Автоматизация нечеткого поиска в задаче распознавания старопечатных кириллических текстов

М.Н. Мокроусов[1]

[1]Ижевский государственный технический университет имени М.Т. Калашникова, Студенческая 7, Ижевск, Россия, 426069

Аннотация. В статье описывается вариант решения проблемы выделения слов в старопечатных кириллических текстах после этапа графического распознавания символов на сканированных документах. В статье предложен алгоритм нечеткого текстового поиска с использованием грамматического словаря древнерусского языка, с последующей оценкой полноты и точности результатов поиска. Для оценки релевантности и ранжирования результатов поиска разработана методика расчета ранга варианта распознавания символа на основе метрики TF-IDF. Также в статье представлена программная система автоматизированного поиска слов, представлены результаты экспериментов, доказывающие эффективность разработанных алгоритмов и программ.

1. Введение

Актуальность задачи перевода древних кириллических рукописей из графического представления в текстовую форму обусловлена их исключительной ценностью для исторических и лингвистических исследований, проведение которых наиболее эффективно с использованием методов автоматического анализа текста и распознавания электронно-графического представления рукописей. Для максимальной автоматизации процесса оцифровки старинных текстов требуется привлечение методов распознавания образов. Относительно небольшое количество публикаций по системам распознавания рукописных и старопечатных кириллических текстов X–XVIII веков говорит о необходимости совершенствования методов и технологий решения этой проблемы.

В данной работе описывается этап "Уточнение по словарю" после графического распознавания отдельных символов [1]. Данный этап предполагает применение нечеткого алгоритма поиска в словаре различных комбинаций вариантов символов и последующую автоматизированную выборку слов и символов.

Почти для всех алгоритмов нечеткого поиска имеются программные реализации на различных языках программирования, однако, полное заимствование программных решений для задачи затруднено по ряду причин.

Во-первых, старопечатные тексты не имеют разделителей (знаков препинания), что затрудняет процесс выделения отдельных слов в исходном тексте. Максимальная длина слова в словаре – 26 символов, минимальная – 1 символ, который может обозначать цифру. Количество вариантов распознавания каждого символа может быть в диапазоне от 1 до 6. С учетом максимальной длины слова, количество возможных цепочек символов, с учетом максимальной

длины в 26 символов, определяется *правилом произведения* комбинаторики теории вероятностей. Максимальное количество поисковых запросов для одного только потенциально длинного слова, в таком случае, будет равно 6^{26}, что неоправданно много.

Во-вторых, в процессе графического распознавания изображений невысокого качества, "зашумленных" различными пятнами, потертостями, разрывами страниц с последующей склейкой, последствий реставрации печатных документов, результат распознавания может иметь следующие ошибки:

1) один символ распознается как два или три;
2) два или три подряд идущих символа распознаются как один;
3) пропуски символов, когда программа распознавания не смогла подобрать вариантов; чаще всего данный случай возникает при наличии темного пятна или царапин;
4) в последовательности вероятностей распознавания символа отсутствует верный вариант;
5) слова нет в словаре.

На текущий момент указанные "шумовые" проблемы могут быть решены лишь с использованием ручной правки результатов графического распознавания.

В-третьих, сложности поиска создают сокращения, применяемые в славяно-русских рукописях с использованием простого или буквенного титла. Такие сокращения, чаще всего, применялись для обозначения слов, связанных с богом и церковью (*богъ, аминъ, троице* и т.п.), а также и для других слов (*дъвица, промеудрость, памать* и т.п.).

В-четвертых, необходимо учитывать морфологические правила преобразования и замены символов. Это различные правила приравнивания отдельных букв и их вариантов или буквосочетаний в конце словоформы, после определенных гласных, согласных или с учетом позиции символов в тексте.

В-пятых, отдельным пунктом, затрудняющим полную автоматизацию процесса распознавания символов, является наличие буквицы (инициала) – первой заглавной буквы текста или главы, изображающейся в форме миниатюры.

Было принято решение использовать рекурсивный метод поиска на основе регулярных выражений, который бы учитывал позицию символа в слове при оценке релевантности поиска. Такой нечеткий "позиционный" (точечный) поиск позволит абстрагироваться от возможных "ошибок" графического распознавания и дать большую свободу эксперту, управляющему поиском.

2. Алгоритм нечеткого «позиционного» поиска по словарю

Для сокращения вариантов символов, полученных на этапе графического распознавания, необходимо выполнить поиск слов в грамматическом словаре древнерусского языка, созданного в рамках проекта Manuscripts.ru [2]. Для этого используется рекурсивный алгоритм поиска на основе регулярных выражений.

2.1. Предварительная обработка входного текста

Входной текст представляет собой последовательность символов с вариантами значений после графического распознавания. Каждый символ представлен следующей структурой:

<Символ>::=‘[’[<Координаты>’,’{<ВариантЗначения>’;’}]’]’

<Координаты>::=<X>’,’<Y>’,’<Ширина>’,’<Высота>

<ВариантЗначения>::=<Значение>’,’<Релевантность>[’,’<Количество_Диактрика>],

где *<Координаты>* – координаты прямоугольной области, в которой заключен символ на исходном изображении; *<X>,<Y>* – координаты верхнего левого угла прямоугольной области символа на изображении; *<Ширина>,<Высота>* – соответственно ширина и высота прямоугольной области символа на изображении; *<ВариантЗначения>* – вариант значения символа; *<Значение>* – написание варианта значения символа; *<Релевантность>* – процент релевантности варианта на этапе графического распознавания; *<Количество_Диактрика>* – количество предыдущих символов, на которые распространяется диактрический символ.

Пример входного текста:

[(422,1281,64,71),в,89,б,92,г,98,н,81,п,80][(506,1280,50,70),л,97,а,94,б,83,о,82]

[(473,1316,72,18),¯,100,2][(583,1268,17,18),·,78][(0,0,0,0),]

Как видно из примера, в тексте могут встречаться диактрические символы:

[(473,1316,72,18),¯,100,2]

В таком случае, после процента релевантности указывается, на сколько предыдущих символов распространяется данный диактрический символ. Также, возможны пустые символы, которые на этапе графического распознавания определить не удалось.

Исходный текст преобразуется в табличную форму, где каждый столбец таблицы содержит варианты значения символа. Количество столбцов равняется количеству символов.

2.2. Генерация регулярных выражений и рекурсивный поиск в словаре

Далее происходит генерация регулярных выражений и рекурсивный поиск в словаре. На данном шаге рассматриваются комбинации вариантов символов длиной от 1 до 10 (настраивается в опциях поиска). Каждая комбинация вариантов представляет собой регулярное выражение, которое подставляется в SQL-запрос:

```
SELECT * FROM Dictionary WHERE word like N'[нн][сс][к][сс][ннп]'
```

Результатом запроса является набор слов, найденных по данному выражению. Если количество слов больше порогового значения (по умолчанию – 100), то такой результат не рассматривается.

Входные данные алгоритма поиска

$Table – массив символов с вариантами распознавания полученный из исходного текста;
|$Table| – количество символов во входном тексте.
$maxWordLength – максимальная длина рассматриваемых комбинаций символов;
$maxWordCount – максимальное количество слов в результате запроса.

Выходные данные

$Result – итоговый массив с наборами слов для каждого символа.

Основной цикл поиска по словарю

```
$b ← 0
$e ← 0
Повторять
{
    РаспознатьСледующееСлово( $b, вернуть $e);
    $b ← $e;
} Пока ($b < |$Table|);
```

Рекурсивная функция поиска

```
РаспознатьСледующееСлово( $b, вернуть $e)
{
    $d ← $e – $b + 1;
    Пока (($d <= $maxWordLength) И ($e < |$Table|))
    {
        $reg ← СгенерироватьРегулярноеВыражение($b, $e);
        $symbol ← $Table[$b];
        if (РезультатЗапросаУжеСуществует($reg, $symbol) = Ложь)
        {
            $res ← ВыполнитьЗапросКСловарю($reg, $symbol);
            $Result ← $Result ∪ $res;
            if (|$res| > 0) И (|$res| <= $MaxWordCount))
            {
                $e2 ← $e +1;
                РаспознатьСледующееСлово($e + 1, вернуть $e2);
```

```
                    }
                }
                $e++;
                $d ← $e - $b + 1;
            }
        }
```

где *$b* – индекс символа, с которого начинается просмотр; *$e* – индекс символа, которым заканчивается просмотр; *$d* – текущая длина рассматриваемой комбинации символов; *$reg* – регулярное выражение по текущей комбинации символов; *$symbol* – первый символ текущей комбинации; *$res* – набор слов текущей комбинации символов.

СгенерироватьРегулярноеВыражение($b, $e) – функция, которая возвращает регулярное выражение для текущей комбинации символов;

РезультатЗапросаУжеСуществует($reg, $symbol) – функция, которая возвращает *Истина*, если для регулярного выражения *$reg* и символа *$symbol* уже существует набор слов, и *Ложь* – в противном случае;

ВыполнитьЗапросКСловарю($reg, $symbol) – функция, которая выполняет поиск в словаре по регулярному выражению *$reg* и первому символу *$symbol* и возвращает набор слов;

СохранитьНабор($res) – функция, которая сохраняет набор слов *$res* в рабочей памяти.

После выполнения основного алгоритма будет сформирован массив с наборами слов для каждой комбинации символов, не превышающей установленного порога.

Таким образом, алгоритм нечеткого поиска состоит из двух основных шагов:

1) преобразование исходного текста в табличную форму;

2) рекурсивный поиск в словаре комбинаций вариантов символов с использованием регулярных выражений.

3. Сокращение вариантов распознавания символов на основе алгоритма нечеткого поиска

Метод сокращения вариантов графического распознавания символов на основе результатов алгоритма нечеткого поиска заключается в оценке релевантности вариантов символов путем вычисления весов каждого варианта символа по таким данным как:

- частота встречаемости символа в результатах нечеткого поиска по словарю;
- частота встречаемости символа в результатах нечеткого поиска по словарю с учетом позиции в слове и длины слова;
- частота встречаемости символа в словаре;
- частота встречаемости символа в словаре с учетом позиции в слове и длины слова.

Для вычисления веса варианта используется метрика TF-IDF [3], адаптированная для данной задачи. Было предложено применять в данной метрике в качестве слова – вариант символа (далее символ), а в качестве документов – наборы слов. Таким образом, *SF* (*symbol frequency*) – важность символа *s* в контексте наборов слов *W*, в которых этот символ участвовал, оценивается по формуле:

$$SF(s, W) = \frac{|(s \in w_i)|}{|W|},$$

где $|s \in w_i|$ – число вхождений символа в наборы слов, полученных по регулярным выражениям с участием данного символа; $|W|$ – общее количество слов в наборах, полученных по регулярным выражениям с участием данного символа. *IDF (inverse dictionary frequency – обратная частота словаря)* – инверсия частоты, с которой некоторый символ *s* встречается в словаре *D*. Для каждого символа в пределах словаря существует только одно значение *IDF*:

$$IDF(s, D) = \log \frac{|D|}{|(s \in d_i)|},$$

где $|D|$ – количество слов в словаре (1846721); $|s \in d_i|$ – количество слов d_i словаря D, в которых встречается символ s.

Для того чтобы учесть позицию p символа s и длину слов l в наборах W, в которых присутствует рассматриваемый символ, формулы расчета SF и IDF можно записать следующим образом:

$$SF(s,W,l,p) = \frac{|(w_i(l,p,s) \in W)|}{|(w_i(p,s) \in W)|},$$

где $w_i(l,p,s)$ – i-е слово наборов W длиной l, на позиции p которого стоит символ s; $w_i(l,p,s) \in W$ – количество слов $w_i(l,p,s)$ в наборах W; $w_i(p,s)$ – i-е слово наборов W, на позиции p которого стоит символ s; $w_i(p,s) \in W$ – количество слов $w_i(p,s)$ в наборах W.

$$IDF(s,D,l,p) = \log \frac{|(d_i(s,p) \in D)|}{|(d_i(s,p,l) \in D)|},$$

где $d_i(s,p)$ – i-е слово словаря D, на позиции p которого стоит символ s; $|d_i(s,p) \in D|$– количество слов $d_i(s,p)$ в словаре D; $d_i(s,p,l)$ – i-е слово словаря D длиной l, на позиции p которого стоит символ s; $|d_i(s,p,l) \in D|$ – количество слов $d_i(s,p,l)$ в словаре D.

Таким образом, вес (релевантность) символа оценивается двумя способами:

1) с учетом числа вхождений символа в наборы слов, полученных по регулярным выражениям с участием данного символа:

$$Weight(s) = SF(s,W) * IDF(s,D)$$

2) учетом позиции символа и длины слов в наборах, в которых присутствует рассматриваемый символ:

$$Weight(s,l,p) = IDF(s,D,l,p) * SF(s,W,l,p)$$

Т.к., в наборах слов W символ s может встречаться на разных позициях p в словах разной длины l, то итоговый вес символа с учетом данных параметров вычисляется как сумма весов:

$$WeightSum(s,l,p) = \sum_{i=1}^{n} Weight_i(s,l,p),$$

где n – количество весов символа s с учетом разных позиций p и длины слова l в наборах слов, в которых данный символ участвует.

Итоговый вес варианта символа вычисляется как сумма весов с учетом числа вхождений символа в наборы и с учетом позиции и длины слов:

$$WeightTotal(s) = Weight(s) + WeightSum(s,l,p)$$

Далее вычисляется ранг $Rank(s_i)$ каждого варианта s_i символа s как порядковый номер в отсортированном по убыванию массиве весов вариантов. Для каждого варианта символа вычисляется 3 ранга по каждому весу: $Rank(Weight(s_i))$, $Rank(WeightSum(s_i,l,p))$, $Rank(WeightTotal(s_i))$. Итоговый ранг $RankTotal$ варианта символа вычисляется как среднее гармоническое рангов варианта символа. Вариант каждого символа ранжируется с учетом $RankTotal$. Т.к. ранг вычислялся по позиции варианта символа в отсортированных массивах весов, то чем ниже ранг, тем выше степень релевантности.

4. Описание эксперимента

Сокращение вариантов графического распознавания текста «Остромирово Евангелие», проводилось по следующей методике:

1) выделялся фрагмент текста;
2) для выделенного фрагмента выполнялся нечеткий поиск слов по словарю;
3) вычислялись веса и ранги вариантов символов на основе результатов поиска;
4) устанавливались корректные варианты символов путем ручного подтверждения найденного слова из результатов поиска по словарю;
5) вычислялись и фиксировались в таблице значения полноты, точности и сокращения вариантов.

Результат поиска слов в словаре распознанного текста 9-10 строк «Остромирово Евангелие» [4] с последующим подтверждением слов показан на рисунке 1.

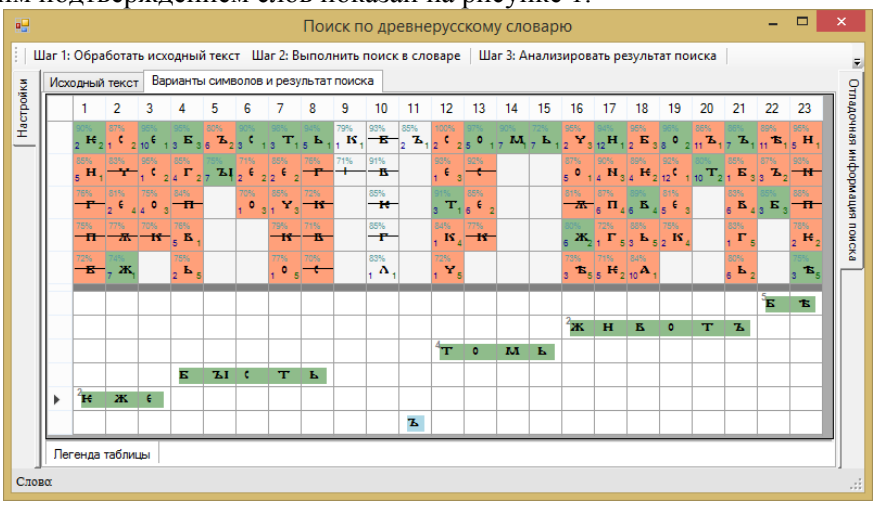

Рисунок 1. Анализ текста 9-10 строк «Остромирово Евангелие».

Ранг варианта символа указан в нижнем правом углу ячейки с вариантом символа, в нижнем левом углу показана частота встречаемости варианта в результатах поиска, а в верхнем левом углу – процент релевантности графического распознавания. Если вариант символа не встречается в найденных наборах слова, то он зачеркивается (частота встречаемости – 0).

В нижней части таблицы зеленым цветом выделены слова, подтвержденные экспертом. Число в верхнем левом углу слова показывает, сколько всего слов нашлось в словаре по комбинации вариантов (количество слов в наборе). Точность и полнота результатов сокращения вариантов оценивались по вариантам с наивысшим рангом (1 и 2).

Пусть S – множество вариантов символов из входного текста; $S^+ \subset S$ – подмножество символов, которые имеют наивысший ранг; $S^\Sigma \subset S$ – подмножество символов, которые использовались при формировании наборов слов (не зачеркнутые варианты); $S^E \subset S$ – подмножество символов, которые были подтверждены экспертом (выделены зеленым); S' – множество символов входного текста (количество столбцов). Помимо *Точности* и *Полноты* оценивался процент *Сокращения вариантов*, не участвовавших в поисковых запросах.

$$Точность = \frac{\left|S^+ \cap S^E\right|}{\left|S'\right|},\ Полнота = \frac{\left|S^+ \cap S^E\right|}{\left|S^E\right|},\ Сокращение = \frac{|S| - \left|S^\Sigma\right|}{|S|}$$

Расчетные параметры полноты и точности сокращения вариантов для данного фрагмента представлены на рисунке 2.

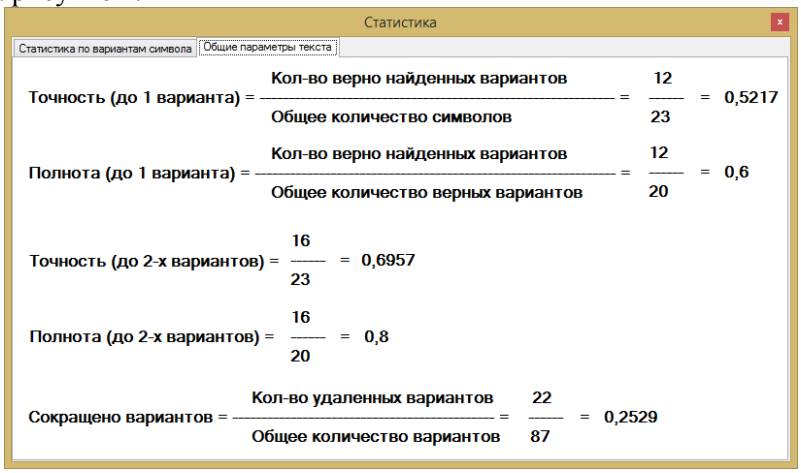

Рисунок 2. Полнота и точность 9-10 строк.

Результаты сводились в таблицу (таблица 1). Точность и полнота до одного (1-го) варианта означает, что при расчете учитывался только один вариант символа, ранг которого равен 1 и который помечен экспертом как корректный. Точность и полнота до двух (2-х) вариантов означает, что при расчете учитывались два варианта символа с высшим рангом, среди которых имеется вариант, помеченный экспертом как корректный.

Таблица 1. Полнота и точность результатов сокращения вариантов.

№ эксперимента	Точность до 1-го варианта	Полнота до 1-го варианта	Точность до 2-х вариантов	Полнота до 2-х вариантов	Сокращено
1	0,67	0,67	0,9	0,9	0,15
2	0,57	0,67	0,86	1	0,18
3	0,52	0,79	0,62	0,93	0,27
4	0,52	0,52	0,83	0,83	0,17
5	0,52	0,6	0,7	0,8	0,26
6	0,77	0,77	0,86	0,86	0,31
7	0,55	0,55	0,83	0,83	0,26
8	0,36	0,45	0,64	0,8	0,14
9	0,24	0,67	0,32	0,89	0,44
10	0,5	0,59	0,62	0,73	0,14
Среднее	**0.52**	**0.63**	**0.72**	**0.86**	**0.23**

Проведенные эксперименты по сокращению вариантов символов на основе алгоритма нечеткого поиска показывают, что предложенный метод сокращения вариантов до одного варианта дает показатель точности 52%, а полноты – 63%, до двух вариантов – 72% и 86%, соответственно.

5. Литература

[1] Кучуганов, А.В. Автоматизация распознавания старопечатных символов с помощью дескрипционных логик / А.В. Кучуганов, Д.Р. Касимов // El'Manuscript–2016. Rašytinis palikimas ir skaitmeninės technologijos: VI tarptautinė mokslinė konferencija, Vilnius. – Vilnius; Iževskas, 2016. – P. 104-109.

[2] Портал "Манускрипт" [Электронный ресурс]. – Режим доступа: http://manuscripts.ru/ (15.12.2016).

[3] Jurafsky, D. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition / D. Jurafsky, J.H. Martin. — New Jersey: Prentice Hall, 2000. – 934 p.

[4] Остромирово евангелие. Листы рукописи // Российская национальная библиотека [Электронный ресурс]. – Режим доступа: http://www.nlr.ru/exib/Gospel/ostr/ill.html (01.12.2018).

Благодарности

Автор выражает благодарность сотрудникам кафедры АСОИУ ИжГТУ имени М.Т. Калашникова Касимову Денису Рашидовичу, Кучуганову Александру Валерьевичу, Кучуганову Валерию Никоноровичу за ценные советы и рекомендации, а также подготовку экспериментальных данных.

Fuzzy search automation in the problem of recognition of old Cyrillic texts

M.N. Mokrousov[1]

[1]Kalashnikov Izhevsk State Technical University, Studencheskaya 7, Izhevsk, Russia, 426069

Abstract. The article describes a solution to the problem of word allocation in old Cyrillic texts after the stage of graphic character recognition on scanned documents. The article proposes an algorithm for fuzzy text search using the grammatical dictionary of the old Russian language, with the completeness and accuracy of search results evaluate. To assess the relevance and ranking of the search results, a method for calculating the rank of the symbol recognition variant based on the TF-IDF metric is developed. The article also presents a software system of automated word search, presents the results of experiments that prove the effectiveness of the developed algorithms and programs.