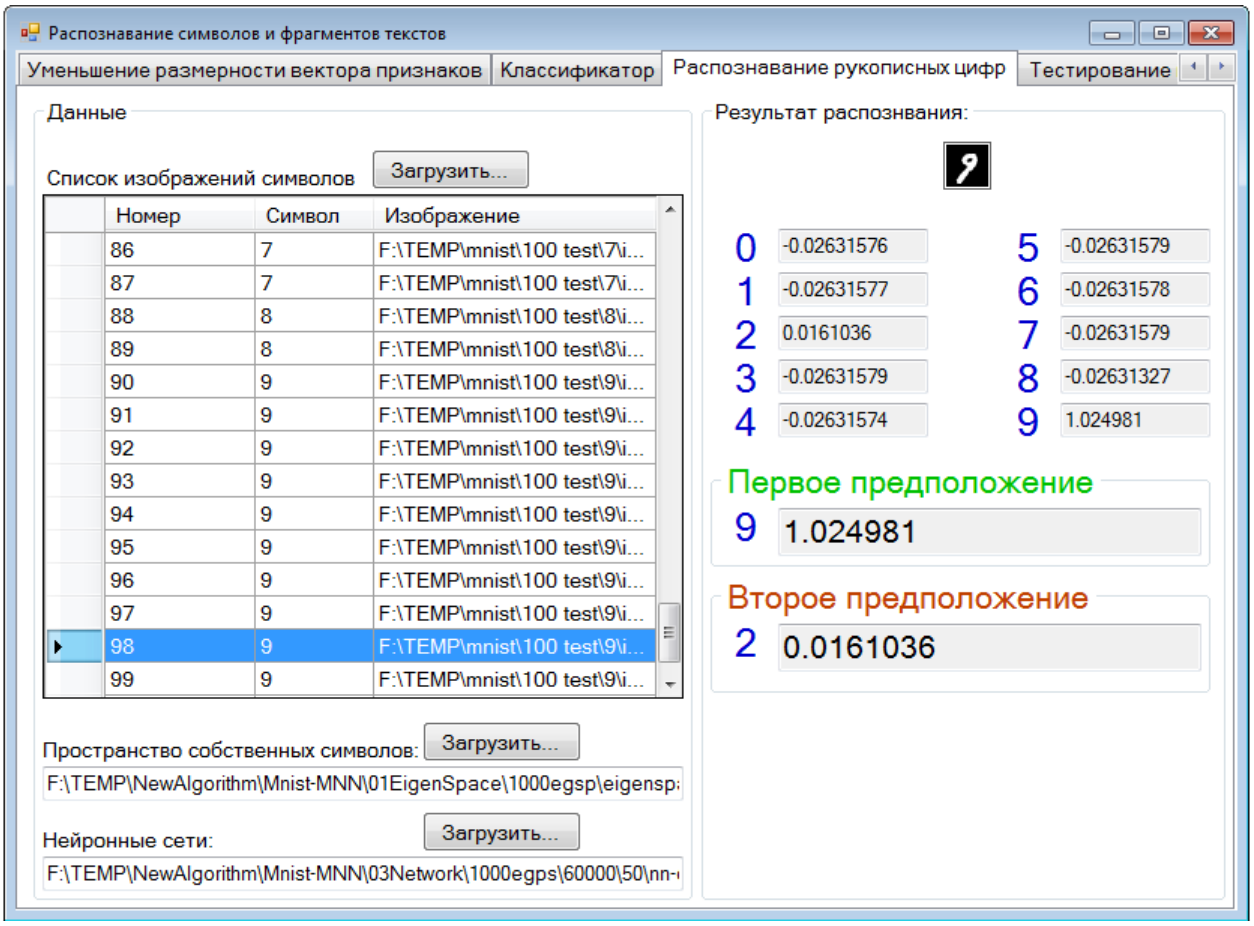


Рисунок 3.13. Вкладка «Распознавание рукописных цифр»

Вначале пользователю необходимо указать изображения распознаваемых цифр, пространство собственных символов и обученные нейронные сети. Данные изображений цифр представляются в виде таблицы, состоящей из трех столбцов: «*Номер*», «*Символ*» и «*Изображение*». На этих столбцах представлены номер изображения, название цифры и ссылка на

изображение. На поле «*Пространство собственных символов*» указана ссылка на файл пространства собственных символов, а на поле «*Нейронные сети*» – папка, в которой сохранены обученные нейронные сети. Кнопки «*Загрузить…*» открывают стандартный диалог для выбора изображений распознаваемых цифр или файла пространства собственных символов или папки с файлами обученных нейронных сетей.

После того как указаны необходимые данные, при выборе изображения цифры программа автоматически распознает его указанными нейронными сетями. В результате этого на экране показывается степень близости распознаваемой цифры к каждой цифре обучающей выборки. В дополнение приводятся результаты распознавания как без учета второго предположения, так и с его учетом.

e. Распознавание печатных символов

Наблюдение результатов распознавания печатных символов осуществляется с помощью вкладки «*Распознавание печатных символов*» (рис. 3.14).

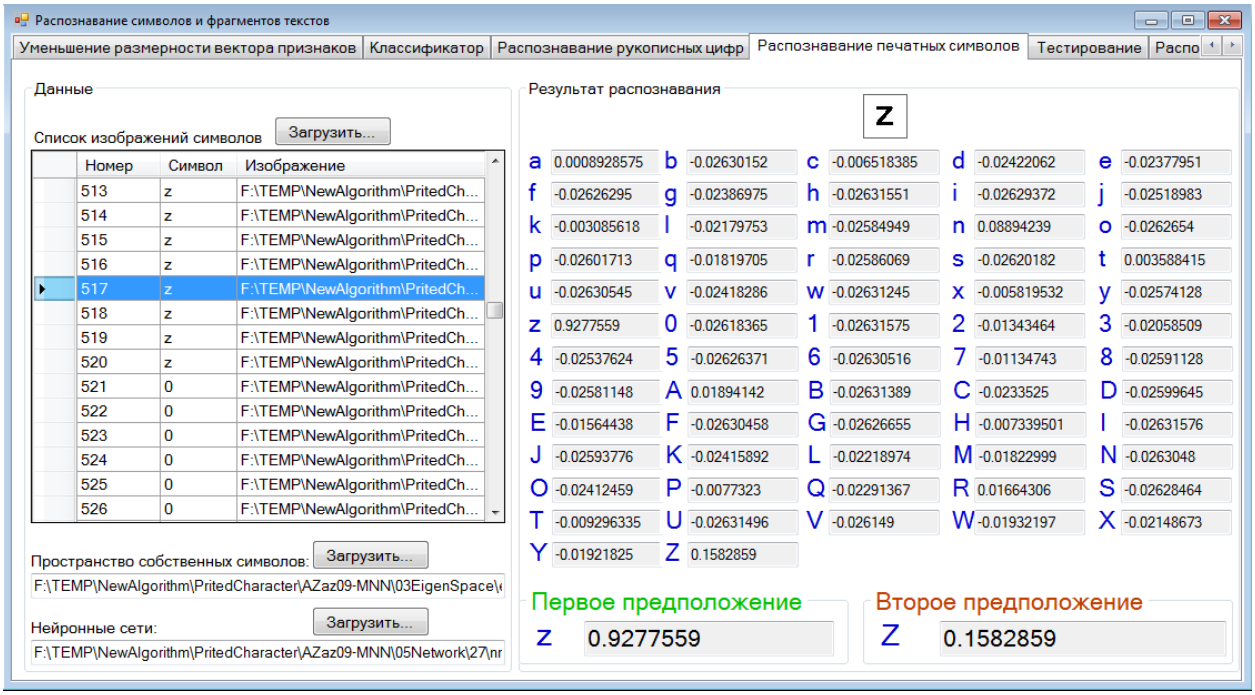


Рисунок 3.14. Вкладка «Распознавание печатных символов»

Вначале пользователю необходимо, как и во вкладке распознавания рукописных цифр (рис. 3.13), указать изображения распознаваемых символов, пространство собственных символов и обученные нейронные сети.

После того как указаны необходимые данные, при выборе изображения символа программа автоматически распознает его с использованием указанных нейронных сетей. В результате этого, на экране показывается степень близости распознаваемого символа к каждому обучающему символу. Также приведены результаты распознавания как без учета второго предположения, так и с его учетом.

f. Автоматическое тестирование процесса распознавания рукописных цифр и печатных текстов

Для автоматического тестирования процесса распознавания рукописных цифр и печатных текстов используется вкладка «*Тестирование*» (рис. 3.15).

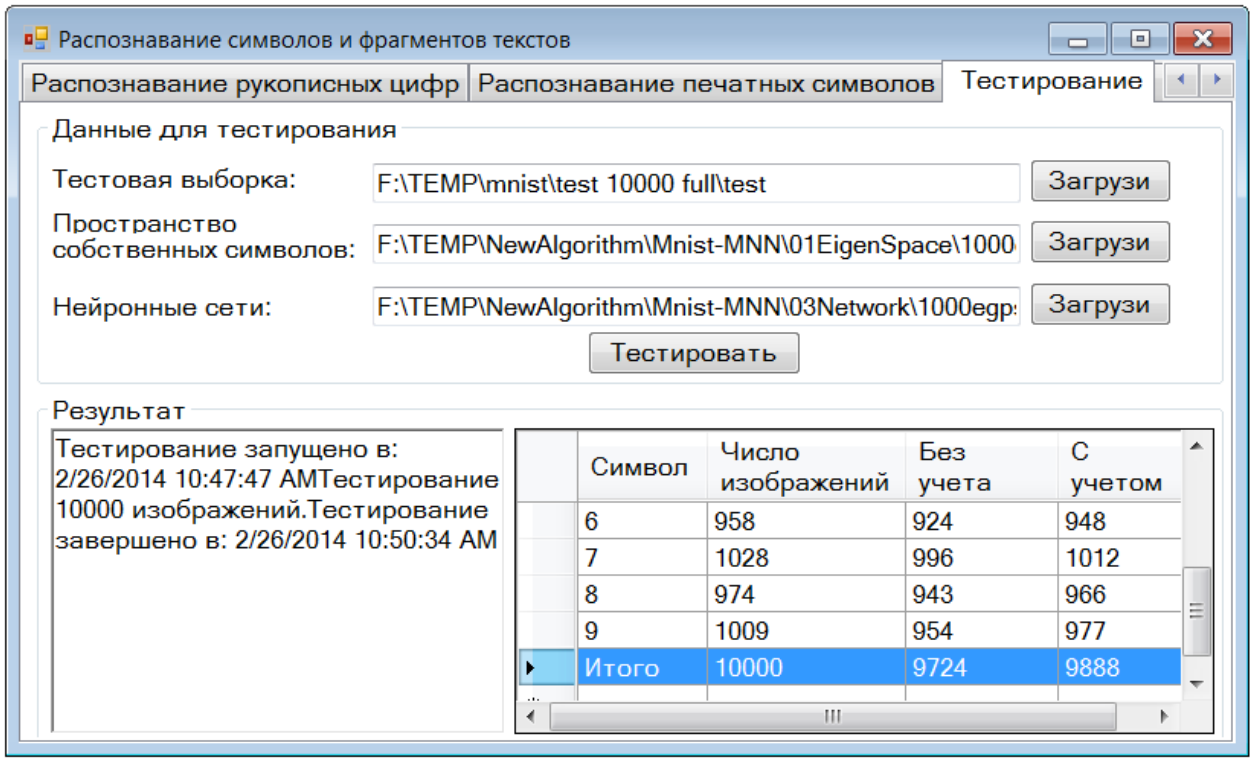


Рисунок 3.15. Вкладка «Тестирование»

Вначале пользователю необходимо указать папку с изображениями символов тестовой выборки, используемые пространство собственных символов и применяемые нейронные сети. Поле «*Тестовая выборка*» представляет ссылку на папку с изображениями символов тестовой выборки. На поле «*Пространство собственных символов*» указана ссылка на файл пространства собственных символов, а на поле «*Нейронные сети*» – папка, в которой сохранены обученные нейронные сети. После заполнения необходимых данных, пользователь нажимает кнопку «*Тестировать*» для запуска процесса автоматического тестирования. Кнопки «*Загрузить…*» открывают стандартный диалог для выбора папки с изображениями символов тестовой выборки или файла пространства собственных символов или папки с файлами обученных нейронных сетей.

После того как завершено тестирование, на экране показывают результаты тестирования, представленные в виде таблицы. Таблица результатов тестирования имеет три столбца: «*Символ*», «*Число изображений*», «*Без учета*» и «*С учетом*». На этих столбцах представляются название символа и соответствующие ему количество изображений, число правильных распознаваний как без учета второго предположения, так и с его учетом.

g. Распознавание фрагментов печатных текстов

Для наблюдения результатов распознавания фрагментов печатных символов используется вкладка «*Распознавание фрагментов текстов*» (рис. 3.16). Вкладка позволяет пользователям наблюдать результаты не только задачи распознавания фрагментов печатных символов, но и результаты других задач, таких как поворот изображения фрагментов текстов, выделение строк, слов и символов фрагментов текстов.

Для **поворота изображения наклонного фрагмента текста**, вначале пользователю необходимо открыть исходное изображение фрагмента текста с помощью кнопки «*Загрузить…*». Затем пользователь нажимает на кнопку «*Повернуть*» для поворота наклонного изображения фрагмента текста. В результате повернутое изображение появляется на поле «*Обработанное изображение*», а размер изображения фрагмента текста, угол и время его поворота – на поле «*Результат*» (рис. 3.16).

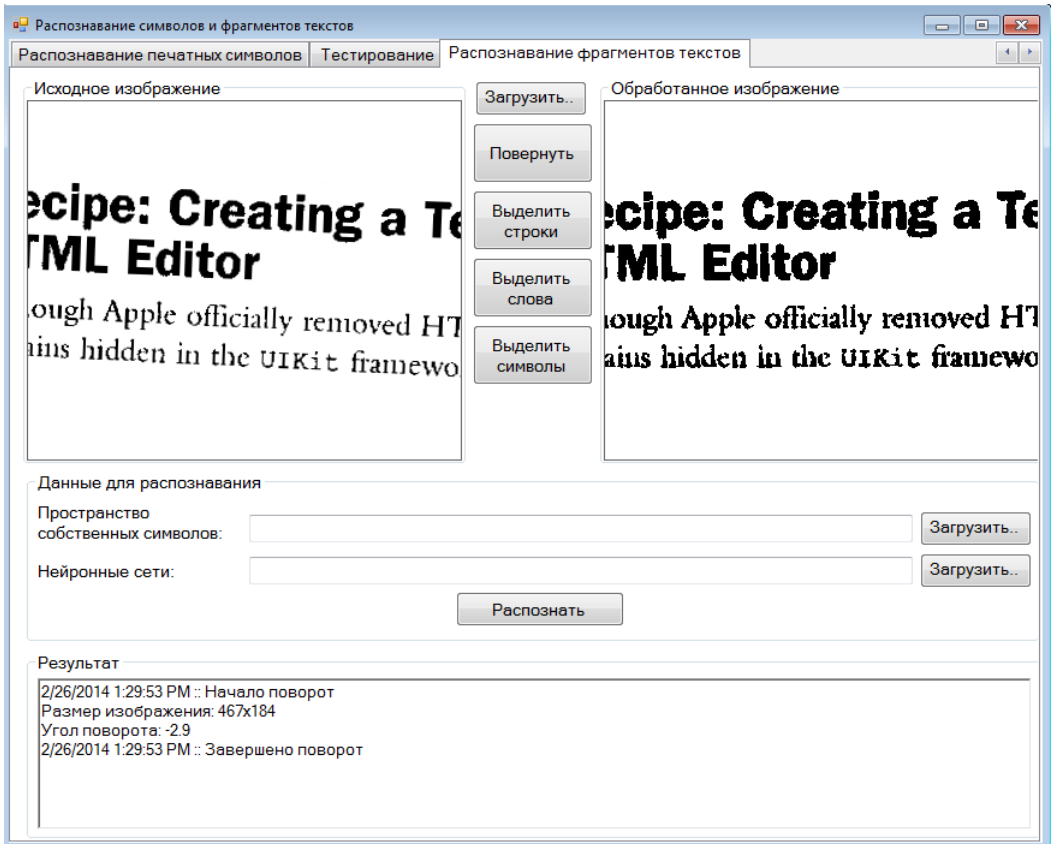


Рисунок 3.16. Поворот наклонного изображения фрагмента текста

Для **выделения строк фрагмента текста** после загрузки исходного изображения фрагмента текста пользователь нажимает на кнопку «*Выделить строки*» для выделения строк фрагмента текста. В результате на поле «*Обработанное изображение*» представляется изображение с границами областей выделенных строк, а на поле «*Результат*» – размер изображения фрагмента текста и время выделения его строк (рис. 3.17).

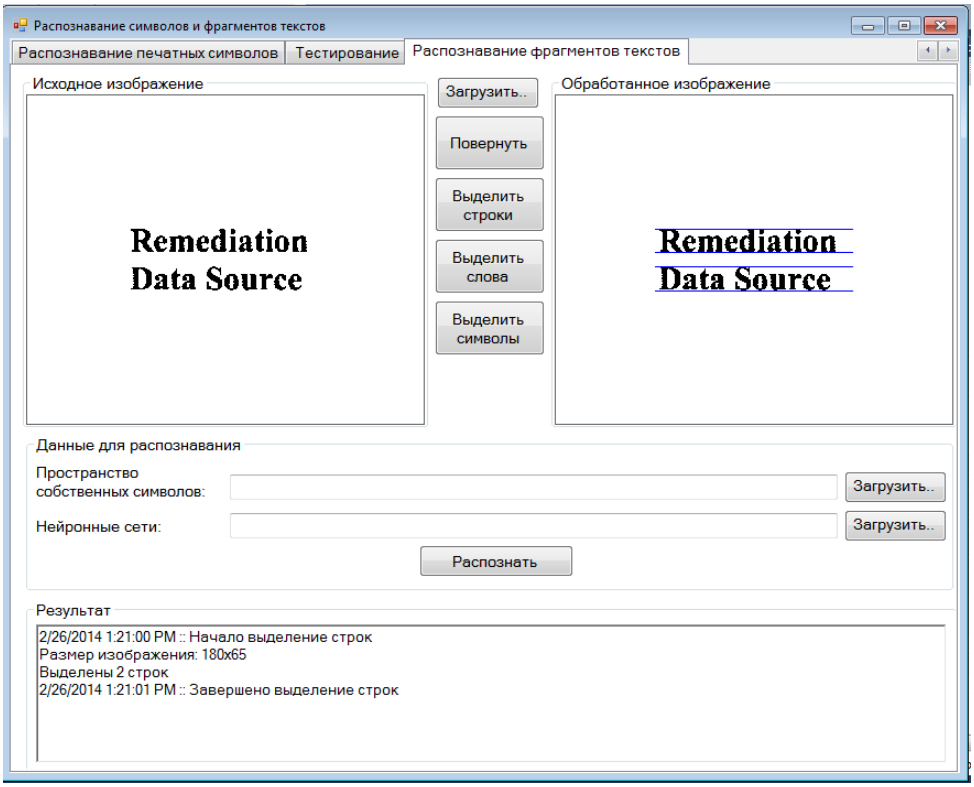


Рисунок 3.17. Выделение строк

Для **выделения слов фрагмента текста**, вначале пользователю необходимо нажать на кнопку «*Загрузить…*», которая открывает стандартный диалог для выбора исходного изображения фрагмента текста. Затем пользователь нажимает на кнопку «*Выделить слова*» для выделения слов фрагмента текста. После того как выделены слова, на поле «*Обработанное изображение*» представляется изображение с границами областей выделенных слов, а на поле «*Результат*» – размер изображения фрагмента текста и время выделения его слов (рис. 3.18).

Для **выделения символов фрагмента текста**, после загрузки исходного изображения фрагмента текста с помощью кнопки «*Загрузить…*», пользователь нажимает на кнопку «*Выделить символы*» для выделения символов фрагмента текста. В результате на поле «*Обработанное изображение*» представляется изображение с границами областей выделенных символов, а на поле «*Результат*» – размер изображения фрагмента текста и время выделения его символов (рис. 3.19).

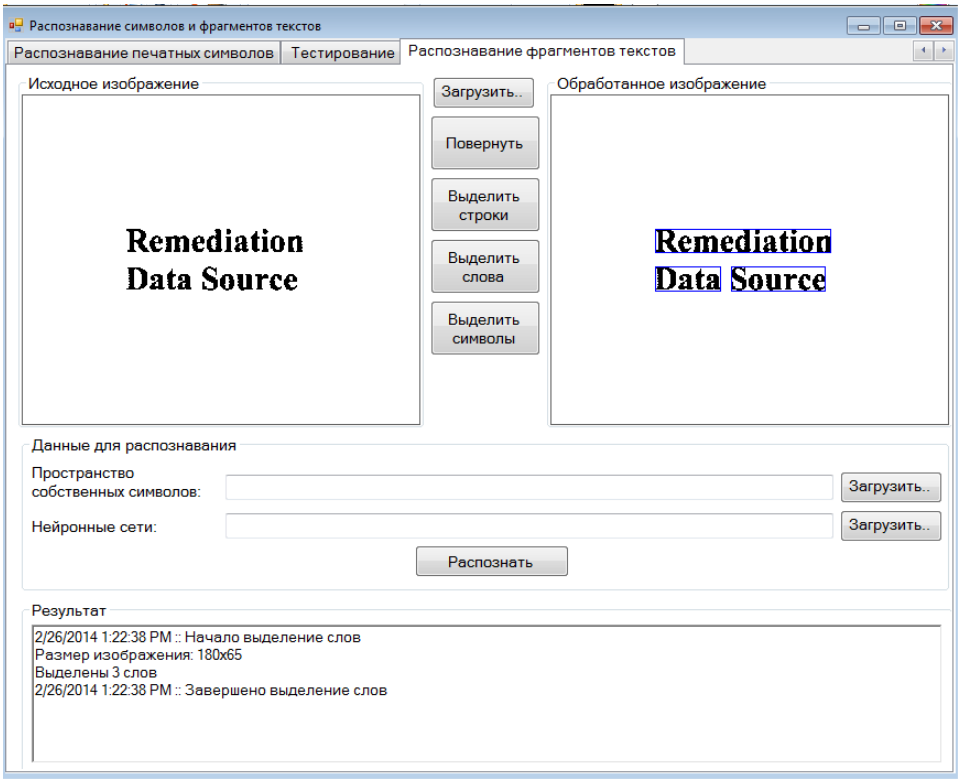


Рисунок 3.18. Выделение слов

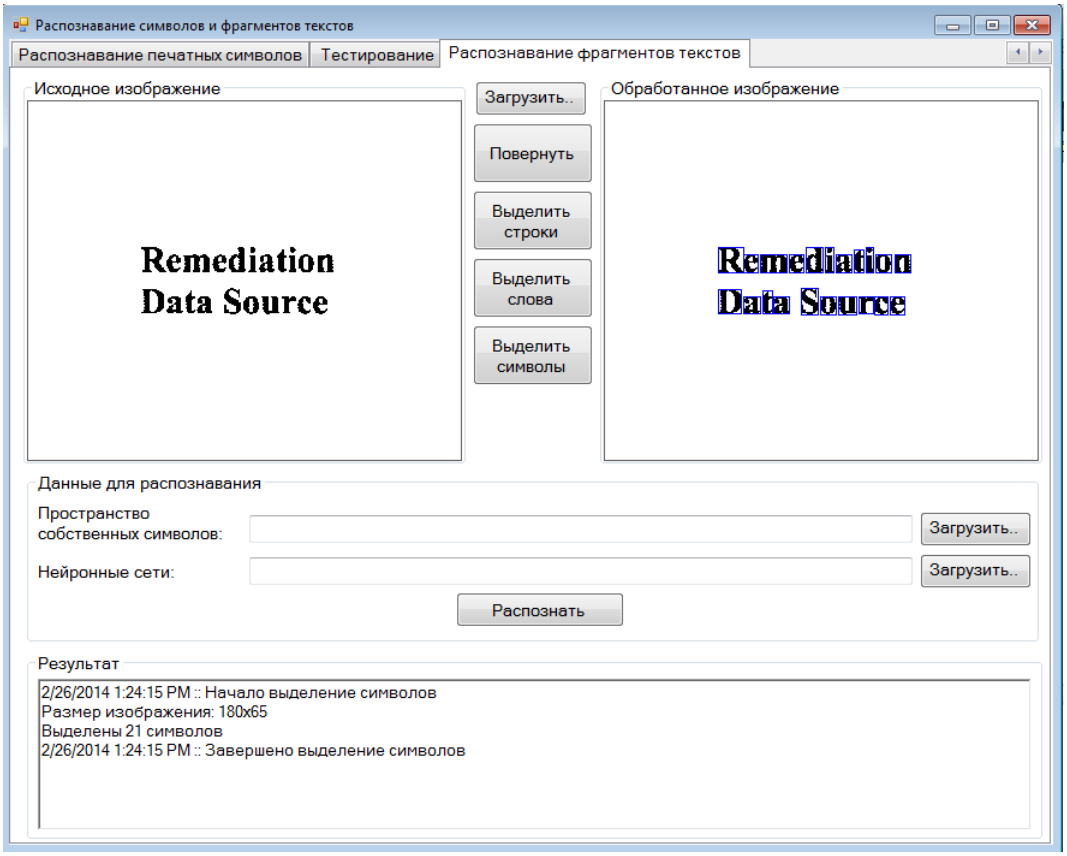


Рисунок 3.19. Выделение символов

Для **распознавания фрагмента текста**, после загрузки исходного изображения фрагмента текста, пользователю необходимо указать пространство собственных символов (поле «*Пространство собственных символов*») и нейронные сети (поле «*Нейронные сети*») с помощью кнопки «*Загрузить…*». Затем пользователь нажимает на кнопку «*Распознать*» для распознавания фрагмента текста.

В результате на поле «*Результат*» показываются размер изображения фрагмента текста, время его распознавания и результат распознавания (рис. 3.20).

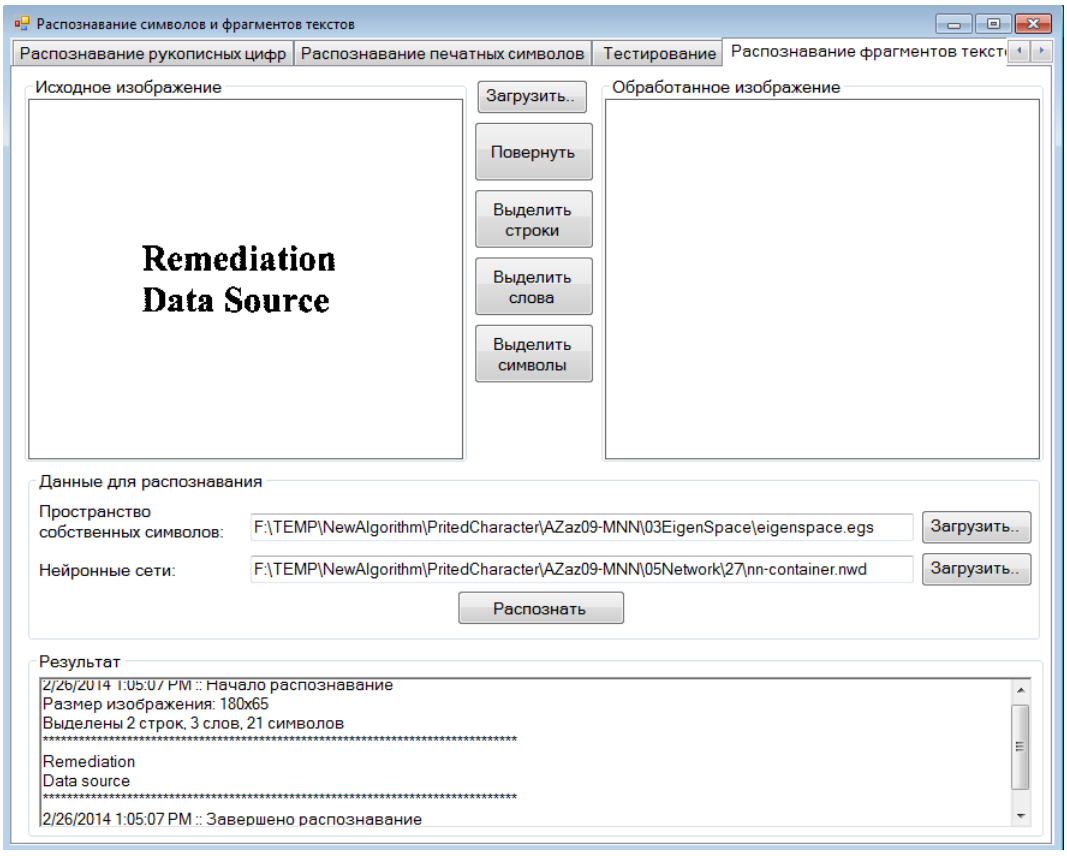


Рисунок 3.20. Распознавание фрагмента текста

3.3.2 Приложение для обычных пользователей

Приложение для обычных пользователей имеет внешний вид интерфейса, состоящей из следующих компонент: поля «*Исходное изображение*» и «*Результат распознавания*», и две кнопки «*Загрузить…*» и

«*Распознать*» (рис. 3.21). Обычным пользователям не нужно указать используемые пространство собственных символов и нейронные сети. Вначале пользователю необходимо загрузить изображение фрагмента текста в поле «*Исходное изображение*». Затем ему необходимо нажать на кнопку «*Распознать*». Результат распознавания представляется на поле «*Результат*».

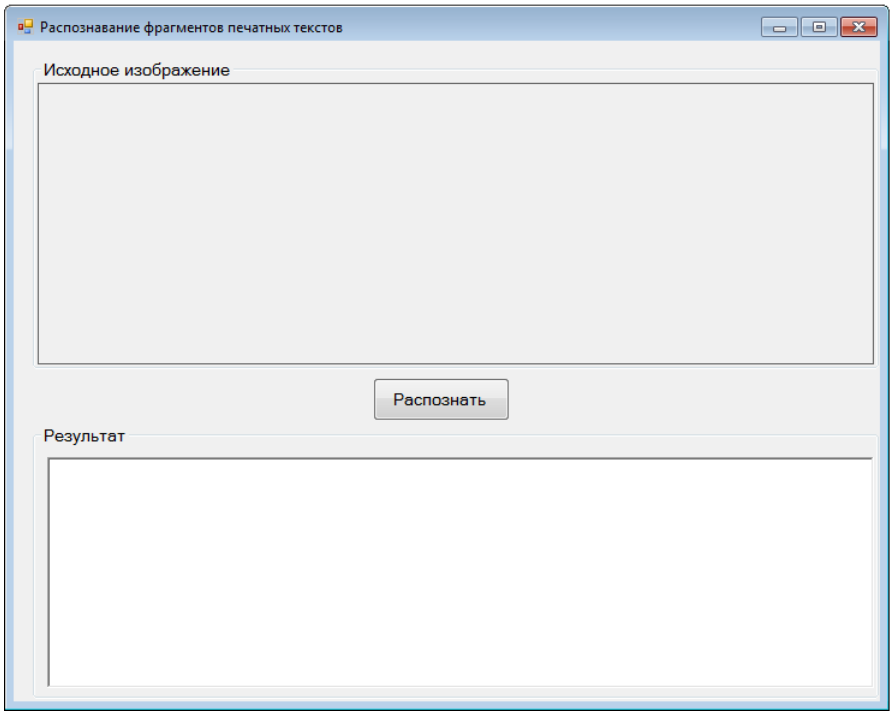


Рисунок 3.21. Интерфейс приложения распознавания фрагмента печатного текста для обычных пользователей

3.4 Основные результаты и выводы по главе 3

Произведен анализ и сравнение особенностей языков программирования: Java, Visual C++ и Visual C#. Сделан аналитический обзор распространенных открытых платформ и библиотек по обработке изображений. На основе анализа языков программирования, платформ и библиотек по обработке изображений произведен выбор средств разработки.

Предложена простая и гибкая архитектура библиотеки для реализации алгоритмов распознавания символов и фрагментов печатных текстов. Классы библиотеки разделены на две группы: классы для выделения символов фрагмента текста и классы для распознавания символов.

Разработаны программные средства, использующие реализованную библиотеку для решения задач распознавания символов и фрагментов печатных текстов. Созданы два приложения: первое используется для исследователей, а второе для обычных пользователей.

Глава 4. Численные эксперименты и анализ результатов распознавания разработанными алгоритмами

В данной главе описываются результаты тестирования разработанных алгоритмов на задачах распознавания рукописных цифр, печатных символов и фрагментов печатных текстов. Представлены данные для обучения и тестирования, приведены таблицы и диаграммы с результатами распознавания. Проводится сопоставление разработанных программных алгоритмов с другими современными алгоритмами распознавания.

Все численные эксперименты проводились на ноутбуке с процессом Intel Core Duo P7350 2.0 ГГц с 2.0 ГБ оперативной памяти. Для оценки эффективности разработанного алгоритма используются как результаты распознавания без учета второго предположения, так и результаты с его учетом. Вторым предположением является символ обучающей выборки, нейронная сеть которого возвращает второе наибольшее значение. При использовании результата с учетом второго предположения правильным распознаванием является случай, в котором либо первое, либо второе предположение дает правильный ответ.

4.1 Тестирование на задаче распознавания рукописных цифр

4.1.1 Обучающая выборка

Первым экспериментом является тестирование разработанного алгоритма на задаче **распознавания рукописных цифр**. Для проведения этого тестирования использовалась известная база рукописных цифр MNIST [70]. Данная база состоит из 60000 изображений для обучения и 10000 изображений для тестирования. Все изображения имеют одинаковый размер 28×28 пикселей и все цифры центрированы внутри изображения. На рис. 4.1. представлены примеры изображений рукописных цифр обучающей выборки

из базы MNIST.

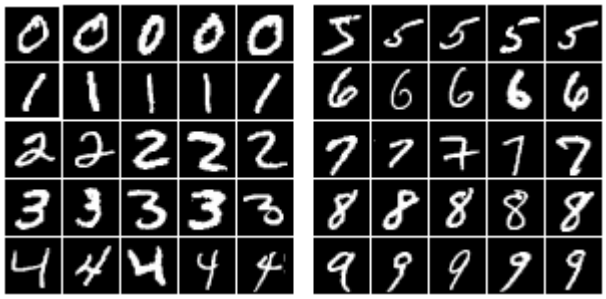


Рисунок 4.1. Примеры изображений рукописных цифр обучающей выборки

4.1.2 Описание тестирования

Для тестирования, кроме исходной тестовой выборки базы MNIST, созданы дополнительные тестовые выборки путем добавления к изображениям этой базы шума «соли и перца» с уровнем 5, 10, 15, 20, 25 и 30%. На рис. 4.2 приведены примеры изображений использованных тестовых выборок. На рис. 4.2(а) представлены изображения исходной тестовой выборки базы MNIST, а на рис. 4.2(б–в) представлены изображения с уровнем шума 15 и 30%.

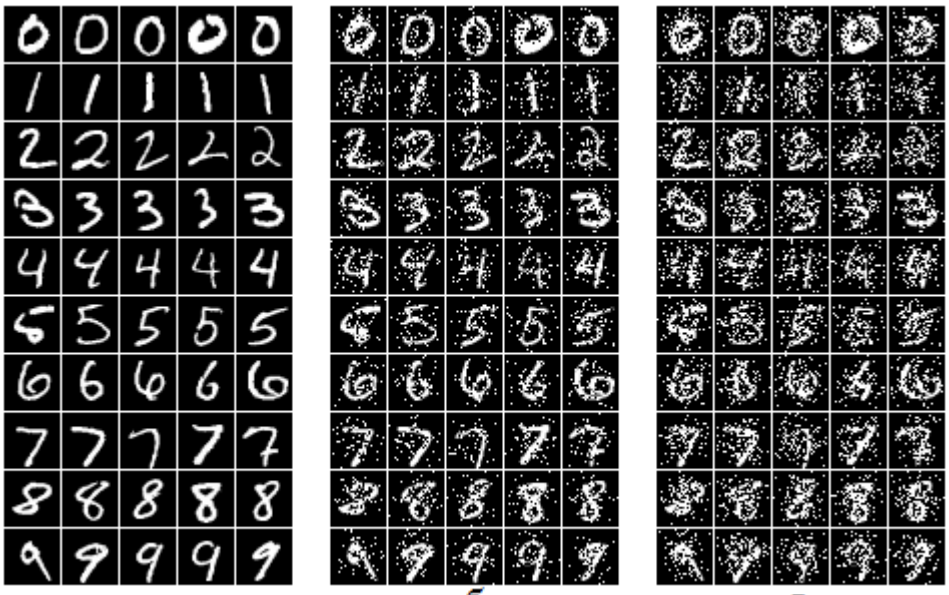


Рисунок 4.2. Примеры использованных рукописных цифр тестовых выборок

4.1.3 Результаты тестирования

Результаты распознавания разработанным алгоритмом рукописных цифр из базы MNIST в зависимости от размерности вектора признаков символов представлены на рис. 4.3. По вертикальной оси отсчитывается точность распознавания в процентах (δ, %), а по горизонтальной оси – количество использованных признаков (K).

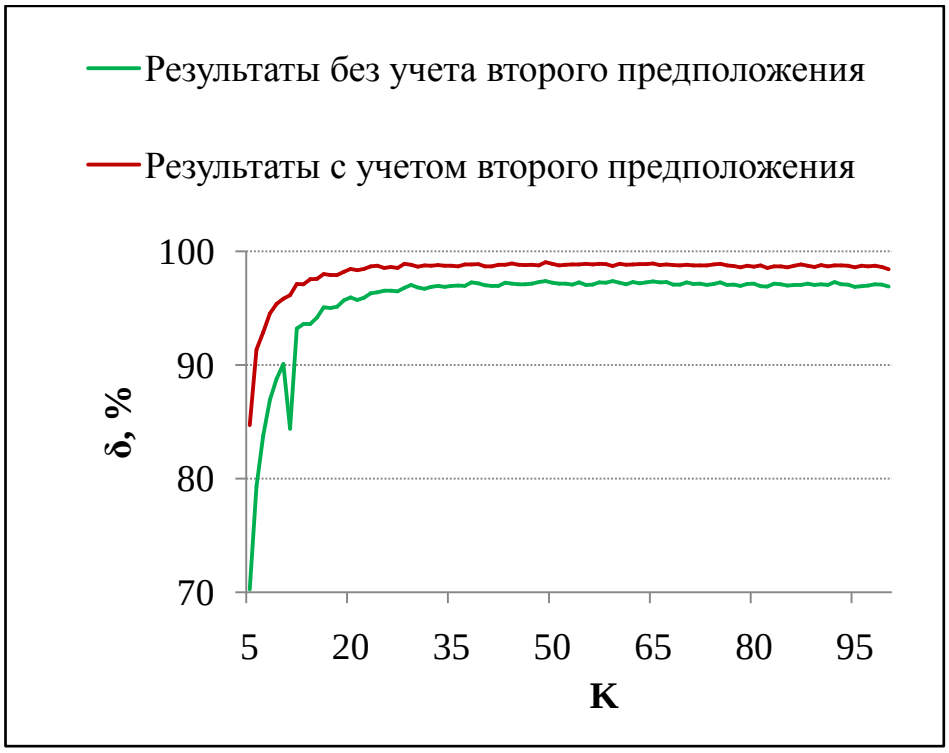


Рисунок 4.3. Результаты распознавания разработанным алгоритмом рукописных цифр

Показано, что точность распознавания разработанным алгоритмом зависит от размерности вектора признаков символов. При использовании вектора из 5 признаков точность распознавания составляет только 70,3%. При увеличении количества использованных признаков точность распознавания увеличивается. Когда количество использованных признаков равно 37, точность распознавания составляет 97%. Результаты проведенных экспериментов показывают более устойчивую работу алгоритма при увеличении количества использованных признаков. При количестве

признаков больше 37 точность распознавания находится в пределах от 97% до 97,5%. При использовании вектора из 49 признаков точность распознавания достигает 97,5%.

В случае использования результатов распознавания с учетом второго предположения, точность распознавания разработанным алгоритмом существенно увеличивается. При использовании вектора из 5 признаков точность распознавания уже составляет 84,7%. Для количества использованных признаков равного 37 получен результат распознавания 98,8%. При использовании количества признаков больше 37 точность распознавания устойчиво находится в пределах от 98,8% до 99%. При размере вектора признаков равном 49 также как и в предыдущем случае получен наилучший результат распознавания – 99%.

Таким образом, для проведения дальнейших численных экспериментов по распознаванию рукописных цифр на зашумленных выборках выбран вектор, состоящий из 49 признаков.

В работе Ю.А. Болотовой [4] предложена сеть иерархической временной памяти (ИВП), также протестированная на задаче распознавания рукописных цифр из базы MNIST. При этом обучение сети ИВП проводилось различными алгоритмами, такими как *Жадный алгоритм кластеризации* (Greedy), *Aglomerative Hierarchical Clustering* (AHC) и *Maximum Temporal Connection* (MTC). Результаты сопоставления предложенного алгоритма и сети ИВП по распознаванию рукописных цифр из базы MNIST приведены в табл. 4.1.

В результате сопоставления показано, что разработанный алгоритм обучается в 13 раз быстрее и распознает цифры в 15 раз быстрее, чем сеть ИВП. Результат распознавания цифр разработанным алгоритмом составляет 97,5% и он сопоставим с результатом их распознавания сетью ИВП, обученной алгоритмами Greedy и AHC. Точность распознавания разработанным алгоритмом с учетом второго предположения является наилучшей и составляет 99%.

Таблица 4.1. Результаты распознавания выборки MNIST разработанным алгоритмом и сетью ИВП

Процессор	**Алгоритм**	**Точность, %**	**Время[1]**	
			обучения	**тестирования**
Intel Core™ 3.47ГГц	Сеть ИВП (Greedy)	97,3	05:34:12	01:38:43
	Сеть ИВП (AHC)	97,6	05:15:17	01:30:56
	Сеть ИВП (MTC)	98,5	05:21:47	01:32:35
Intel Core Duo P7350 2.0 ГГц	Разработанный алгоритм	97,5	00:24:36	00:06:08
	Разработанный алгоритм (с учетом второго предположения)	99,0	00:24:36	00:06:08

Полученные результаты также были сопоставлены с другими алгоритмами, протестированными на базе рукописных цифр MNIST [4, 69]. Результаты сопоставления различных алгоритмов приведены в табл. 4.2. Точность распознавания разработанным алгоритмом сравнима с лучшими алгоритмами распознавания.

Таблица 4.2. Сравнение различных алгоритмов распознавания на базе MNIST

Алгоритм	**Минимальная ошибка, %**	**Максимальная ошибка, %**
Линейный классификатор	7,6	12
Нелинейный классификатор	3,3	3,6
Сети ИВП	1,5	1,5
Boosted stumps	0,87	7,7
K-Nearest Neighbors	0,63	5
SVM	0,56	1,4
Нейронные сети	0,35	4,7
Сверточные сети	0,23	1,7
Разработанный алгоритм	2,5	3
Разработанный алгоритм (с учетом второго предположения)	1	1,2

[1] Время обучения и тестирования разработанного алгоритма включает время на создание пространства собственных символов, выделение признаков, уменьшение размерности вектора признаков, обучение нейронных сетей и распознавание нейронными сетями

В этом эксперименте также проводилось тестирование разработанного алгоритма на созданных зашумленных выборках рукописных цифр базы MNIST. При тестировании использовался вектор из 49 признаков. Результаты распознавания рукописных цифр разработанным алгоритмом на зашумленных выборках представлены на рис. 4.4. Следует отметить, что разработанный алгоритм обеспечивает возможность распознавания рукописных цифр в присутствии шума на изображениях. При этом точность распознавания существенно уменьшается, когда уровень шума превышает 20%.

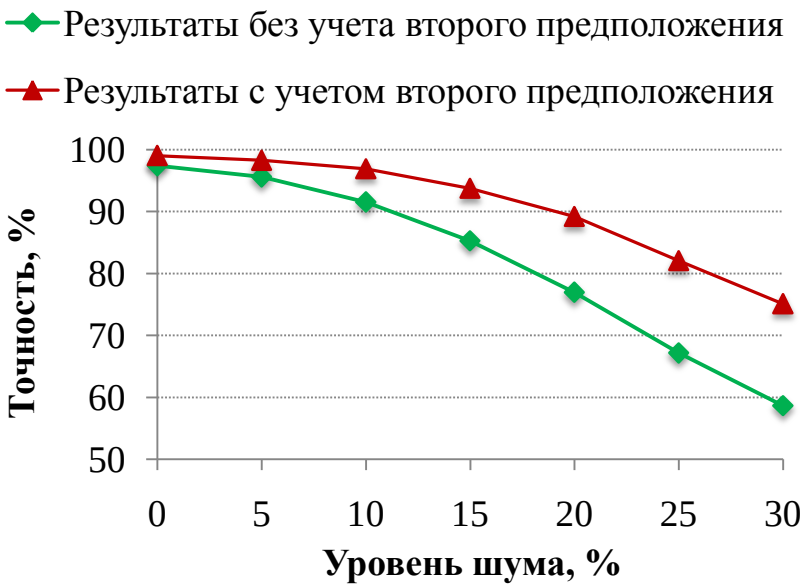


Рисунок 4.4. Результаты распознавания рукописных цифр зашумленных выборок

4.2 Тестирование на задаче распознавания печатных символов

4.2.1 Обучающая выборка

При оформлении документов широко используются два типа шрифтов: шрифт с засечками и шрифт без засечек. Шрифты с засечками, например, Times New Roman, Garamond, Courier New и Bookman Old Style, имеют маленькие черточки по краям. Эти шрифты обычно используются для

оформления основного текста документа, потому что они наиболее читаемы. Шрифты без засечек, например, Arial, Lucida Sans, Tahoma и Verdana, состоят из линий без маленьких черточек по краям. Эти шрифты без засечек обычно используются для оформления заголовков документа.

Для обучения алгоритма создана обучающая выборка, состоящая из 1488 изображений 10 цифр (0–9) и 52 английских букв (a–z, A–Z). Каждый символ представлен двумя распространенными шрифтами Times New Roman и Arial в обычном и полужирном начертаниях (рис. 4.5) с размерами шрифта: 16, 18, 20, 22, 24 и 26 (рис. 4.6). Каждый символ обучающей выборки представлен 24 изображениями.

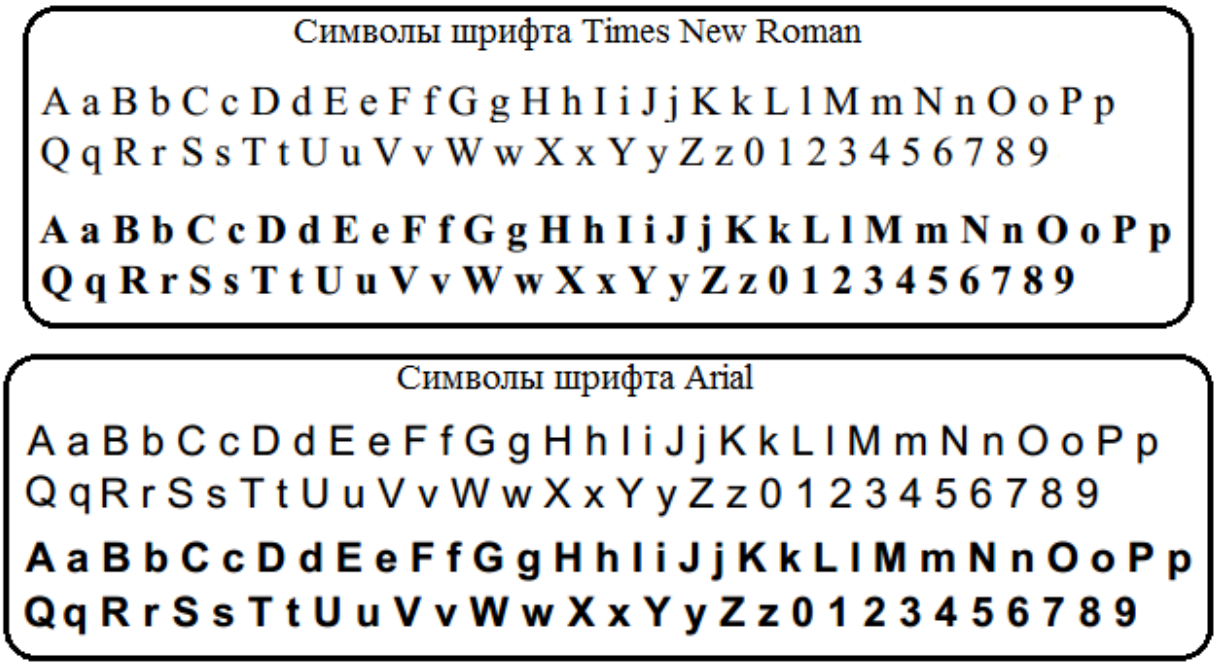


Рисунок 4.5. Примеры использованных символов для создания обучающей выборки

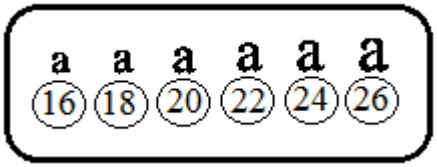


Рисунок 4.6. Пример использованных размеров шрифта для символа «a»

4.2.2 Описание тестирования

Для тестирования использовались изображения символов 8 популярных шрифтов: 4 шрифта с засечками – Times New Roman; Garamond; Courier New; Bookman Old Style; и 4 шрифта без засечек– Arial; Lucida Sans; Tahoma; Verdana. Для каждого шрифта подготовлена тестовая выборка, содержащая 2480 изображений 10 цифр (0–9) и 52 английских букв (a–z, A–Z). Каждый символ представлен в разных размерах шрифта: 12, 14, 16, 18, 20, 22, 24, 26, 28 и 36 в обычном и полужирном начертаниях. Примеры изображений символа «A» разных размеров представлены на рис. 4.7.

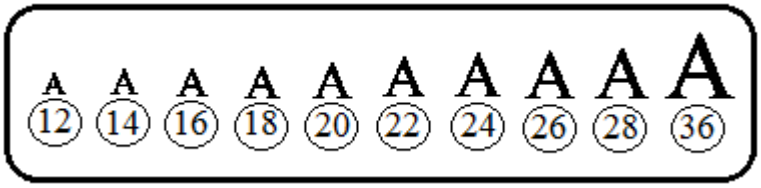


Рисунок 4.7. Пример использованных размеров шрифта для символа «A»

Для каждого шрифта созданы дополнительные тестовые выборки путем добавления 5, 10, 15, 20, 25 и 30% шума типа «соли и перца» к изображениям существующей тестовой выборки. Примеры зашумленных изображений печатных символов приведены на рис. 4.8. Слева-направо на рис. 4.8 представлены изображения печатных символов с уровнем шума 15 и 30%.

Рисунок 4.8. Примеры изображений печатных символов тестовой выборки с уровнем шума 15 и 30%

4.2.3 Результаты тестирования

На рис. 4.9 приведены результаты распознавания печатных символов разных шрифтов в зависимости от размерности вектора признаков символа. По вертикальной оси отсчитывается точность распознавания в процентах (δ, %), а по горизонтальной оси – количество использованных признаков (K).

Экспериментальные результаты показывают, что разработанный алгоритм, обученный только символами двух шрифтов, может распознавать символы других шрифтов. Показано, что кроме шрифта Bookman Old Style, для всех остальных шрифтов, когда количество использованных признаков символа находится в пределах от 20 до 60, точность распознавания разработанным алгоритмом становится приемлемой. Результаты распознавания символов, представленных шрифтами без засечек, лучше и устойчивее, чем результаты распознавания шрифтов с засечками.

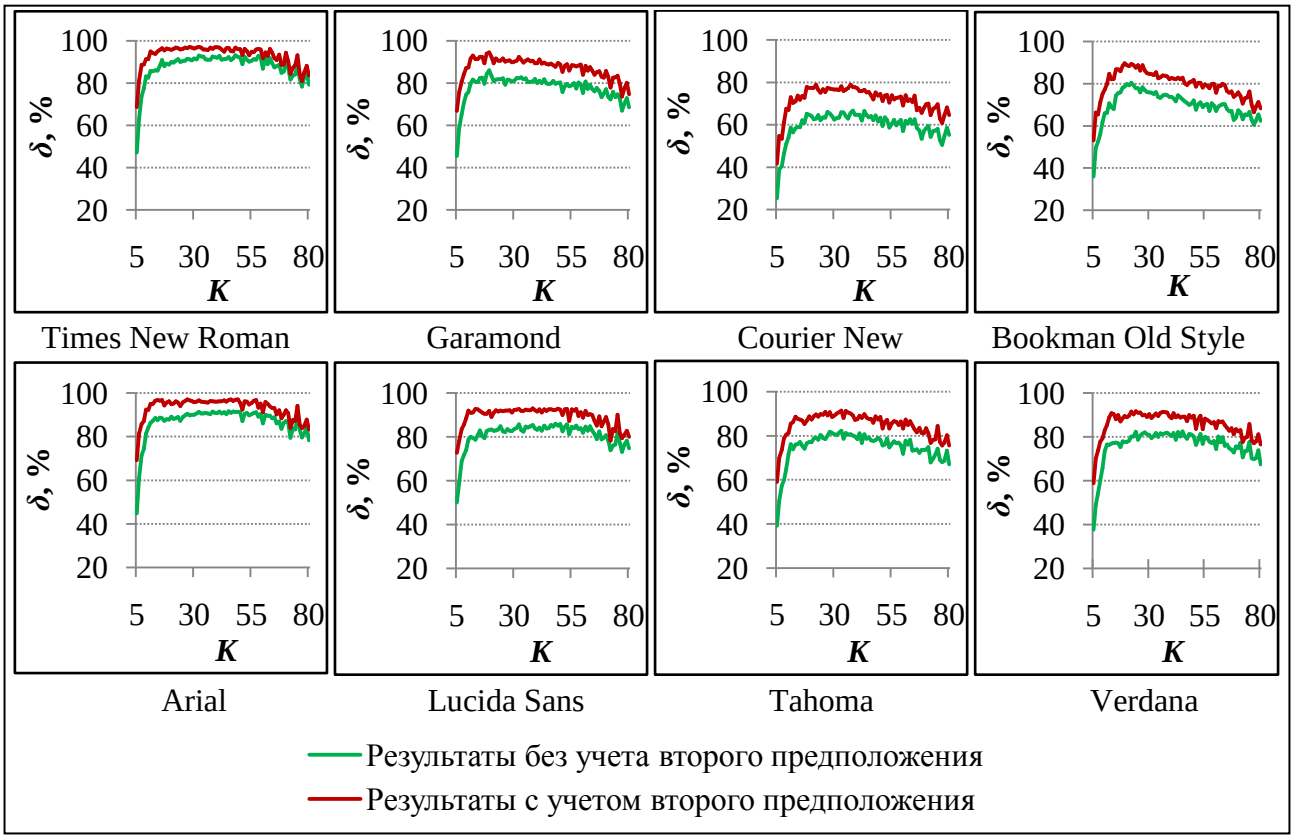


Рисунок 4.9. Результаты распознавания печатных символов разных шрифтов

Результаты распознавания с учетом второго предположения всегда превосходят результаты распознавания без его учета. Наилучший результат распознавания для большинства шрифтов получен при использовании вектора из 27 признаков. Таким образом, вектор, состоящий из 27 признаков, выбран для дальнейшего тестирования печатных символов на зашумленных выборках.

Результаты тестирования разработанного алгоритма на зашумленных выборках представлены на рис. 4.10. По вертикальной оси отсчитывается точность распознавания в процентах, а по горизонтальной оси – уровень шума на изображениях в процентах. На рис. 4.10(а) представлены результаты распознавания без учета второго предположения, а на рис. 4.10(б) – результаты распознавания с его учетом.

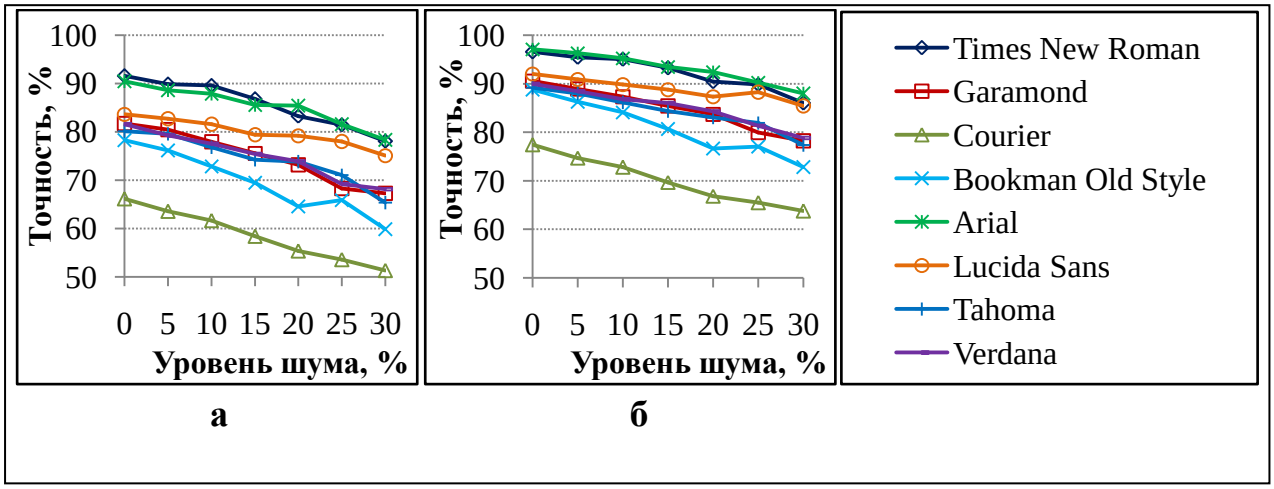


Рисунок 4.10. Результаты распознавания печатных символов зашумленных выборок

Показано, что разработанный алгоритм способен эффективно распознавать печатные символы разных шрифтов в присутствии шума на изображениях. Точность распознавания зависит от уровня шума на изображениях. Чем больше уровень шума на изображениях, тем меньше точность распознавания. В присутствии 5% шума на изображениях, точность распознавания разработанным алгоритмом больше 75% для 7 шрифтов, а в присутствии 30% – только для 3 шрифтов. При использовании результатов

распознавания с учетом второго предположения точность распознавания существенно увеличивается.

Результаты сопоставления разработанного алгоритма и систем распознавания текста ABBY FineReader 11 и Tesseract OCR по распознаванию символов двух шрифтов Times New Roman и Arial на зашумленных выборках представлены на рис. 4.11.

Показано, что при увеличении уровня шума на изображениях точность распознавания системами ABBY FineReader 11 и Tesseract OCR существенно уменьшается, а точность распознавания разработанным алгоритмом падает медленнее. Разработанный алгоритм распознает печатные символы в присутствии шума на изображениях эффективнее, чем системы ABBY FineReader 11 и Tesseract OCR. При уровне шума больше 15% разница между их результатами распознавания становится более заметной.

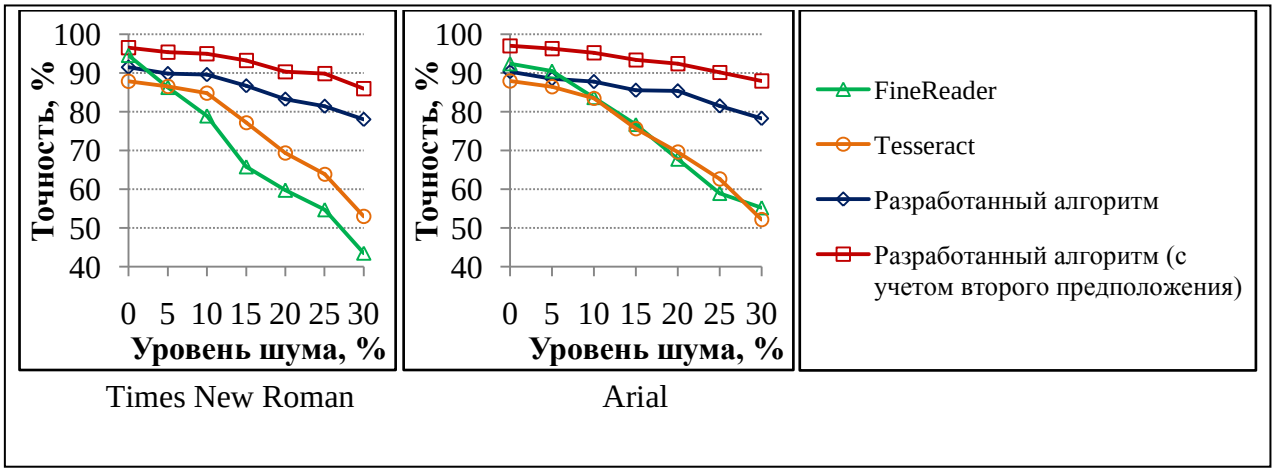


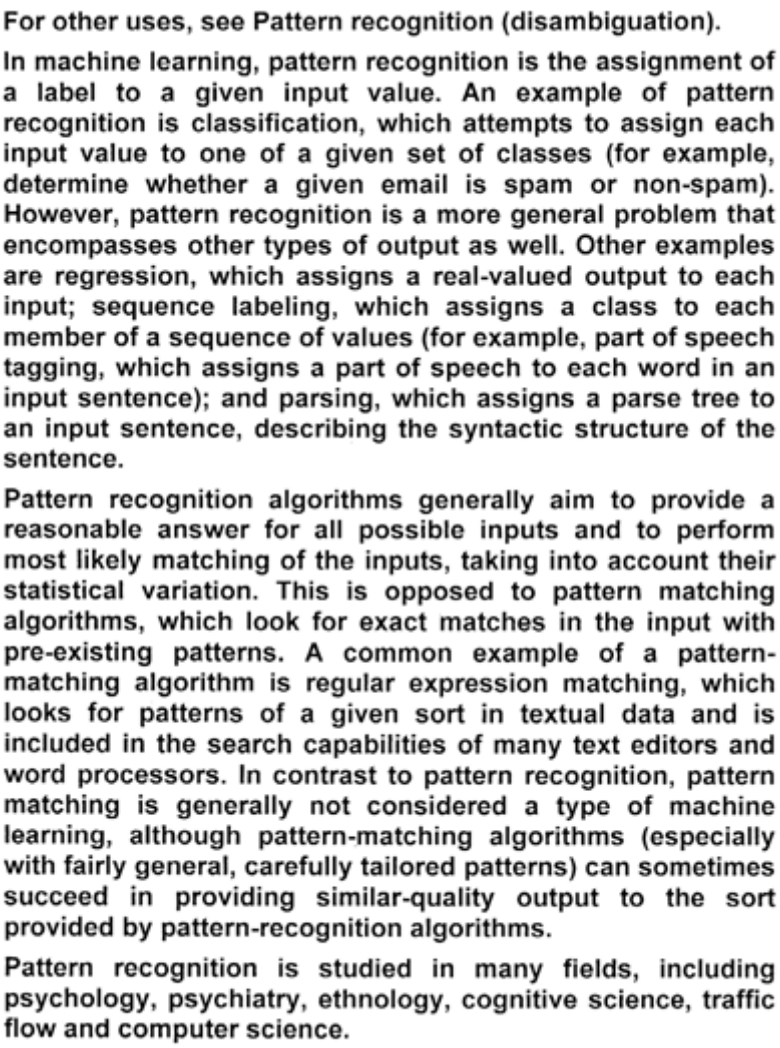
Рисунок 4.11. Сравнение результатов распознавания зашумленных печатных символов

4.3 Тестирование на задаче распознавания фрагментов печатных текстов

4.3.1 Описание тестирования

В данной главе также проведены эксперименты **по распознаванию фрагментов печатных текстов.** Для тестирования разработанного алгоритма распознавания фрагментов текста использовались

отсканированные документы. Фрагменты текстов имеют одинаковое содержание. Каждый фрагмент включает 1581 символ, набранный в текстовом редакторе Microsoft Office Word 2007. При наборе текста фрагментов использовались два популярных шрифта: Times New Roman и Arial с размерами шрифта: 14, 16, 18, 20, 22, 24, 26, 28 и 36 в обычном и полужирном начертаниях. Эти фрагменты текстов распечатывались, затем полученные документы сканировались с разрешением 300dpi и сохранялись в файлах в формате «bmp». Таким образом, тестовая выборка имеет всего 36 фрагментов печатных текстов. Пример использованного фрагмента печатного текста представлен на рис. 4.12.



For other uses, see Pattern recognition (disambiguation).

In machine learning, pattern recognition is the assignment of a label to a given input value. An example of pattern recognition is classification, which attempts to assign each input value to one of a given set of classes (for example, determine whether a given email is spam or non-spam). However, pattern recognition is a more general problem that encompasses other types of output as well. Other examples are regression, which assigns a real-valued output to each input; sequence labeling, which assigns a class to each member of a sequence of values (for example, part of speech tagging, which assigns a part of speech to each word in an input sentence); and parsing, which assigns a parse tree to an input sentence, describing the syntactic structure of the sentence.

Pattern recognition algorithms generally aim to provide a reasonable answer for all possible inputs and to perform most likely matching of the inputs, taking into account their statistical variation. This is opposed to pattern matching algorithms, which look for exact matches in the input with pre-existing patterns. A common example of a pattern-matching algorithm is regular expression matching, which looks for patterns of a given sort in textual data and is included in the search capabilities of many text editors and word processors. In contrast to pattern recognition, pattern matching is generally not considered a type of machine learning, although pattern-matching algorithms (especially with fairly general, carefully tailored patterns) can sometimes succeed in providing similar-quality output to the sort provided by pattern-recognition algorithms.

Pattern recognition is studied in many fields, including psychology, psychiatry, ethnology, cognitive science, traffic flow and computer science.

Рисунок 4.12. Пример использованного фрагмента печатного текста

4.3.2 Результаты тестирования

Тестирование разработанного алгоритма распознавания фрагмента печатного текста и систем распознавания текста ABBY FineReader 11 и Tesseract осуществлялось на подготовленных 36 фрагментах текстов. Результаты сопоставления распознавания фрагментов печатных текстов, набранных шрифтом Arial, приведены в табл. 4.3. На основе сопоставления результатов распознавания показано, что точность распознавания, достигнутая системой FineReader 11, является наилучшей и составляет 99,94–100%, точность распознавания системой Tesseract составляет 99,87–100%, а точность распознавания разработанным алгоритмом составляет 97,66–99,62%.

Таблица 4.3. Результаты распознавания фрагментов печатных текстов шрифта Arial

Начертание	Размер шрифта	ABBY FineReader 11 (%)	Tesseract OCR (%)	Разработанный алгоритм (%)
Обычное	14	100	99,87	97,66
	16	100	99,94	98,48
	18	100	100	98,99
	20	100	99,94	98,99
	22	100	99,94	98,36
	24	100	100	99,11
	26	100	99,94	99,37
	28	100	100	99,49
	36	100	100	99,56
Полужирное	14	100	100	98,99
	16	100	100	98,92
	18	100	100	99,24
	20	100	100	99,30
	22	100	100	99,62
	24	100	100	99,05
	26	100	100	98,86
	28	99,94	100	99,49
	36	99,94	100	99,43

Наибольшее количество ошибок, возникающих при распознавании разработанным алгоритмом фрагментов печатных текстов шрифта Arial, наблюдается для таких пар символов, как i и j, i и l, e и o, I и l и неразделенных букв rt, rf, ff и ry.

В табл. 4.4 приведены результаты сопоставления распознавания фрагментов печатных текстов, набранных шрифтом Times New Roman. На основе сопоставления результатов распознавания показано, что точность распознавания, достигнутая системой FineReader 11, также является наилучшей и составляет 99,87–100%, точность распознавания системой Tesseract составляет 99,43–99,62%, а точность распознавания разработанным алгоритмом составляет 90,89–98,17%.

Таблица 4.4. Результаты распознавания фрагментов печатных текстов шрифта Times New Roman

Начертание	**Размер шрифта**	**ABBY FineReader 11 (%)**	**Tesseract OCR (%)**	**Предложенный алгоритм (%)**
Обычное	14	99,87	99,62	90,89
	16	100	99,62	93,36
	18	100	99,43	92,98
	20	99,94	99,43	91,78
	22	100	99,49	92,60
	24	100	99,56	92,54
	26	100	99,56	96,02
	28	99,87	99,43	96,71
	36	99,87	99,43	96,02
Полужирное	14	100	99,43	96,71
	16	100	99,49	95,57
	18	100	99,49	96,46
	20	100	99,62	96,77
	22	100	99,49	97,52
	24	100	99,49	97,79
	26	100	99,49	97,72
	28	100	99,49	97,41
	36	100	99,62	98,17

Наибольшее количество ошибок, возникающих при распознавании разработанным алгоритмом фрагментов печатных текстов шрифта Times New Roman, наблюдается для таких пар символов, как 1 и l, t и f, r и f, h и b, n и H и неразделенных букв rn, rm, rt, ry, fi, fo, fa, fu и ffi.

На рис. 4.13 представлены усредненные результаты распознавания фрагментов печатных текстов. Результаты распознавания фрагментов текста шрифта Times New Roman в обычном и полужирном начертаниях представлены на рис. 4.13 столбцами 1 и 2. Результаты распознавания фрагментов текста шрифта Arial в обычном и полужирном начертаниях представлены на рис. 4.13 столбцами 3 и 4.

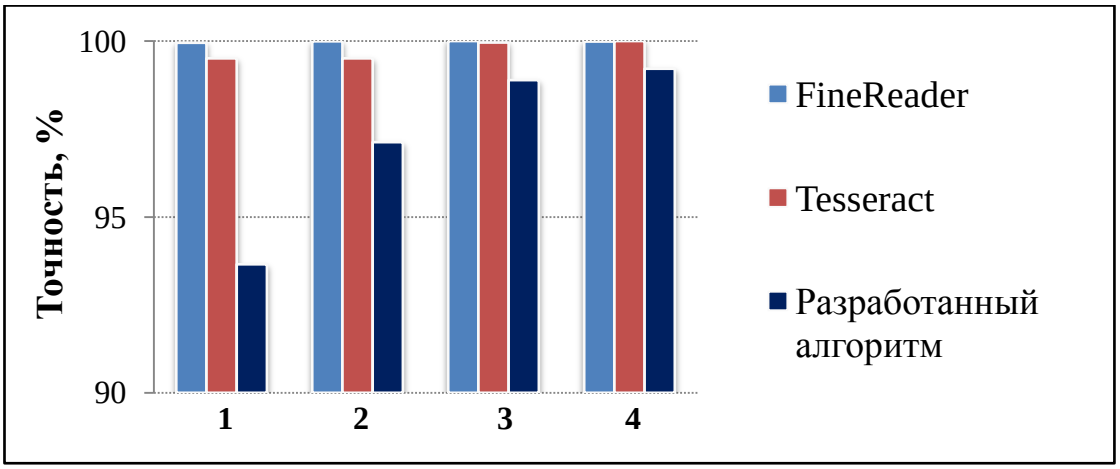


Рисунок 4.13. Сравнение результатов распознавания фрагментов печатных текстов

Следует отметить, что для шрифта Arial результаты распознавания разработанным алгоритмом лучше, чем для шрифта Times New Roman. Это объясняется тем, что шрифт Times New Roman является шрифтом с засечками, из-за которых количество неразделенных букв (rn, rm, rt, ry, fi, fo, fa, fu и ffi) для шрифта Times New Roman больше, чем количество неразделенных букв (rt, rf, ff и ry) для шрифта Arial.

Показано, что средняя точность распознавания, достигнутая системой ABBY FineReader 11, составляет 99,95–100%, точность распознавания системой Tesseract OCR составляет 99,51–99,99%, а точность распознавания разработанным алгоритмом составляет 93,65–99,21%.

4.4 Основные результаты и выводы по главе 4

В данной главе проведено тестирование разработанного алгоритма при распознавании рукописных цифр базы MNIST. Численные эксперименты показали, что при использовании вектора из 49 признаков получен наилучший результат распознавания – 97,5% без учета второго предположения и 99% – с учетом второго предположения.

Результаты экспериментов также показали, что разработанный алгоритм позволяет распознавать рукописные цифры в присутствии шума на изображениях. Следует отметить, что точность распознавания существенно уменьшается, когда уровень шума на изображениях превышает 20%.

В данной главе проведено тестирование разработанного алгоритма на задаче распознавания печатных символов, набранных 8 разными шрифтами. Экспериментальные результаты показывают, что разработанный алгоритм, обученный только символами двух шрифтов, может распознавать символы других шрифтов. Результаты распознавания символов, представленных шрифтами без засечек, лучше и устойчивее, чем результаты распознавания шрифтов с засечками.

Результаты распознавания с учетом второго предположения всегда превосходят результаты распознавания без его учета. Наилучший результат распознавания для большинства шрифтов получен при использовании вектора из 27 признаков. Показано, что разработанный алгоритм также способен эффективно распознавать печатные символы разных шрифтов в присутствии шума на изображениях.

Также проведено тестирование разработанного алгоритма при распознавании фрагментов печатных текстов. Результаты численных экспериментов показывают, что средняя точность распознавания фрагментов печатных текстов, достигнутая разработанным алгоритмом, составляет 93,65–99,21%.

ЗАКЛЮЧЕНИЕ

В результате выполнения диссертационной работы были получены следующие основные научные и практические результаты и сделаны следующие выводы:

1. Разработан новый способ построения классификатора для распознавания символов на основе нейронных сетей, отличающийся от других тем, что каждая нейронная сеть соответствует только одному символу обучающей выборки.

2. Разработан оригинальный алгоритм распознавания символов, основанный на вейвлет-преобразовании, методе главных компонент и нейронных сетях.

3. Создан новый алгоритм распознавания фрагментов печатных текстов, основанный на разработанном алгоритме распознавания символов и способе выделения символов из фрагмента текста.

4. Создано программное обеспечение, которое используется для распознавания рукописных цифр, печатных символов и фрагментов печатных текстов.

ОБОЗНАЧЕНИЯ

I_n	n-ое изображение.
$\vec{I}_n$	Вектор, который представляет n-ое изображение.
$\vec{I}_{\text{ср}}$	Вектор, который представляет «среднее» изображение.
$\vec{\Phi}_n$	Вектор, который представляет вычитание векторов n-ого изображения и «среднего» изображения.
$\vec{u}_k$	Собственные векторы ковариационной матрицы.
λ_k	Собственные значения ковариационной матрицы.
C	Ковариационная матрица.
$I_{вх}$	Входное изображение.
$\vec{I}_{вх}$	Одномерный вектор, который представляет входное изображение.
w_i	Коэффициенты разложения по собственным символам.
$\vec{\Omega}^T$	Вектор, описывающий вклад каждого собственного известного символа в представление входного изображения символа.

СПИСОК СОКРАЩЕНИЙ

ИВП	Иерархическая временная память.
МГК	Метод главных компонент.
ПО	Программное обеспечение.
AHC	Aglomerative Hierarchical Clustering.
CAMShift	Continuously Adaptive Mean Shift.
CCA	Connected Component Analysis.
CFNN	Cascaded Forward Neural Network.
FFNN	Feed Forward Neural Network.
Greedy	Жадный алгоритм кластеризации.
KAM	Kernel Associative Memory.
kNN	K-Nearest Neighbor.
LDA	Linear Discriminant Analysis.
LDP	Locally Discriminating Projection.
LPP	Locality Preserving Projection.
MTC	Maximum temporal connection.
OCR	Optical Character Recognition.
PA	Projection Analysis.
SVM	Support Vector Machines.
WKLDP	Wavelet based Kernel Locally Discriminating Projection.
WSP	White Space and Pitch.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. CuneiForm [Электронный ресурс]. – Режим доступа: http://cognitiveforms.ru/products/cuneiform/. Дата обращения: 30.11.2013.

2. FineReader 11 [Электронный ресурс]. – Режим доступа: http://www.abbyy.ru/finereader-professional/. Дата обращения: 30.11.2013

3. **Афонасенко, А.В.** Обзор методов распознавания структурированных символов / А.В. Афонасенко, А.И. Елизаров // Доклады ТУСУРа, – июнь 2008. – Vol. 18. – № 2. – часть 1. – С. 83–88.

4. **Болотова, Ю.А.** Алгоритмы обработки и анализа изображений иерархической временной сетью: диссертация на соискание ученой степени кандидата наук 05.13.01 / Ю.А. Болотова. – Томск, 2013. – 162 с.

5. **Буй, Т.Т.Ч.** Алгоритмическое и программное обеспечение для классификации цифровых изображений с помощью вейвлет-пребразования Хаара и нейронных сетей. / Т.Т.Ч. Буй, Н.Х. Фан, В.Г. Спицын // Известия Томского политехнического университета. – Томск: ТПУ, 2011. – Т. 319. – № 5. – С. 103–106.

6. **Буй, Т.Т.Ч.** Разложение цифровых изображений с помощью двумерного дискретного вейвлет-преобразования и быстрого преобразования Хаара. / Т.Т.Ч. Буй, В.Г. Спицын // Известия Томского политехнического университета. – Томск: ТПУ, 2011. – Т. 318. – № 5. – С. 73–76.

7. **Буй, Т.Т.Ч.** Распознавание лиц и жестов на основе применения вейвлет-преобразования и метода главных компонент. / Т.Т.Ч. Буй, Н.Х. Фан, В.Г. Спицын // Нелинейный мир. – Москва: Радиотехника, 2012. – Т. 10 – № 6. – С. 371–379.

8. **Буй, Т.Т.Ч.** Распознавание лиц на основе применения метода Виолы–Джонса, вейвлет-преобразования и метода главных компонент. / Т.Т.Ч. Буй, Н.Х. Фан, В.Г. Спицын // Известия Томского политехнического университета. – Томск: ТПУ, 2012. – Т. 320. – № 5. – С. 54–59.

9. **Гайдуков, Н.П.** Обзор методов распознавания рукописного

текста / Н.П. Гайдуков, Е.О. Савкова // Международная научно-техническая конференция студентов, аспирантов и молодых учёных «Информационно-управляющие системы и компьютерный мониторинг» – 2012. [Электронные ресурсы] / Н.П. Гайдуков, Е.О. Савкова. – Режим доступа: http://masters.donntu.edu.ua/2012/fknt/gaydukov/library/5_gaydukov.pdf. Дата обращения: 01.09.2012.

10. **Гонсалес, Р.** Цифровая обработка изображений. / Р. Гонсалес, Р. Вудс. – М.: Техносфера, 2005. – 1072 с.

11. **Лукьяница, А.А.** Цифровая обработка видеоизображений. / А.А. Лукьяница, А.Г. Шишкин. – М.: Ай-Эс-Эс Пресс, 2009. – 518 с.

12. **Переберин, А.В.** Многомасштабные методы синтеза и анализа изображений: диссертация на соискание ученой степени кандидата физико-математических наук 05.13.11 / А.В. Переберин. – Москва: Институт прикладной математики им. М.В. Келдыша. – 2002. 138 с.

13. **Столниц Э.,** ДеРоуз Т., Салезин Д. Вейвлеты в компьютерной графике: Пер. с англ.– Ижевск: НИЦ «Регулярная и хаотическая динамика», 2002. – 272 с.

14. **Фан, Н.Х.** Алгоритмы для классификации отпечатков пальцев на основе применения фильтра Габора, вейвлет-преобразования и многослойной нейронной сети / Н.Х. Фан, В.Г. Спицын // Известия Томского политехнического университета. – Томск: ТПУ, 2012. – Т. 320. – № 5. – С. 60–64.

15. **Фан, Н.Х.** Распознавание жестов на видеопоследовательности в режиме реального времени на основе применения метода Виолы-Джонса, алгоритма CAMShift, вейвлет-преобразования и метода главных компонент. / Н.Х. Фан, Т.Т.Ч. Буй, В.Г. Спицын // Вестник Томского государственного университета. – Томск: ТГУ, 2013. – Т. 23. – № 2. – С. 102–111.

16. **Фан, Н.Х.** Распознавание печатных текстов на основе применения вейвлет-преобразования и метода главных компонент. / Н.Х. Фан, Т.Т.Ч. Буй, В.Г. Спицын // Известия Томского политехнического

университета. – Томск: ТПУ, 2012. – Т. 321. – № 5. – С. 154–158.

17. **Филитович, А.Ю.** Распознавание текста: OCR системы [Электронные ресурсы] / А.Ю. Филитович. – Режим доступа: http://it-claim.ru/Education/Course/Lingvistika/Lecture/Lecture11.pdf. Дата обращения: 01.09.2012.

18. **Шокуров, А.В.** Оптимальное использование вейвлет-компонент. / А.В. Шокуров, А.В. Михалев // Успехи математических наук. – 2007. – Т. 62. – № 4. – С. 171–172.

19. Accord.NET Framework [Электронный ресурс]. – Режим доступа: http://code.google.com/p/accord/downloads/list. Дата обращения: 01.03.2012.

20. AForge.NET Framework [Электронный ресурс]. – Режим доступа: http://code.google.com/p/aforge/. Дата обращения: 01.03.2012.

21. OpenCV Library [Электронный ресурс]. – Режим доступа: http://opencv.org/downloads.html. Дата обращения: 01.03.2012.

22. AT&T (Olivetti) Research Laboratories, Cambridge. ORL face databases [Электронный ресурс]. – Режим доступа: http://www.uk.research.att.com/pub/data/orl_faces.zip. Дата обращения: 07.09.2012.

23. Tesseract OCR [Электронный ресурс]. – Режим доступа: https://code.google.com/p/tesseract-ocr/. Дата обращения: 01.03.2012

24. Yale University, Yale face database: [Электронный ресурс]. – Режим доступа: http://cvc.yale.edu/projects/yalefaces/yalefaces.html. Дата обращения: 07.09.2012.

25. **Annadurai, S.** Wavelet Based Enhanced Color Image Compression Relying on Sub-Band Vector Quantization / S. Annadurai, Sundaresan M. // ICGST-GVIP Journal, – 2009. – Vol. 1. – P. 9–16.

26. **Arfan Jaffar, M.** Wavelet-Based Color Image Segmentation using Self-Organizing Map Neural Network / M.Arfan Jaffar, M. Ishtiaq, A. Hussain, A.M. Mirza // 2009 International Conference on Computer Engineering and Applications IPCSIT – IACSIT Press, Singapore, – 2011 – Vol. 2. – P. 403– 434.

27. **Arivazhagan, S.** Texture classification using wavelet transform. / Arivazhagan, S., Ganesan, L. // Pattern Recognition Letter, – 2003. – Vol. 24. – P. 1513–1521.

28. **Aujol, J.-F.** Wavelet-Based Level Set Evolution for Classification of Textured Images / J.-F. Aujol, G. Aubert, L. Blanc-Feraud // IEEE Transactions on Image Processing, – 2003. – Vol. 12. – № 12. – P. 1634–1641.

29. **Baek, K.** PCA vs. ICA: A Comparison on the FERET Data Set. / K. Baek, B.A. Draper, J.R. Beveridge, K. She // Proceedings of the 6th Joint Conference on Information Science (JCIS), – 2002, – P. 824–827.

30. **Banham, M.R.** Spatially adaptive wavelet-based multiscale image restoration. / M.R. Banham, A.K. Katsaggelos // IEEE Trans. ImageProc. – 1996, – Vol. 5 – P. 619–634.

31. **Belhumeur, P.** Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. / P. Belhumeur, J. Hespanha, D. Kriegman // IEEE Transactions on Pattern Analysis and Machine Intelligence, – 1997 – Vol.19 – № 7 – P. 771–720.

32. **Buccigrossi, R.W.** Image compression via joint statistical characterization in the wavelet domain. / R.W. Buccigrossi, E.P. Simoncelli // IEEE Transaction on Image Processing, – 1999. – Vol. 8. – No. 12.

33. **Bunke, H.** Handbook of Character Recognition and Document Image Analysis / H. Bunke, P.S.P. Wang – World Scientific, USA, 1992. – 883 p.

34. **Chang, S.G.** Spatially adaptive wavelet thresholding with context modeling for image denoising. / S. G. Chang, B. Yu, M. Vetterli // IEEE Transaction on Image Processing, – Sep 20009. – Vol. 9. – P. 1522–1531.

35. **Chang, T.** Texture analysis and classification with tree-structured wavelet transform. / T. Chang, K. Jay // IEEE Trans. Image Process. – 1993. Vol. 2 – № 4. – P. 429–440.

36. **Chen, C.H.** Image Segmentation Using Multiresolution Wavelet Analysis and Expectation-Maximization (EM) / C.H. Chen, G.G. Lee // Algorithm for Digital Mammography, Wiley, – 1997.

37. **Chen, Y.-L.** Color Image Segmentation Using Wavelet Transform Techniques. / Y.-L. Chen// 16th IPPR Conference on Computer Vision, Graphics and Image Processing, – 2003. – P.669–675.

38. **Cheriet, M.** Character Recognition Systems A Guide for Students and Practioners / M. Cheriet, N. Kharma, C.-L. Liu, C.Y. Suen – Wiley, Canada, 2007. – 327 p.

39. **Choudhary, D.** Performance Analysis of Texture Image Classification Using Wavelet Feature / D. Choudhary, A.K. Singh, S. Tiwari, V.P. Shukla // International Journal of Image, Graphics and Signal Processing. – 2013. – Vol. 1. – P. 58–63.

40. **Christensen, O.** Approximation Theory, From Taylor Polynomials to Wavelets. / O. Christensen, – Birkhäuser, Boston, – 2004.

41. **Chuang, G.** Wavelet descriptor of planar curves: Theory and applications. / G. Chuang, C.-H. Kuo // IEEE Transaction on Image Processing, – 1996. – Vol. 5. – P. 56–70.

42. **Cope, G.** Optical Character Recognition using PCA [Электронные ресурсы] / G. Cope – Режим доступа: http://www.algosome.com/articles/optical-character-recognition-java.html. Дата обращения: 01.12.2013.

43. **Daniel, M.R.S.** ANN and SVM Based War Scene Classification using Wavelet Features: A Comparative Study / M.R.S. Daniel, A. Shanmugam // Journal of Computational Information Systems. – 2011. – P. 1402–1411.

44. **Daubechies, I.** Orthonormal bases of compactly supported wavelets, / I. Daubechies // CPAM, – 1998. Vol. 41 – P. 909 – 996.

45. **DeVore, R.** Image Compression through Wavelet Transform Coding. / R. DeVore, B. Jawerth, B. Lucier // IEEE Transactions on Information Theory, – March 1992. – Vol. 38. – P. 719–746.

46. **Duda, R.O.** Use of the Hough transformation to detect lines and curves in pictures. / R.O. Duda, P.E. Hart // Communications of ACM, – 1972. – Vol. 4. – № 1. – P. 11–15.

47. **Fan, G.** Image denoising using local contextual hidden Markov model

in the wavelet domain. / G. Fan, X.-G. Xia // IEEE Signal Processing Letter May – 2001. – Vol. 8. – № 5. – P. 125–128.

48. **Fujisawa, H.** Directional Pattern Matching for Character Recognition revisited / H. Fujisawa, C.-L. Liu // Proceedings of the 7th International Conference on Document Analysis and Recognition. – Edinburgh, Scotland, 2003. – P. 794–798.

49. **Fukunaga, K.** Introduction to Statistical Pattern Recognition, 2nd edition / K. Fukunaga – Academic Press, New York, 1990. – 591 p.

50. **George, L.E.** Image Compression Based on Wavelet, Polynomial and Quadtree. / L.E. George, B. Sultan // Journal of Applied Computer Science and Mathematics, – 2011. Vol. 11. – № 5. – P. 15–20.

51. **Gonzalez, A.C.** Histograms, Wavelets and Neural Networks Applied to Image Retrieval / A.C. Gonzalez, J.H. Sossa, E.M.F. Riveron, O. Pogrebnyak // Proceedings of the 5th Mexican international conference on Artificial Intelligence: Lecture Notes in Computer Science. – 2006. – Vol. 4293. – P. 820–827.

52. **Grossmann, A.** Decomposition of Hardy functions into square integrable wavelets of constant shape. / A. Grossmann, J. Morlet // SIAM Journal of Analysis, – 1984. – Vol. 15 – P. 723–736.

53. **Gumus, E.** Evaluation of face recognition techniques using PCA, wavelets and SVM / E. Gumus, N. Kilic, A. Sertbas, O.N. Ucan // Expert Systems with Applications. – 2010. – Vol. 37. – P. 6404–6408.

54. **Guo, H.** Wavelet based speckle reduction with application to SAR based ATD/R. / H. Guo, J.E. Odegard, M. Lang, R.A. Gopinath, I.W. Selesnick, C.S. Burrus // First International Conference on Image Processing, – Nov. 1994. – Vol. 1. – P. 75–79.

55. **Haykin, S.** Neural Networks – a Comprehensive Foundation (2nd Edition). / S. Haykin. – India.: Prentice Hall, 2005. – 823 p.

56. **Hu, M.-K.** Visual pattern recognition by moment invariants / M.-K. Hu // IRE Transactions on Information Theory. – 1962. – P. 179–187.

57. **Jansen, M.** Geometrical Priors for Noise-free Wavelet Coefficient

Configurations in Image Denoising in Bayesian inference in wavelet based models. / M. Jansen, A. Bultheel. / Editors P. Muller, B. Vidakovic – Springer Verlag, 1999. – P. 223–242.

58. **Jiang, Q.** Principal Component Analysis and Neural Network Based Face Recognition [Электронные ресурсы] / Режим доступа: http://bit.csc.lsu.edu/~jianhua/zhifeng-yun.pdf. Дата обращения: 07.10.2013.

59. **Kakarwal, S.** Wavelet Transform based Feature Extraction for Face Recognition / S. Kakarwal, R. Dsehmukh // Informatica, 2004. – Vol. 15. – №. 2. – P. 243–250.

60. **Kan, C.** Invariant character recognition with Zernike and orthogonal Fourier-Mellin moments. / C. Kan, M.D. Srinath // Pattern Recognition, – 2000. – Vol. 35. – P. 143–154.

61. **Katsavounidis, I.** Image compression with embedded wavelet coding via vector quantization. / I. Katsavounidis, C.J. Kuo // In SPIE Conference on Mathematical Imaging. – San Deigo, California, 1995. – P. 333–344.

62. **Khotanzad, A.** Invariant image recognition by Zernike moments. / A. Khotanzad, Y.H. Hong // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1990. – Vol. 12. – № 5. – P. 489–497.

63. **Kim, B.-G.** Fast image segmentation based on multi-resolution analysis and wavelets. / B.-G. Kim, J.-I. Shim, D.-J. Park // Pattern Recognition Letters/ – 2003. – Vol. 24. – P. 2995–3006.

64. **Kim, K.** Face Recognition using Principle Component Analysis. / K. Kim // International Conference on Computer Vision and Pattern Recognition, – 1996. – P. 586–591.

65. **Kimura, F.** Modified quadratic discriminant functions and the application to Chinese character recognition / F. Kimura, K. Takashina, S. Tsuruoka, Y. Miyake // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1987. – Vol. 9. – № 1. – P. 149–153.

66. **Kirk, R.** C# and Java: Comparing Programming Languages [Электронные ресурсы] / Режим доступа: http://msdn.microsoft.com/en-

us/library/ms836794.aspx. Дата обращения: 07.10.2011.

67. **Kumar, S.A.** Wavelet based Multi Class image classification using Neural Network / S.A. Kumar, S.Tiwari, V.P. Shukla // International Journal of Computer Applications. – 2012. – Vol. 37. – № 4. – P. 21–25.

68. **Lai, J.H.** Face recognition using holistic Fourier invariant features. / J.H. Lai, P.C. Yuen, G.C. Feng // Pattern Recognition, – 2001. – Vol. 34. P. 95–109.

69. **LeCun, Y.** Convolutional Networks and Applications in Vision / Y. LeCun, K. Kavukcuoglu, C. Farabet // International Symposium on Circuits and Systems (ISCAS'10). – Paris: IEEE, 2010. – P. 253–256.

70. **LeCun, Y.** Gradient-based learning applied to document recognition. / Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. // Proceedings of the IEEE, – November 1998. – Vol. 86. – № 11. – P. 2278–2324.

71. **Lienhart, R.** Comparison of Automatic Shot Boundary Detection Algorithms / R. Lienhart // Proceedings SPIE Conference on Storage and Retrieval for Image and Video Databases. – San Jose, 1999. – Vol. 3656. – P. 209–301.

72. **Liu, C.-L.** Handwritten digit recognition: bench-marking of state-of-the-art techniques. / C.-L. Liu, K. Nakashima, H. Sako, H. Fujisawa // Pattern Recognition. – 2003. – Vol. 36. – № 10. – P. 2271–2285.

73. **Low, Y.** Wavelet based Medical Image Compression using EZW. / Y. Low, R. Besar // Proceedings of 4th National Conference on Telecommunication Technology, – Shah Alam, Malaysia, 2005. – P. 203–206.

74. **Luettin, J.** Evaluation Protocol for the Extended M2VTSDatabase (XM2VTSDB). / J. Luettin, G. Maitre // IDIAP-COM 05, IDIAP, 1998.

75. **Mahmood, R.K.** Wavelets and Support Vector Machines for Texture Classification [Электронные ресурсы] / R.K. Mahamood, R.N. Mahmood // Режим доступа: http://eprints.dcs.warwick.ac.uk/494/1/inmic04.pdf. Дата обращения: 06.05.2011.

76. **Malfait, M.** Wavelet based image denoising using a Markov Random Field a priori model. / M. Malfait, D. Roose // IEEE Transactions on Image

Processing, – 1997. – Vol. 6. – № 4. – P. 549–565.

77. **Mallat, S.** A theory for multiresolution signal decomposition; the wavelet representation. / S. Mallat // IEEE transaction on PAMI, – July 1989. – Vol. 11. – № 7. – P. 674–693.

78. **Mallat, S.** A wavelet Tour of Signal Processing. / Mallat, S. / Academy Press, New York, 1999.

79. **Mallat, S.** Multifrequency channel decompositions of images and wavelet models. / S. Mallat // IEEE Transaction on Acoust., Speech, Signal Processing, – Dec. 1989. – Vol. 37. – P. 2091–2110.

80. **Maltoni, D.** Handbook of fingerprint recognition / D. Maltoni, D. Maio, A.K. Jain, S. Prabhakar – N.Y.: Springer, 2003. – 348 p.

81. **Mazloom M,** Face Recognition using PCA, Wavelets, and Neural Networks / M. Mazloom, S. Kasaei // Proceeding of the First International Conference on Modeling, Simulation and Applied Optimization – Sharjah, U.A.E., February 1–3, 2005. – P. 1–6.

82. **Mehdi, L.** Combining Wavelet Transforms and Neural Networks for Image Classification / M. Lotfi, A. Solimani, A. Dargazany, H. Afzal, M. Bandarabadi // 41st Southeastern Symposium on System Theory, – Tullahoma, TN, USA. 2009. – P. 44–48.

83. **Moon, H.** Analysis of PCA-based Face Recognition Algorithms, Empirical Evaluation Techniques in Computer Vision. / H. Moon, P.J. Phillips/ Editors K.J. Bowyer, P.J. Phillips – IEEE CS, 1998.

84. **Morlet, J.** Wave propagation and sampling theory 1, complex signal and scattering in multilayered media. / J. Morlet, G. Arens, E. Fourgeau, D. Giard // Geophysics, – 1982. – Vol. 47 – P. 203–221.

85. **Ning, L.I.** An Implementation of OCR System Based on Skeleton Matching [Электронные ресурсы] / L.I. Ning – Режим доступа: http://kar.kent.ac.uk/21129/1/OCRNing.pdf. Дата обращения: 01.12.2013.

86. **Noreen, N.** MRI Segmentation through Wavelets and Fuzzy C-Means. / N. Noreen, K. Hayat, S.A. Madani // World Applied Sciences Journal 13

(Special Issue of Applied Math), – 2011. – P. 34–39.

87. **Nowak, R.D.** Wavelet-based Rician noise removal for magnetic imaging. / R.D. Nowak // IEEE Transaction on Image Processing, – Oct 1999. – Vol. 8. – № 10. – P. 1408–1419.

88. **Odegard, J.E.** Wavelet based SAR speckle reduction and image compression / J.E. Odegard, H. Guo, M. Lang, C.S. Burrus, R.O. Wells // In Proc. SPIE Symposium on OE/Aerospace Sensing and Dual Use Photonics, – Orlando, Florida, April 1995. – P. 17–21.

89. **Park, S.B.** Content-based image classification using a neural network / S.B. Park, J.W. Lee, S.K. Kim // Pattern Recognition Letters. – 2004. – P. 287–300.

90. **Pasnur, M.A.** Image Retrieval Using Modified Haar Wavelet Transform and K Means Clustering / M.A. Pasnur, P. S. Malge // International Journal of Emerging Technology and Advanced Engineering. – 2013. – Vol. 3. – Issue 3. – P. 89–93.

91. **Phillips, P.** The FERET Evaluation Methodology for Face-Recognition Algorithms. / P. Phillips, H. Moon, S.Y. Rizvi, P.J. Rauss // Tech. Rep. NI-STIR 6264, – 1998.

92. **Pizurica, A.** A wavelet-based image denoising technique using spatial priors. / A. Pizurica, W. Philips, I. Lemahieu, M. Acheroy // In Proceedings IEEE International Conference on Image Processing ICIP, – Vancouver, BC, Canada, 2000. – P. 296–299.

93. **Pizurica, A.** Image de-noising in the wavelet domain using prior spatial constraints. / A. Pizurica, W. Philips, I. Lemahieu, M. Acheroy // Proceedings IEE Conference on Image Processing and its Applications IPA, – Manchester, UK, 1999. – P. 216–219.

94. **Pizurica, A.** The application of Markov random field models to wavelet-based image denoising. / A. Pizurica, W. Philips, I. Lemahieu, M. Acheroy // In Imaging and Vision Systems: Theory, Assessment and Applications. / Editors J. Blanc-Talon, D. Popescu, – NOVA Science Books, Huntington, USA,

2001.

95. **Pratt, W.K.** Digital Image Processing / W.K. Pratt – N.Y.: Wiley Interscience, 2001. – 738 p.

96. **Ramana Reddy, B.V.** Texture Classification Method using Wavelet Transforms Based on Gaussian Markov Random Field / B.V. Ramana Reddy, M. Radhika Mani, K.V. Subbaiah // International Journal of Signal and Image Processing. – 2010. – Vol. 1. – Issue. 1. – P. 35–39.

97. **Rice, R.S.** The Fourth Annual Test of OCR Accuracy [Электронные ресурсы] / V.S. Rice, R.F. Jenkins, T.A. Nartker // – Режим доступа: http://stephenvrice.com/images/AT-1995.pdf. Дата обращения: 02.09.2012.

98. **Richard Casey, M.** A Survey of Methods and Strategies in Character Segmentation / M. Richard Casey, E. Lecolinet // IEEE Transaction on PAMI. 1996 – Vol. 18. – № 7. – P. 690–706.

99. **Srikantan, G.** Gradient-based contour encoder for character recognition. / G. Srikantan, S.W. Lam, S.N. Srihari // Pattern Recognition. – 1996. – Vol. 29. – № 7. – P. 1147–1160.

100. **Strutz, T.** Image data compression with pdf-adaptive reconstruction of wavelet coefficients. / T. Strutz, E. Muller // In SPIE Conference on Mathematical Imaging, – San Deigo, California, July 1995. – P. 747–758.

101. **Su, C.K.** Wavelet Tree Classification and Hybrid Coding for Image Compression. / C.K. Su, H.C. Hsin, S.F. Lin // IEE Proceedings on Vision, Image and Signal Processing, – 2005. – Vol. 152. – № 6. – P. 752–756.

102. **Turk, M.A.** Face recognition using Eigen faces. / M.A. Turk, A.L. Pentland // Proceedings IEEE Computer Society Conference of Computer Vision and Pattern Recognition, – 1991. – P. 586–591.

103. **Turk, M.A.** Eigenfaces for Recognition. / M.A. Turk, A.L. Pentland // Journal of Cognitive Neuroscience, – 1991. – Vol. 3. – № 1. – P. 71–86.

104. **Unser, M.** Texture classification and segmentation using wavelet frames. / M. Unser // IEEE Transaction on Image Processing. – 1995. – Vol. 4. – № 11. – P. 1549–1560.

105. **Vaktin, M.** The system of handwritten characters recognition on the basis of Legendre moments and neural network [Электронные ресурсы] / M. Vaktin, M. Selinger – Режим доступа: http://www.iie.uz.zgora.pl/iie_archiwum/desdes01/files/ref/V-5.pdf. Дата обращения: 01.12.2013.

106. **Venkatrama Phani Kumar S.** Face Recognition Using Wavelet Based Kernel Locally Discriminating Projection / S. Venkatrama Phani Kumar, K.V.K. Kishore, K. Hemantha Kumar // International Journal of Computer Theory and Engineering. – August, 2010. – Vol. 2. – No. 4. – P. 1793–8201.

107. **Wadkar, P.D.** Face Recognition using Discrete Wavelet Transforms / P.D. Wadkar, M. Wankhade // International Journal of Advanced Engineering Technology. – 2012. – Vol. III. – Issue I. – P. 239–242.

108. **Wang, J.Z.** Content-based image indexing and searching using Daubechies' wavelets / J.Z. Wang, G. Wiederhold, O. Firschein, S.X. Wei // International Journal on Digital Libraries. – 1997. – P. 311–328.

109. **Weibao, Z.** Image Classification Using Wavelet Coefficients in Low-pass Bands / Z. Weibao, Y. Li // Proceedings of International Joint Conference on Neural Networks, Orlando, Florida, USA. – 2007. – P. 114–118.

110. **Wunsch, P.** Wavelet descriptors for multiresolution recognition of handprinted characters. / P. Wunsch, A.F. Laine // Pattern Recognition, – August 1995. – Vol. 28. – Issue. 8. – P. 1237–1249.

111. **Yamashita, Y.** Classification of handprinted Kanji characters by the structured segment matching method. / Y. Yamashita, K. Higuchi, Y. Yamada, Y. Haga // Pattern Recognition Letters, – 1983. – Vol. 1. – P. 475–479.

112. **Yambor, W.S.** Analyzing pca-based face recognition algorithms: Eigenvector selection and distance measures. / W.S. Yambor, B.A. Draper, J.R. Beveridge // In Empirical Evaluation in Computer Vision, – July 2000.

113. **Yuka, H.** Image Classification by Lifting Wavelet PCA [Электронные ресурсы]. / H. Yuka, S. Takano, K. Niijima – Режим доступа: http://www.i.kyushu-u.ac.jp/doitr/trcs228.pdf. Дата обращения: 01.09.2012.

114. **Zhang, B.-L.** Face Recognition by Applying Wavelet Subband Representation and Kernel Associative Memory / B.-L. Zhang, H. Zhang, S.S. Ge // IEEE Transactions on Neural Networks. – January, 2004. – Vol. 15. – No. 1. – P. 166–177.

115. **Zhang, D.** A comparative study on shape retrieval using Fourier descriptors with different shape signatures / D. Zhang, G. Lu // In Proceedings of the IEEE International Conference on Multimedia and Expo. – Tokyo, Japan, 2001. – P. 1139–1142.

116. **Zhang, J.** A Medical Image Segmentation Method based on SOM and Wavelet Transform. / J. Zhang, Q. Liu, Z. Chen // Journal of Communication and Computer, – May 2005. – Vol. 2. – №. 5.

117. **Zhujie, Y.L.Y.** Face recognition with Eigen faces. / Y.L.Y. Zhujie // Proceedings IEEE International Conference Industrial Technology, – 1994. – P. 434–438.

ПЕРЕЧЕНЬ ПУБЛИКАЦИЙ ПО ТЕМЕ ДИССЕРТАЦИИ

Статьи в журналах, которые включены в перечень российских рецензируемых научных журналов и изданий для опубликования основных научных результатов диссертаций:

1. **Фан, Н.Х.** Анализ алгоритмов обнаружения импульсного шума на цифровых изображениях / Н.Х. Фан, В.Г. Спицын // Известия Томского политехнического университета. – Томск: ТПУ, 2011. – Т. 318. – № 5. – С. 70–73.

2. **Буй, Т.Т.Ч.** Алгоритмическое и программное обеспечение для классификации цифровых изображений с помощью вейвлет-пребразования Хаара и нейронных сетей / Т.Т.Ч. Буй, Н.Х. Фан, В.Г. Спицын // Известия Томского политехнического университета. – Томск: ТПУ, 2011. – Т. 319. – № 5. – С. 103–106.

3. **Фан, Н.Х.** Алгоритмы для классификации отпечатков пальцев на основе применения фильтра Габора, вейвлет-преобразования и многослойной нейронной сети / Н.Х. Фан, В.Г. Спицын // Известия Томского политехнического университета. – Томск: ТПУ, 2012. – Т. 320. – № 5. – С. 60–64.

4. **Буй, Т.Т.Ч.** Распознавание лиц на основе применения метода Виолы–Джонса, вейвлет-преобразования и метода главных компонент / Т.Т.Ч. Буй, Н.Х. Фан, В.Г. Спицын // Известия Томского политехнического университета. – Томск: ТПУ, 2012. – Т. 320. – № 5. – С. 54–59.

5. **Буй, Т.Т.Ч.** Распознавание лиц и жестов на основе применения вейвлет-преобразования и метода главных компонент / Т.Т.Ч. Буй, Н.Х. Фан, В.Г. Спицын // Нелинейный мир. – Москва: Радиотехника, 2012. – Т. 10 – № 6 – С. 371–379.

6. **Фан, Н.Х.** Распознавание печатных текстов на основе применения вейвлет-преобразования и метода главных компонент / Н.Х. Фан, Т.Т.Ч. Буй, В.Г. Спицын // Известия Томского политехнического

университета. – Томск: ТПУ, 2012. – Т. 321. – № 5. – С. 154–158.

7. **Фан, Н.Х.** Распознавание жестов на видеопоследовательности в режиме реального времени на основе применения метода Виолы-Джонса, алгоритма CAMShift, вейвлет-преобразования и метода главных компонент. / Н.Х. Фан, Т.Т.Ч. Буй, В.Г. Спицын // Вестник Томского государственного университета. Управление, вычислительная техника и информатика – Томск: ТГУ, 2013. – Т. 23. – № 2. – С. 102–111.

Публикации в других научных изданиях:

8. **Фан, Н.Х.** Удаление шумов на изображениях на основе применения искусственных нейронных сетей / Н.Х. Фан, В.Г. Спицын // Молодежь и современные информационные технологии: сборник трудов VIII Всероссийской научно-практической конференции студентов, аспирантов и молодых ученых: в 2 ч. – Томск, 3–5 марта 2010. – Томск: СПБ Графикс, 2010. – Ч. 2. – С. 227–228.

9. **Фан, Н.Х.** Методы удаления шумов на изображениях на основе применения искусственных нейронных сетей / Н.Х. Фан, В.Г. Спицын // Современные техника и технологии: Сборник трудов XVI Международной научно-практической конференции студентов, аспирантов и молодых ученых: в 3 т. – Томск, 12–16апреля 2010. – Томск: ТПУ, 2010. – Т.2.– С. 399–401.

10. **Фан, Н.Х.** Снижение шумов на цифровых изображениях / Н.Х. Фан, В.Г. Спицын // Научная инициатива иностранных студентов и аспирантов российских вузов: Сборник докладов III Всероссийской научно-практической конференции. – Томск, 19–21 мая 2010. – Томск: ТПУ, 2010. – С. 190–194.

11. **Фан, Н.Х**. Модифицированный медианный фильтр подавления импульсного шума на изображениях / Н.Х. Фан, В.Г. Спицын // Электронные средства и системы управления: Материалы докладов VI Международной научно-практической конференции: в 2 ч. – Томск, 13–16 октября 2010. – Томск: В-Спектр, 2011. – Ч. 1. – С. 118–121.

12. **Буй, Т.Т.Ч**. Подавление шумов и реконструкция изображений на основе применения ядра регрессии / Т.Т.Ч. Буй, Н.Х. Фан // Современные техника и технологии: Сборник трудов XVII Международной научно-практической конференции студентов, аспирантов и молодых ученых: в 3 т. – Томск, 18–22 апреля 2011. – Томск: ТПУ, 2011. – Т.2.– С. 299–300.

13. **Фан, Н.Х.** Анализ алгоритмов обнаружения импульсного шума на изображениях / Н.Х. Фан, В.Г. Спицын // Молодежь и современные информационные технологии: Сборник трудов IX Всероссийской научно-практической конференции студентов, аспирантов и молодых ученых: в 2 ч. – Томск, 11–13 мая 2011. – Томск: СПБ Графикс, 2011. – Ч. 1. – С. 126–127.

14. **Фан, Н.Х.** Анализ алгоритмов обнаружения импульсного шума на изображениях / Н.Х. Фан, В.Г. Спицын // Проблемы информатики. Новосибирск: НГТУ, 2011. – №2(10). – С. 26–30.

15. **Буй, Т.Т.Ч.** Классификация изображений на основе применения цветовой информации, вейвлет-преобразования Хаара и многослойной нейронной сети / Т.Т.Ч. Буй, Н.Х. Фан, В.Г. Спицын // Проблемы информатики. – Новосибирск: НГТУ, 2011. – Спецвыпуск. С. 81–86.

16. **Буй, Т.Т.Ч.** Способ классификации изображений на основе применения вейвлет-преобразования Хаара и нейронных сетей / Т.Т.Ч. Буй, Н.Х. Фан, В.Г. Спицын // Нейроинформатика, ее приложения и анализ данных: материалы XIX Всероссийского семинара. – Красноярск, 1–3 октября 2011. – Красноярск: СФУ, 2011. – С. 159–164.

17. **Bui, T.T.T.** Face and Hand Gesture Recognition based on Wavelet Transforms and Principal Component Analysis / T.T.T. Bui, N.H. Phan, V.G. Spitsyn // 7th International Forum on Strategic Technology IFOST: Proceedings of IFOST. – Tomsk: TPU Press, 2012. – V. 1. – P. 588–591.

ПРИЛОЖЕНИЕ

УТВЕРЖДАЮ
Проректор по образовательной и
международной деятельности ТПУ
_____ А.А. Чучалин
«__» 24.03. 2014 г.

АКТ ВНЕДРЕНИЯ

результатов кандидатской диссертационной работы

Фан Нгок Хоанг

Данный документ удостоверяет успешное применение в учебном процессе Института кибернетики ТПУ результатов кандидатской диссертационной работы Фан Нгок Хоанг.

Начиная с 2012 г. студенты гр. 8ВМ11 и 8ВМ2А, ИК ТПУ, обучающиеся по направлению 230100 «Информатика и вычислительная техника» магистерской программы «Компьютерный анализ и интерпретация данных», при изучении спецкурса «Методы интеллектуальной обработки и анализа изображений» выполняют подготовленные Фан Нгок Хоанг лабораторные работы. Лабораторные работы содержат теоретическую часть, описывающую применяемые для обработки изображений методы и алгоритмы, а также задания для выполнения лабораторных работ. В лабораторных работах проводится изучение цветовых моделей и преобразований цветов при переходах между моделями, рассматривается применение вейвлет-пребразований Хаара и Добеши для разложения цифровых изображений, использованных при подготовке материалов диссертации.

Изданы соответствующие методические указания по проведению лабораторных работ: Ю.А. Болотова, Т.Т.Ч. Буй, К.А. Кермани, В.Г. Спицын, Н.Х. Фан. Алгоритмы интеллектуальной обработки цифровых изображений. Часть 1: Методические указания к лабораторным работам по курсу «Методы интеллектуальной обработки и анализа изображений». Томск: Издательство ТПУ, 2012. – 58 с.

Директор ИК ТПУ,
д.т.н.
_____ А.А. Захарова
«24» 03 2014 г.

Зав. кафедрой ВТ,
профессор, д.т.н.
_____ Н.Г. Марков
«24» 03 2014 г.