Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Панкратов Антон Михайлович

Распознавание входящих ключевых слов в оцифрованных изображениях рукописных текстов

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
к.ф.-м.н., доцент
С.И. Гуров

Москва, 2016

Аннотация

В данной работе рассматривается задача поиска ключевых слов в неразмеченном корпусе отсканированных изображений рукописных документов. Производится обзор существующих методов по распознаванию рукописных текстов. Также предлагается реализация решения задачи распознавания нумерации документов как первого шага для создания системы навигации.

Содержание

1 Введение

В настоящее время очень много информации хранится в электронном виде. Для работы с большими объемами информации часто удобно использовать системы навигации, способные проводить поиск файлов, релевантных пользовательскому запросу, который обычно задается строкой на естественном языке. Для электронных документов (тесктов) задача поиска хорошо исследована [1], в то время как для изображений и медиафайлов проблема остается открытой.

В частности, для навигации по отсканированным документам нужно выделить на изображении текст и, по возможности, распознать его. Таким образом, одной из задач компьютерного зрения является задача поиска и распознавания текста на изображении. Существует несколько направлений данной задачи:

- оптическое распознавание символов (OCR, optical character recognition) - задача представления изображения отсканированного текста, как рукописного, так и печатного, в виде текстовых данных;
- распознавание рукописного ввода (HWR, handwriting movement analysis) - задача распознавания текста параллельно с его написанием.

Задача OCR в настоящее время решена в случае печатного текста [2], существуют как открытые (Tesseract), так и проприетарные (ABBYY Finereader) программы, позволяющие получить результат с достаточно высокой точностью. При этом качество работы программ зависит от распространенности шрифта и качества изображения.

Задача распознавания рукописного текста на данный момент решена частично. Высокая точность распознавания получена, например, для отдельных цифр [5] и банковских бланков [4]. Из основных проблем, возникающих при распознавании отсканированного рукописного текста, можно выделить

- низкое качество исходных изображений;
- сцепленность букв внутри одного слова;
- высокая вариативность написания букв;
- возможные орфографические ошибки.

Задача распознавания рукописного ввода подразумевает использование специальных устройств, как, например, стилус или сенсорный экран. С их помощью собирается дополнительная информация о вводе (координаты, скорость написания текста, сила нажима), что позволяет сильно повысить точность распознавания. В настоящее время существуют алгоритмы, позволяющие распознавать рукописный ввод с высоким качеством [3], что позволяет использовать их в потребительской технике, например, в смартфонах.

Глобальной задачей является реализация системы навигации, позволяющей искать отсканированные документы, на которых содержатся заданные пользователем слова, словосочетания или числа. Целью данной работы является изучение существующих подходов оффлайн распознавания рукописного текста и реализация алгоритма по поиску и распознаванию номера страницы.

2 Современные подходы к оффлайн-распознаванию

Можно выделить две задачи оффлайн-распознавания рукописного текста: распознавание отдельных символов и распознавание целых слов. При этом общая схема [7] остается следующей:

1. Предобработка изображения - выделение интересующих нас объектов на изображении (например, отдельных символов, слов или целых строк) и их нормализация, то есть приведение к некоторому общему виду (например, все символы приводятся к одному размеру, все строки выравниваются по ширине). Обычно для качественного выделения строк изображение заранее стараются избавить от шума. Некоторые алгоритмы сегментации строк также требуют предварительной бинаризации изображения, то есть разделение пикселей, отвечающих тексту и пикселей фона.

2. Извлечение признаков из полученных объектов - каждому объекту ставится в соответствие набор признаков некоторой длины по заранее заданному алгоритму.

3. Распознавание объектов - каждому полученному признаковому описанию ставится в соответствие символ или последовательность символов.

2.1 Предобработка изображения

В [8] описаны стандартные фильтры, используемые для уменьшения зашумленности изображения: фильтр Гаусса для борьбы с шумом и квантильные фильтры для борьбы с темными / светлыми выбросами (называемые «соль и перец»).

Под бинаризацией понимается сопоставление каждому пикселю исходного изображения 1, если данный пиксель принадлежит объекту (то есть, тексту) и 1 в противном случае.

Цель сегментации строк - получить набор прямоугольных изображений (не обязательно одного размера), на каждом из которых присутствует ровно одна строка из исходного изображения. Строки, расположенные не горизонтально, обычно предварительно поворачиваются на необходимый угол. Для выделения строк на изображении существует несколько подходов [6]:

- На основе гистограмм - в предположении, что строки расположены горизонтально и несильно перекрываются между собой, суммируются яркости пикселей, находящихся на одном уровне по горизонтали, по полученным значениям строится вертикальная гистограмма. При этом локальные минимумы на гистограмме будут соответствовать границам

между строками. Данный метод также может быть применен для разделения строки на отдельные слова, если строить гистограмму не по горизонталям, а по вертикалям.

- На основе преобразования Хафа [9]. Преобразование Хафа в общем случае позволяет обнаруживать произвольные кривые, задаваемые параметрически, на бинарных изображениях. Для этого множество значений каждого параметра кривой дискретизируется, после чего для каждого набора параметров подсчитывается число пикселей изображения, удовлетворяющих заданной кривой. Впоследствии кривые, набравшие наибольшее число пикселей, рассматриваются как кандидаты.

  Одним из преимуществ использования преобразования Хафа для сегментации строк является обнаружение строк вне зависимости от их наклона.

2.2 Извлечение признаков

На данном этапе каждому изображению, содержащему выделенный объект (слово, строку или символ) ставится в соответствие описание из вектора признаков.

Одним из стандартных способов генерации признаков по изображениям является метод скользящего окна [7]. Изображение разбивается все возможные прямоугольники фиксированного размера, после чего для каждого из них подсчитывается некоторый агрегированный признак (например, средняя яркость внутри прямоугольника). Признаковым описанием исходного объекта является конкатенация признаковых описаний всех прямоугольников.

2.3 Распознавание

При решении задачи распознавания каждому признаковому описанию ставится в соответствие последовательность символов. При этом алгоритм распознавания должен иметь возможность работать с последовательностями различной длины, так как в общем случае в строках допустимо различное число символов. В настоящее время для распознавания рукописных слов и строк применяются две модели: скрытые марковские цепи [10] и рекуррентные нейронные сети [12].

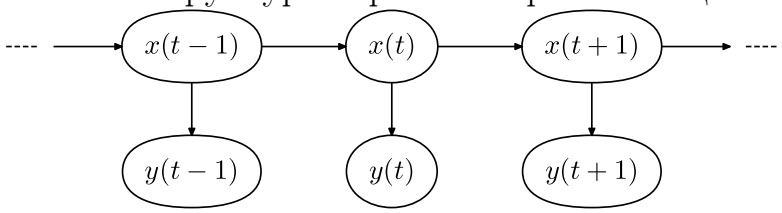
2.3.1 Скрытые марковские цепи

Скрытые марковские цепи позволяют описать последовательность как результат двухэтапного стохастического процесса [10]:

1. генерируется последовательность дискретных величин $t(i)$, отвечающая за скрытые состояния. При этом каждое следующая величина вероятностно зависит только от предыдущего значения,

2. для каждого скрытого состояния $t(i)$ генерируется зависящее от него наблюдаемое состояние $x(i)$.

Рис. 1: Структура скрытой марковской цепи



Вероятностное распределение на скрытые и наблюдаемые переменные имеет следующий вид:

$$P(X,T) = p(t_1)\prod_{i=2}^{N} p(t_i|t_{i-1})\prod_{j=1}^{N} p(x_j|t_j)$$

Такая запись позволяет ставить задачу о поиске наиболее вероятного набора скрытых состояний при известной последовательности наблюдаемых переменных:

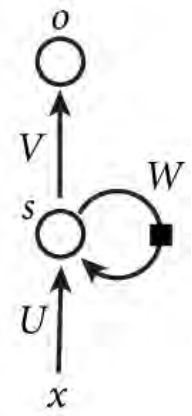
$$T^* = argmax P(T|X)$$

Для задачи распознавания строк (или слов) за скрытые переменные принято брать последовательность символов, записанную в строке, а за наблюдаемые переменные - признаковое описание строки.

2.3.2 Рекуррентные нейронные сети

В [12] описан дискриминативный подход распознавания строк, основанный на RNN (recurrent neural network). Структура простейшей рекуррентной сети имеет следующий вид

Рис. 2: Структура RNN



Здесь x - элемент входной последовательности, s - скрытое состояние, o - элемент выходной последовательности. Обычно элементы последовательностей как и скрытое состояние являются векторами фиксированной длины. Для пересчета o по x используется следующая последовательность вычислений:

1. $s = f_1(Ux + Ws)$
2. $o = f_2(Vs)$

Здесь U, W, V - матрицы весов, а f_1 и f_2 - некоторые нелинейные функции. Использование скрытого состояния, обновляющегося каждым новым входом, позволяет RNN учитывать предыдущие входы при подсчете текущего выхода.

В случае задачи распознавания строк x - это признаковое описание прямоугольника, полученного методом скользящего окна. На выходе чаще всего ожидаются вероятности следующего символа. Для этого в качестве f_2 берется функция softmax [13]:

$$softmax(y)_i = \frac{\exp y_i}{\sum_{i=1}^{n} \exp y_i}$$

3 Постановка задачи

Дано изображение отсканированного рукописного текста, на котором дополнительно могут присутствовать иллюстрации. В верхнем правом углу находится номер документа - последовательность рукописных цифр. Считается, цифры написаны горизонтально, недалеко друг от друга и не сливаются.

Рис. 3: Пример исходного изображения, исходное разрешение 1089 × 1637

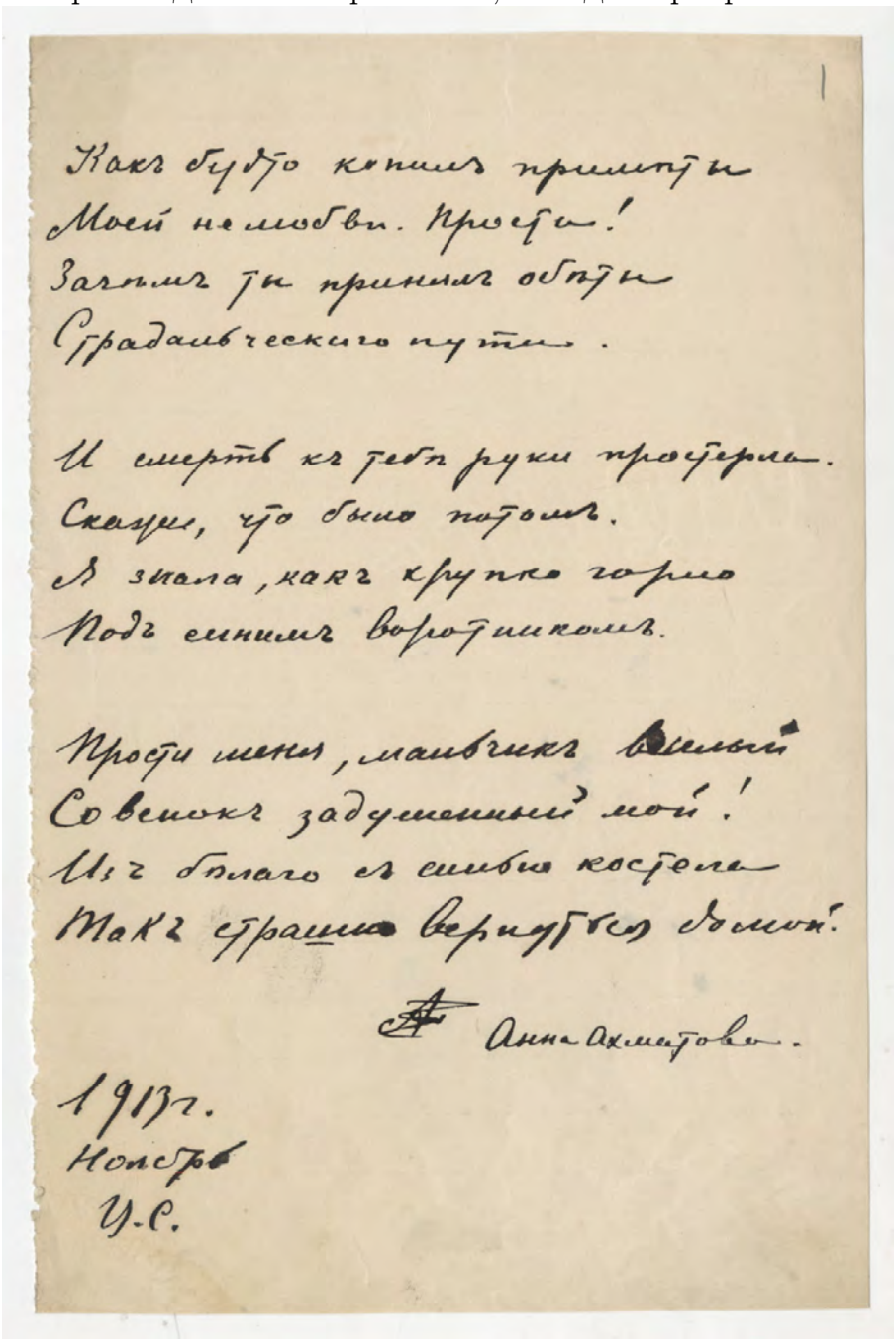
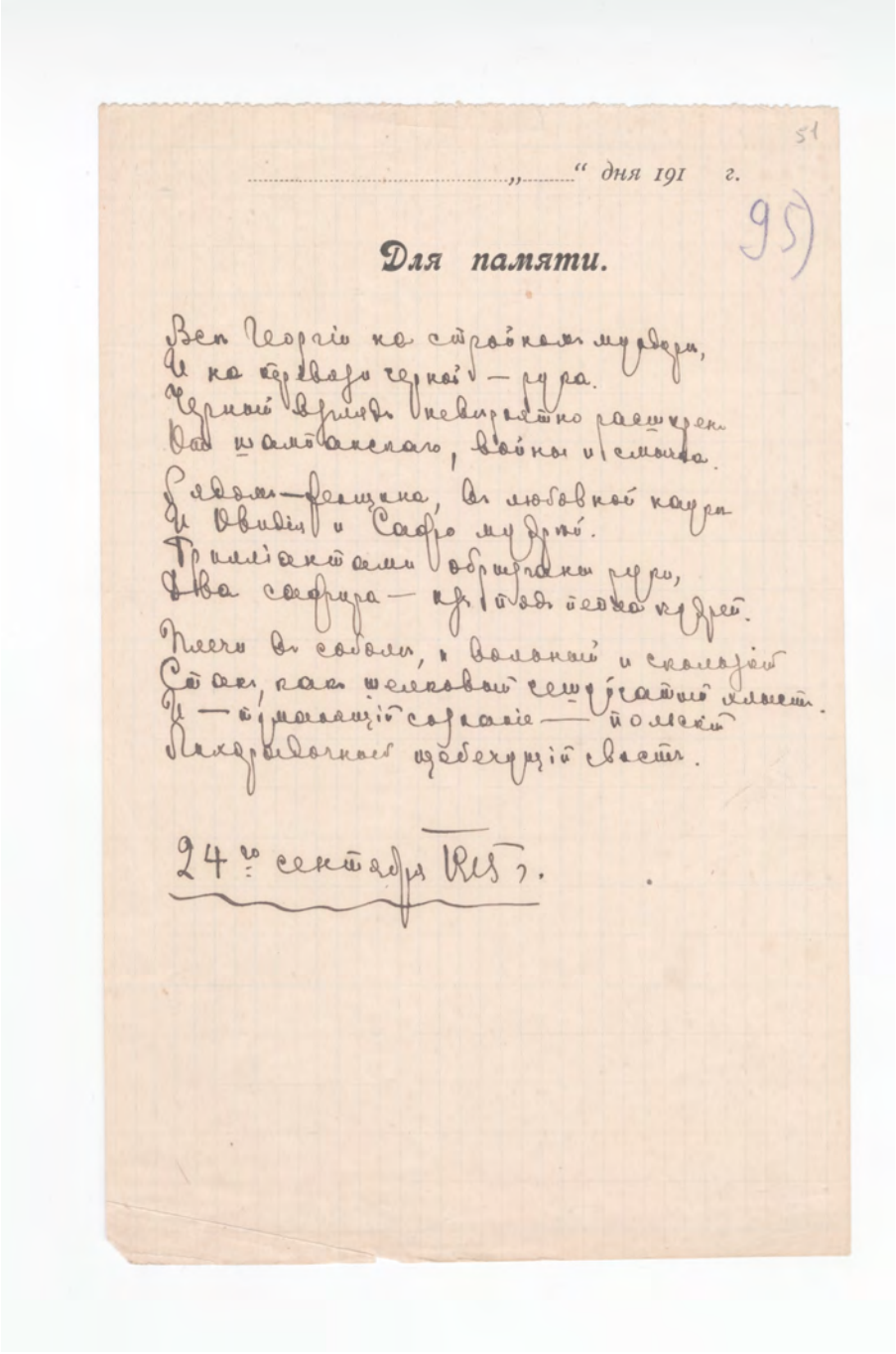


Рис. 4: Пример исходного изображения, исходное разрешение 1814 × 2746



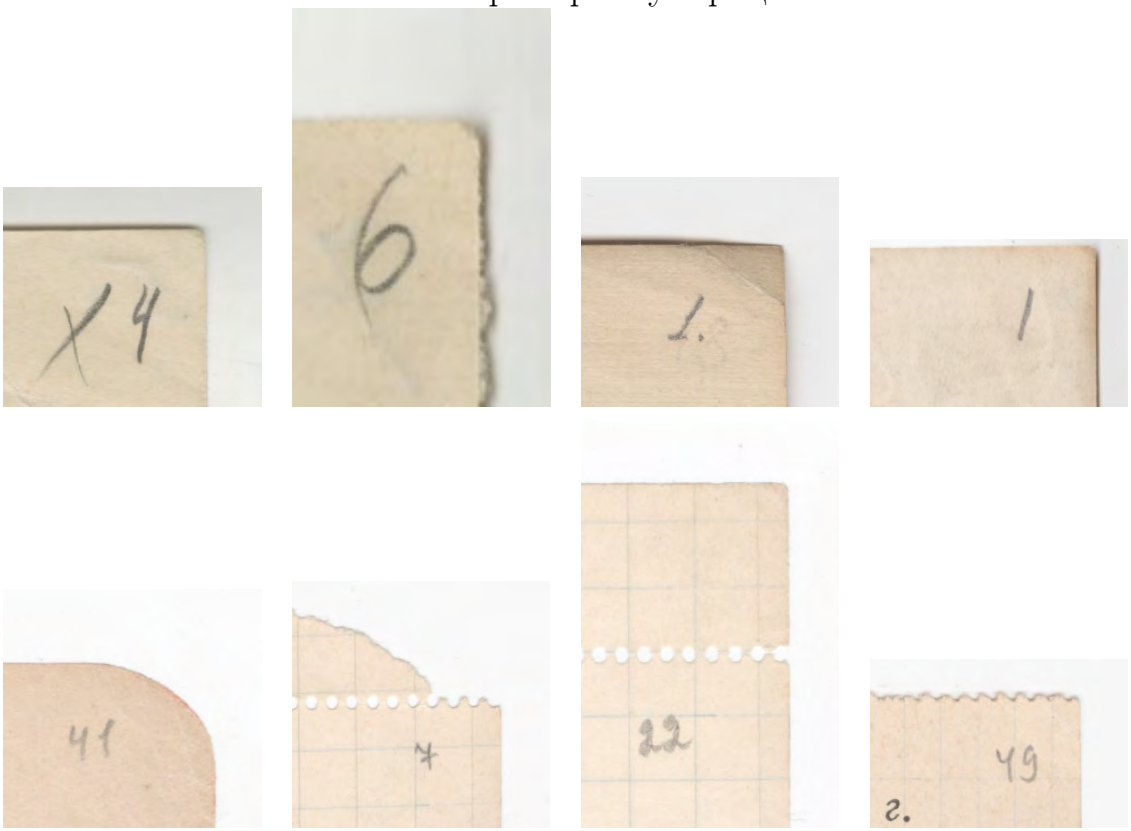
Требуется распознать номер документа и записать его в виде последовательности цифр.

4 Алгоритм распознавания нумерации

4.1 Описание алгоритма

На вход алгоритму подается изображение в оттенках серого - матрица I размера $N \times M$. Требуется распознать нумерацию, расположенную в правом верхнем углу. Пример целого изображения приведён на Рис. 3, примеры нумераций ниже:

Рис. 5: Примеры нумерации



Предложенный алгоритм можно разбить на следующие этапы:

1. Уменьшение шума на изображении - сглаживание дефектов изображении.

2. Сегментация цифр нумерации - бинаризация изображения с последующим выделением и нормализацией компонент, отвечающих цифрам нумерации.

3. Распознавания каждой компоненты нумерации.

Для сглаживания перепада яркости у границы документа и различий в шрифте между разными типами пишущих инструментов использовалось гауссовское сглаживание [8]. Его применение можно описать через операцию

свертки:

$$g_{i,j} = \sum_{k=-w}^{w} \sum_{l=-h}^{h} f_{i+k,j+l} c_{k,l}$$

где f - матрица, задающая исходное изображение, а c - матрица $(2w+1) \times (2h+1)$, задающая ядро свертки. Для получения гауссовского сглаживания коэффициенты ядра свертки берутся произведения двух одномерный гауссиан с нулевым средним и одинаковой дисперсией:

$$c_{k,l} = \frac{1}{A} \frac{1}{2\pi\sigma^2} e^{\frac{k^2+l^2}{2\sigma^2}}$$

где A - коэффициент нормировки, определяемый условием

$$\sum_{k=-w}^{w} \sum_{l=-h}^{h} c_{k,l} = 1$$

Для бинаризации изображения используется алгоритм Оцу [15]. Метод работает в предположении о бимодулярности гистограммы яркости пикселей. Данный алгоритм позволяет подобрать глобальный порог t, такой, что любой пиксель яркости больше t следует отнести к объекту. а остальные - к фону (в предположении, что фон светлее объекта). Для поиска порога минимизируется средняя внутриклассовая дисперсия:

$$\sigma^2(t) = w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t) \to \min_t$$

$$n_1(t) = \sum_{i=1}^{N} \sum_{j=1}^{M} [I_{ij} \leq t] - \text{число пикселей, отнесенных к фону}$$

$$n_2(t) = \sum_{i=1}^{N} \sum_{j=1}^{M} [I_{ij} > t] - \text{число пикселей, отнесенных к объекту}$$

$$w_1(t) = \frac{n_1(t)}{NM} - \text{доля пикселей, отнесенных к фону}$$

$$w_2(t) = \frac{n_2(t)}{NM} - \text{доля пикселей, отнесенных к объекту}$$

$$\mu_1(t) = \frac{1}{n_1(t)} \sum_{i=1}^{N} \sum_{j=1}^{M} I_{ij}[I_{ij} \leq t] - \text{средняя яркость пикселей фона}$$

$$\mu_2(t) = \frac{1}{n_2(t)} \sum_{i=1}^{N} \sum_{j=1}^{M} I_{ij}[I_{ij} > t] - \text{средняя яркость пикселей объекта}$$

$$\sigma_1^2(t) = \frac{1}{n_1(t)} \sum_{i=1}^{N} \sum_{j=1}^{M} (I_{ij}[I_{ij} \leq t] - \mu_1(t))^2 - \text{дисперсия пикселей фона}$$

$$\sigma_2^2(t) = \frac{1}{n_2(t)} \sum_{i=1}^{N} \sum_{j=1}^{M} (I_{ij}[I_{ij} > t] - \mu_2(t))^2 - \text{дисперсия пикселей объекта}$$

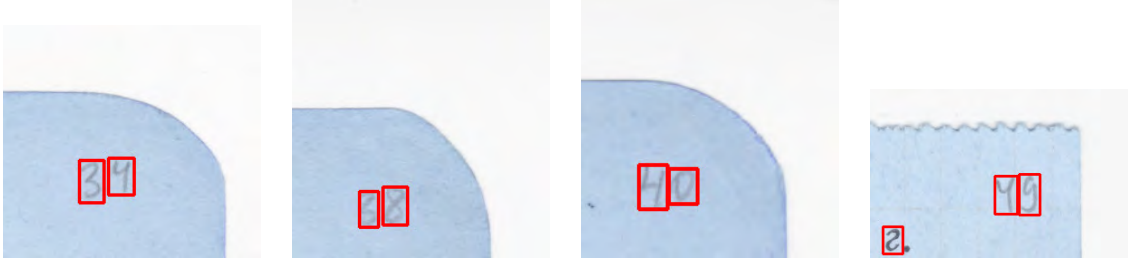
После нахождения оптимального порога t^* строится бинарное изображение по правилу

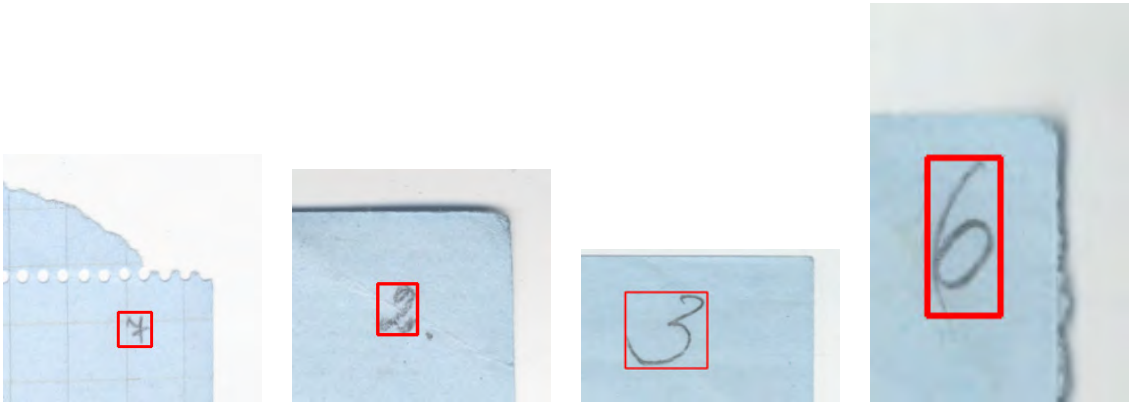
$$B_{i,j} = \begin{cases} 0, & I_{i,j} \leq t^* \\ 1, & I_{i,j} > t^* \end{cases}$$

По полученному изображению строится неориентированный граф по следующему алгоритму: каждому пикселю изображения, отнесенного к объекту ставится в соответствие вершина графа, две вершины соединяются ребром, если соответствующие пиксели имеют общую сторону или угол (8-связность). На полученном графе производится поиск компонент связности - максимальных по включению множеств вершин, связанных друг с другом - с помошью алгоритма поиска в ширину [16].

Для каждой из полученных компонент связности на исходном изображении строится ограничивающий прямоугольник, со сторонами, параллельными осям координат. В качестве координат нижней левой вершины прямоугольника берутся минимальные x и y координаты среди пикселей из компоненты связности. Аналогично, для верхней правой вершины прямоугольника берутся минимальные x и y координаты пикселей компоненты связности. После этого компоненты связности фильтруются по числу пикселей: слишком маленькие компоненты, скорее всего, относятся к остаточному шуму, а слишком большие - к изображениям и краям документа. Затем выбирается компонента связности, центр масс которой находится ближе всего к верхнему правому углу изображения. Соответствующий ей ограничивающий прямоугольник рассматривается в качестве одной из цифр нумерации.

Рис. 6: Примеры ограничивающих прямоугольников





Для распознавания символа, находящегося в найденном ограничивающем прямоугольнике предлагается использовать сверточные нейронные сети, которые превосходят большинство алгоритмов на задаче классификации рукописных цифр [11]. Любая нейронная сеть состоит из последовательности слоев, каждый слой производит некоторое преобразование над своим входом. При этом первому слою на вход подаются исходные данные, а каждый следующий слой получает на вход выход предыдущего слоя. Для построения сверточной нейронной сети чаще всего используются три типа слоев [11]:

- Сверточный слой - на вход подается изображение, выходом слоя является применение операции свертки к входному изображению. Веса ядро свертки являются параметром слоя и настраиваются отдельно. Также часто в одном сверточном слое применяется несколько операций свертки с разными ядрами. В таком случае выходом слоя будет трехмерная матрица, составленная из результатов свертки с каждым из ядер.

- Слой субдискретизации - на вход подается изображение (или несколько изображений, тогда каждое обрабатывается отдельно, а результат формируется в виде трехмерного массива), которое разбивается на сетку из прямоугольных ячеек фиксированного размера. Для каждой ячейки считается некоторая агрегирующая функция, например, в [11] предложено брать максимальное значение яркости в ячейке. Выходом слоя является матрица из значений агрегирующей функции для ячеек. Очевидно, что при применении слоя субдискретизации выходное изображение меньше исходного по размеру.

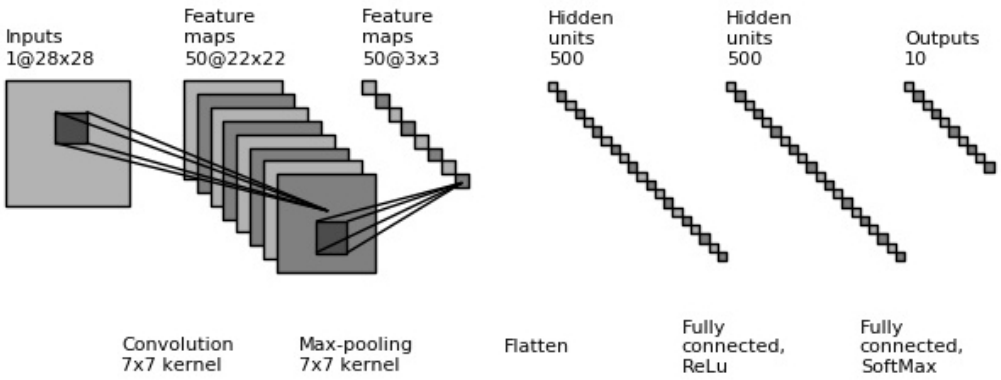
- Полносвязный слой - на вход подается вектор x, выходом слоя является вектор y, полученный по формуле $y = f(Wx)$, где W - матрица весов, а f - некоторая нелинейная функция, называемая функцией активации. В [17] приводится сравнительный анализ некоторых из них.

Для настройки весов нейронных сетей применяются алгоритмы, основанные на обратном распространении ошибок, описанные в [18].

В алгоритме использовалась четырёхслойная нейронная сеть следующей архитектуры:

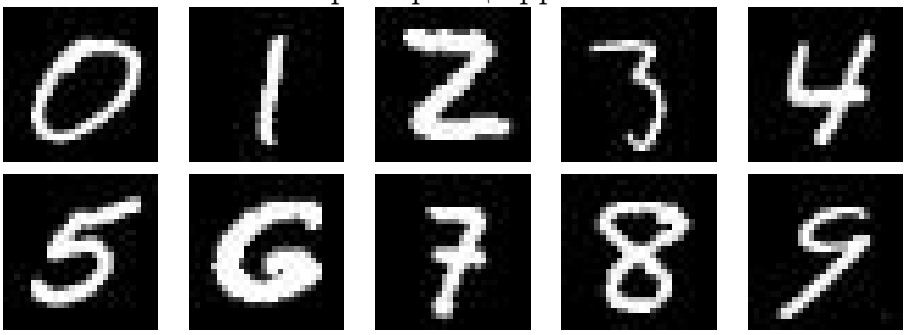
1. сверточный слой из 50 сверток с ядром 7×7
2. слой субдискретизации с ячейками размера 7×7
3. полносвязный слой с вектором из 500 элементов на выходе и функцией нелинейности ReLu: $\mathrm{ReLu}(x) = \max(x, 0)$.
4. полносвязный слой с вектором из 10 элементов на выходе и функцией нелинейности вида Softmax. Выход нейронной сети интерпретировался как вероятности цифр.

Рис. 7: Архитектура нейронной сети



Для обучения нейронной сети использовалась готовая выборка рукописных символов MNIST (`http://yann.lecun.com/exdb/mnist/`). Данная выборка содержит 70000 размеченных рукописных символов и разбита на обучающую выборку из 60000 объектов и 10000 объектов для тестирования. Каждое изображение имеет размер 28×28, каждый пиксель описан яркостью - целым числом в диапазоне от 0 до 255.

Рис. 8: Примеры цифр из MNIST



В качестве функции ошибки взята стандартная для задачи классификации многоклассовая кросс-энтропия, вычисляемая по формуле

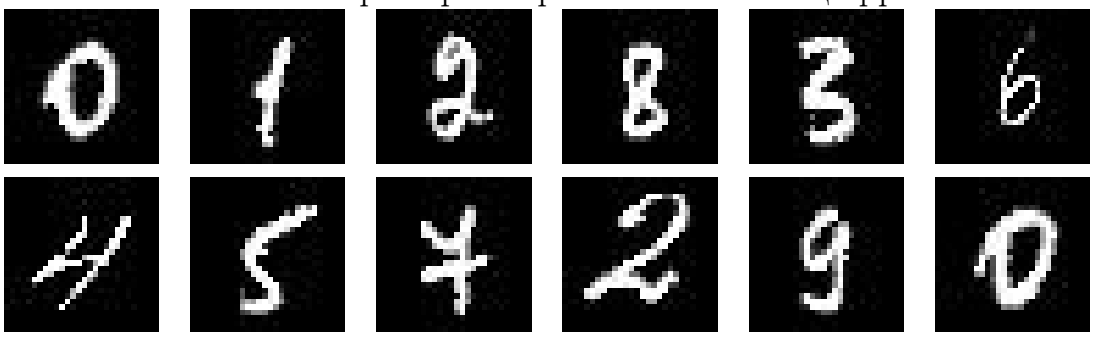
$$L_i = -\sum_{j=0}^{9} t_{i,j} \log(p_{i,j})$$

где i - номер объекта выборки; $t_{i,j}$ - бинаркая переменная, равная 1, если на объекте i изображена цифра j и 0 иначе; $p_{i,j}$ - оценка вероятности того, что на объекте i находится цифра j, полученная алгоритмом. При этом в качестве предсказанной цифры берется цифра с максимальной оценкой вероятности $p_{i,j}$. Настройка весов нейронной сети производилась с помощью стохастического алгоритма ADAM [14].

Для применения обученной нейронной сети к найденным прямоугольникам, содержащим цифры, была применена следующая нормализация:

- яркость пикселей, отвечающих фону, обнулялась;
- яркости пикселей в прямоугольнике инвертировались (так как в MNIST черный цвет соответствует фону);
- размер прямоугольника изменялся пропорциональна так, чтобы его можно было поместить в матрицу 20×20;
- полученная матрица 20×20 помещалась в центр матрицы 28×28 и дополнялась нулями;
- с помощью параллельных переносов центр масс получившегося изображения переносился в центр прямоугольника.

Рис. 9: Примеры нормализованных цифр



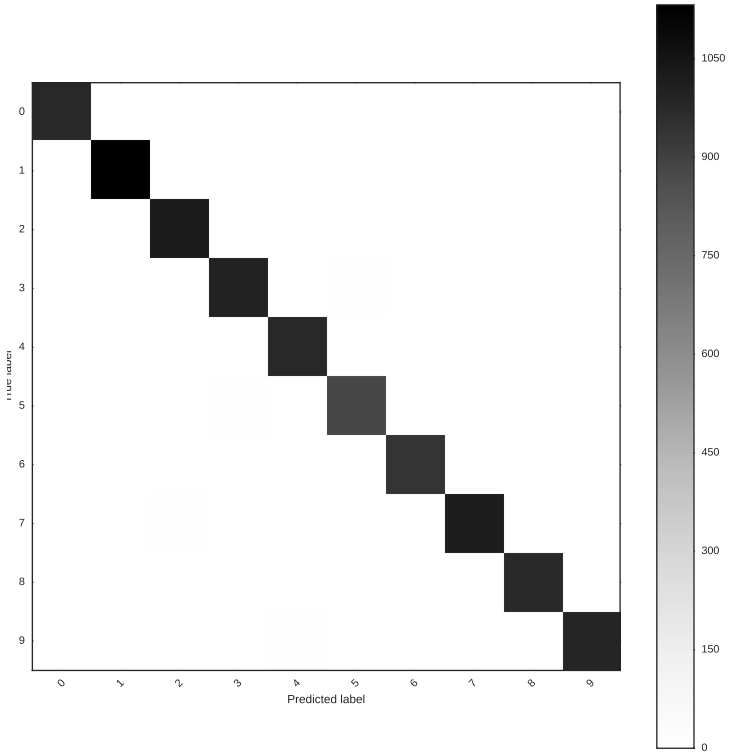
Данные преобразования необходимы, чтобы гарантировать однородность реальных данных и данных из обучения(так как объекты из MNIST были получен описанным образом).

Реализацией описанного выше алгоритма является консольная программа, которой на вход подается путь до изображения, а выводится число, соответствующее распознанной нумерации.

4.2 Тестирование

Для тестирования обученной нейронной сети использовалась тестовая выборка, включенная в поставку данных MNIST. На тестовой выборке точность алгоритма составила 99.1%. Ниже представлена матрица ошибок для тестовой выборки MNIST. На ней по оси x отложены предсказанные классы, а на оси y - истинные.

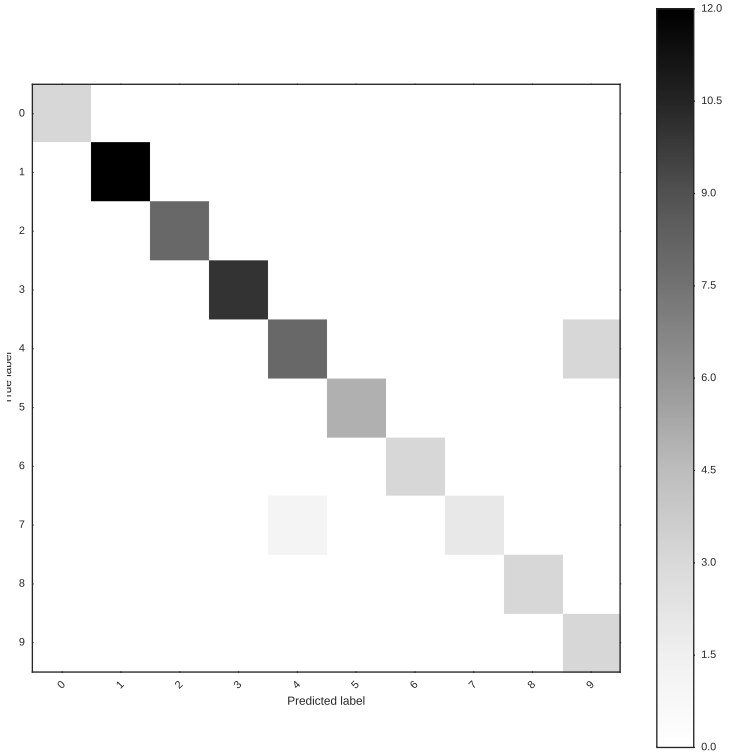
Рис. 10: Матрица ошибок классификации для тестовой выборки MNIST



Все внедиагональные ячейки не превосходят нескольких десятков, поэтому не видны по сравнению с диагональными элементами.

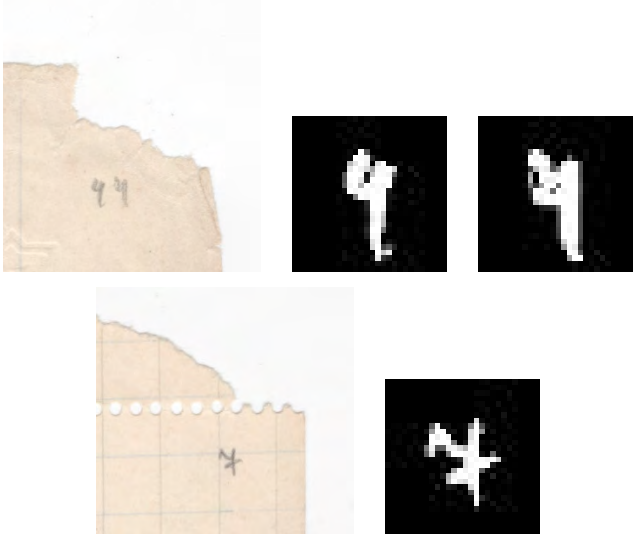
Кроме того, алгоритм тестировался на выборке исторических документов из Российского государственного архива литературы и искусства; пример такого документа представлен на Рис. 3. На выборке из 50 изображений точность алгоритма составила 86%. Для цифр из изображений, в которых нумерация была найдена правильно, также построена матрица ошибок.

Рис. 11: Матрица ошибок классификации для данных РГАЛИ



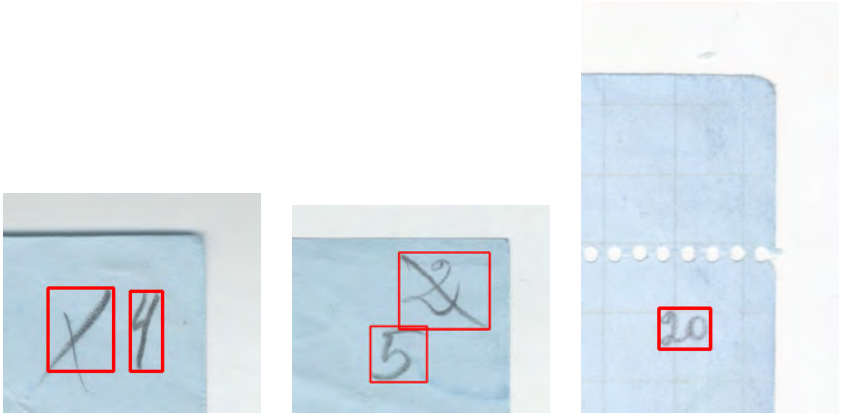
Всего данных три неверные классификации: дважды четверка была распознана как девятка; один раз семерка была распознана как четверка.

Рис. 12: Примеры неверной классификации



Кроме того, в данных встречались зачеркнутые символы и сцепленные цифры, которые не были разделены.

Рис. 13: Примеры ошибок при выделении цифр



5 Заключение

В ходе выполнения выпускной квалификационной работы были получены следующие результаты:

- изучены современные подходы к оффлайн-распознаванию рукописного текста, основанные на скрытых марковских цепях и рекуррентных нейронных сетях;
- предложен и реализовал алгоритм распознавания нумерации на основе сверточных нейронных сетей;
- произведено тестирование алгоритма на валидационной выборке из MNIST и на выборке реальных исторических документов из Российского государственного архива литературы и искусства; на валидационной выборке MNIST получена точность 99.1%, на данных РГАЛИ - 86%.

Целью дальнейших исследований является разработка алгоритма распознавания рукописных текстов и его использование для создания системы поиска по корпусу РГАЛИ.

Список литературы

[1] Manning C., Raghavan P., Schütze H. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[2] Dr. S.Vijayarani1 and Ms. A.Sakila2, *Performance comparision of OCR tools*. International Journal of UbiComp (IJU), Vol.6, No.3, July 2015.

[3] Pham Viet Dung, *Multiple Convolution Neural Networks for an Online Handwriting Recognition System*. The Sixth International Conference on Advances in System Simulation, 2014.

[4] S. Impedovo, P. Wang, H. Bunke. *Automatic Bankcheck Processing*. World Scientific, 1997.

[5] O. Matan, J. Bromley и другие. *Reading handwritten digits: A ZIP code recognition system*. Computer 25.7, 59-63, 1992.

[6] L. Likforman-Sulem, A. Zahour. *Text line segmentation of historical documents: A survey*. Int. Journal on Document Analysis and Recognition, 9(2), 123–138, 2007.

[7] A. Fischer. *Handwriting Recognition in Historical Documents*. PhD Thesis, 2012.

[8] L. Shapiro, G. Stockman. *Computer Vision*. Prentice Hall, 2001.

[9] A. Hanimyan, C. Faure. *A Hough Based Algorithm for Extracting Text Lines in Handwritten Document*. ICDAR'95, 774-777, 1995.

[10] T. Ploetz and G. A. Fink. *Markov models for offline handwriting recognition: A survey*. Int. Journal on Document Analysis and Recognition, 12(4) 269–298, 2009.

[11] Y. LeCun и другие. *Gradient-Based Learning Applied to Document Recognition*. Proceedings of the IEEE, 86(11):2278-2324, 1998

[12] A. Graves, M. Liwicki и другие. *A novel connectionist system for improved unconstrained handwriting recognition*. IEEE Trans. PAMI, 31(5) 855–868, 2009.

[13] Kevin L. Priddy, Paul E. Keller. *Artificial Neural Networks: An Introduction*. SPIE Publications, 2005.

[14] Diederik Kingma, Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 3rd International Conference for Learning Representations, 2015.

[15] N. Otsu. *A threshold selection method from gray-level histograms.* IEEE Trans. Sys., Man., Cyber. 9 62-66, 1979.

[16] T. H. Cormen и другие. *Introduction to Algorithms.* The MIT Press, 2009.

[17] P .Sibi и другие. *Analysis of different activation functions using back propagation neural networks.* Journal of Theoretical and Applied Information Technology, 47(3), 2013.

[18] Y. LeCun и другие. *Deep learning.* Nature 521: 436–444, 2015.