УДК 004.02

С.В. Смирнов

КОРРЕКТИРОВКА ОШИБОК ОПТИЧЕСКОГО РАСПОЗНАВАНИЯ НА ОСНОВЕ РЕЙТИНГО-РАНГОВОЙ МОДЕЛИ ТЕКСТА

Смирнов С.В. **Корректировка ошибок оптического распознавания на основе рейтинго-ранговой модели текста.**
Аннотация. Результаты массового оптического распознавания архивных документов необходимо подвергать корректировке с целью сокращения количества ошибок. В работе описывается алгоритм корректировки, учитывающий особенности русского языка и позволяющий обрабатывать корпуса текстов больших объемов в полностью автоматическом режиме. Процесс корректировки разделяется на этапы анализа всего корпуса текстов, подготовки структур данных, отбора слов-кандидатов и их финального ранжирование. Использование рейтинго-ранговой модели текста для генерации корректировок позволяет обрабатывать тексты, содержащие узкоспециализированную терминологию, различных предметных областей.
Ключевые слова: оптическое распознавание, OCR, корректировка ошибок распознавания, обработка текста, алгоритм, функция хеширования.

Smirnov S.V. **OCR error correction based on rating-rank model of text.**
Abstract. OCR results of archival documents have to be corrected in order to improve accuracy. An algorithm that takes into account peculiarities of the Russian language and allows handling large volumes of text corpus in fully automatic mode is described. The correction process is divided into stages of analysis of the entire corpus of texts, preparation of data structures, the selection of word candidates and their final ranking. Using rank-rating model for generating text corrections allows handling texts containing specific terminology from different subject areas.
Keywords: OCR, detection and correction of OCR errors, text processing algorithms, hash function.

1. Введение. В наше время системы оптического распознавания изображений текста достигают высокой точности результатов при обработке современных качественных документов. В случае же работы с документами, происхождение которых датируется несколькими десятками лет назад, результат оказывается менее качественным и эффективность применения средств автоматического распознавания снижается.

В проектах по оцифровке документов культурного наследия основную массу составляют именно исторические документы, при массовом распознавании которых, становится недостаточным применение лишь средств оптического распознавания. Результаты, получаемые на выходе систем распознавания необходимо подвергать последующей автоматической постобработке с целью коррекции допущенных оши-

бок, более подробно проблематика построения систем массового оптического распознавания архивных документов описывается в [1].

2. Особенности корректировки ошибок распознавания текста и постановка задачи. Качество процесса корректировки во многом зависит от точности нахождения ошибок и их верной классификации. Общая схема классификации ошибок в текстах и уточненная классификация ошибок в результатах оптического распознавания приводятся в работах [2, 3] соответственно.

Известные подходы контекстной пост-обработки [2–5] включают статистические и лингвистические методы, использующие Скрытые Марковские Модели (СММ) [6], конечные автоматы, нейронные сети, N-граммы символов и слов [7,8], алгоритмы нечеткого отображения строк [3]. Существуют методы использующие специальную внешнюю информацию, частично определенный синтаксис [9] или комбинированные подходы [2].

В работе [10] приводится более подробный обзор методов и подходов, применимых к задаче контекстной обработке результатов распознавания, выделен ряд способов предобработки результатов распознавания путем шаблонной замены, сравнения выходных значение нескольких OCR систем, описаны методы обнаружения ошибок, основанные на оценке вероятности, n-грамм анализе, словарной проверке.

Отдельно стоит выделить класс методов [3, 11, 12], относящихся к обработке орфографических ошибок, поскольку эта тема является намного более глубоко исследованной.

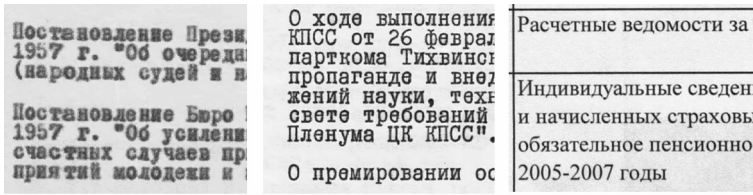
Существующие методы в общем случае неплохо решают ряд задач по обработке результатов распознавания с использованием словарей, статистических моделей языка, хорошо развита тематика обнаружения и коррекции ошибок в тексте. Тем не менее, во многих случаях указанные методы предназначены для обработки современных текстов и не подходят в чистом виде для обработки исторических текстов, содержащих большое количество специализированных терминов, имен собственных, географических наименований и т.п. В большинстве работ корректировка основана на предварительном ручном обучении системы или участии человека на этапе финального выбора слова-заместителя. Также стоит отметить, очень малое количество работ нацеленных на корректировку именно русскоязычных текстов. Это вызвало потребность разработки алгоритма корректировки, учитывающего особенности русского языка и позволяющего обрабатывать корпуса текстов больших объемов в полностью автоматическом режиме.

Практическая задача заключается в автоматической корректировке распознанных документов центральных государственных архи-

вов Санкт-Петербурга, специализирующихся на хранении документов различной направленности: историко-политические документы, документы по литературе и искусству, документы по личному составу ликвидированных государственных предприятий, научно-техническая документация.

Общий объем корпуса материалов составляет чуть менее 50 тысяч документов, 1 миллиона изображений и 200 миллионов слов.

Примеры изображений документов приведены на рисунке 1. Формат изображений – JPEG, разрешение - 300dpi.



а) б) в)

Рис. 1. Примеры изображений: а) печатная машинка, среднее качество; б) печатная машинка, высокое качество; в) принтер, очень высокое качество

Из существующих методов в данной работе используются алгоритм нахождения минимального расстояния между словами (расстояние Левенштейна [13]) и алгоритм поиска схожих слов методом анаграмм [14]. Выбранные алгоритмы позволяют обрабатывать ошибки типичные для систем оптического распознавания, не требуют проведения предварительного обучения и могут применяться для обработки текстов независимо от языка написания.

3. Корректировка результатов распознавания. Разделим весь процесс корректировки результатов распознавания на три основных этапа:

1. Подготовка структур данных.
2. Генерация корректировок.
3. Формирование результата.

В ходе предварительного этапа подготовки структур данных производится сбор статистической информации по всему корпусу распознанных документов, формируется целый ряд словарей и хэш-таблиц, содержащих необходимые данные для этапа генерации корректировок.

Этап генерации корректировок является основным этапом обработки, на котором для каждого ошибочно распознанного слова формируются списки слов-кандидатов на замену, отсортированные по рейтингу. Все множество ошибок распознавания можно разделить на

множество ошибок 1-го рода (пропущенные слова) и множество ошибок 2-го рода (ошибочно распознанные слова). На этапе генерации корректировок обработке подвергаются только ошибки 2-го рода.

На последнем этапе производится выборка наиболее вероятных слов-заместителей, их подстановка и сохранение финального результата распознавания в формате XML (eXtensible Markup Language).

4. Подготовка структур данных. На первом шаге необходимо произвести анализ всего корпуса распознанных документов для формирования статистической информации по встречающимся словам.

4.1. Предварительная обработка. Назовем лексемой последовательность символов, разделенных пробельными символами или определенных системой распознавания как слова.

Выразим весь набор лексем, полученных в результате распознавания документов, в виде упорядоченного по порядку следования элементов множества $L = \{s_1, s_2, ..., s_m\}$.

Под процедурой нормализации будем понимать преобразование последовательности L в нормализованную последовательность лексем $L^N = \{s_1, s_2, ..., s_n\}$.

Процедура нормализации состоит из следующих шагов:

1. Объединение лексем, разделенных знаком переноса.

Две лексемы s_1 и s_2 объединяются в одну, в том случае если:

a) Лексемы расположены на разных строках.

b) Лексема s_1 заканчивается символом «-» после удаления всех неалфавитных символов в конце лексемы. Неалфавитными символами будем считать символы, не входящие во множество символов русского алфавита {а-я, А-Я}.

c) В лексеме s_1 перед символом «-», стоит символ в нижнем регистре

d) Лексема s_2 начинается с символа в нижнем регистре после удаления всех неалфавитных символов в начале лексемы.

2. Разбиение лексем.

В дополнение к разбиению по пробельным символам лексемы следует разбивать по следующим символам {, . ; : ()\"&[]?!´‘{}/+# =< > %}, так как системам распознавания свойственно допускать ошибки сегментации слов.

3. Очистка лексем.

После дополнительного разбиения лексем удаляются все неалфавитные символы с начала и конца каждой лексемы.

4.2. Структуры для отбора слов-кандидатов. Получив нормализованную последовательность L^N, сформируем множество лексем в нижнем регистре символов L^{low} и его рейтинговое распределение $\xi_{L^{low}}$:

$$L^{low} = \{lower(s) \mid s \in L^N\},$$

где $lower(s)$ — функция перевода строки в нижний регистр.

Исходя из предположения, что наиболее часто встречающиеся лексемы с наибольшей вероятностью не содержат ошибок, а также с целью уменьшения поискового пространства проведем сокращение множества L^{low} и его рейтингового распределения $\xi_{L^{low}}$ до множества L^{lp} и рейтингового распределения $\xi_{L^{lp}}$:

$$L^{lp} = \{s \mid s \in L^{low}, \xi_{L^{low}}(s) \geq \alpha\},$$

где α — минимально допустимое количество повторений одной лексемы. Выбор значения α является своего рода компромиссом. При низком значении может остаться большее количество ошибочных лексем, а при высоком могут быть потеряны редкие имена собственные, географические наименования и т.п.

Далее проведем сбор статистической информации о вхождении парных лексем. Для этого сформируем множество биграмм L^{bigram} и его рейтинговое распределение $\xi_{L^{bigram}}$:

$$L^{bigram} = \left\{\left(lower(s_1), lower(s_2)\right) \mid s_1, s_2 \in L^N; \left(seq(s_1, s_2) \vee seq(s_2, s_1)\right)\right\},$$

где $seq(a_1,...,a_z)$ — последовательность элементов от a_1 до a_z, следующих друг за другом. Порядок следования лексем в паре не имеет значения, то есть пары $seq(s_1, s_2)$ и $seq(s_2, s_1)$ считаются равными.

Сбор биграмм производится без учета знаков препинания. Это обусловливается тем, что в результатах распознавания может присутствовать большое количество ошибочных знаков препинания, полученных из-за наличия «шума» на исходном изображении. Главной задачей является сбор максимального количества биграмм для избегания проблем с разреженностью данных и корректировки ошибочно объединенных слов.

Проведем сокращение множества биграмм L^{bigram} и его рейтингового распределения L^{bigram} до множества L^{bgp} и его рейтингового распределения $\xi_{L^{bgp}}$:

$$L^{bgp} = \left\{(s_1, s_2) \mid (s_1, s_2) \in L^{bigram}; len(s_1), len(s_2) > 1; \xi_{L^{bigram}}(s_1, s_2) \geq \beta\right\},$$

где $len(s)$ — количество символов в строке s, а β — минимальное пороговое значения количества повторений одной биграммы. Ограничение по длине лексемы в биграмме введено для того, чтобы избежать нежелательного разбиения слов при корректировке и сократить пространство поиска.

Сформируем основные структуры данных для генерации кандидатов на замену ошибочных слов: множество слов-кандидатов L^W и хэш-таблицу анаграмм $H^{anagram}$.

$$L^W = L^{lp} \bigcup \left\{"s_1\ s_2" \mid (s_1, s_2) \in L^{bgp}\right\},$$
$$H^{anagram} = \left\{\left(hash(s), (s, \xi_{L^W}(s))\right) \mid s \in L^W\right\},$$

где ξ_{L^W} — рейтинговое распределение множества всех лексем для выбора слов-кандидатов L^W.

Для каждого элемента множеств L^W вычисляется значение хэш-функции $hash(s)$ и производится добавление записи в хэш-таблицу $H^{anagram}$, ключом которой является значение хэш-функции, а значением — список всех элементов с их рейтингом, обладающих соответствующим значением хэш-функции. Описание алгоритма вычисления значения хэш-функции представлено ниже.

4.3. Структуры для ранжирования слов-кандидатов. Произведем нормализацию морфологической формы каждой лексемы множества L^N, используя функцию морфологического анализа $morph$:

$$morph(s) = b, s \in \sum_s \rightarrow b \in \sum_s,$$

где $\sum_s$ – множество словоформ одной лексемы, b – нормальная (базовая) форма лексемы (именительный падеж, единственное число для имен существительных; 1-ое лицо, единственное число, настоящее время для глаголов и т.д.).

Функция морфологического анализа $morph$ обладает следующими свойствами:

$$morph(b) = b\,,\ \forall s \notin \sum_s \rightarrow morph(s) = b, b \notin \sum_s .$$

В результате перевода лексем в нормальную форму получим множество лексем:

$$L^{NF} = \left\{ morph\left(lower(s)\right) \mid s \in L^N \right\}.$$

Реализация функции перевода лексем в нормальную форму с помощью внешнего модуля морфологического анализа системы «АОТ» [15], позволит генерировать нормальные (базовые) формы на основе морфологических предсказаний [16] даже для лексем, отсутствующих в словаре.

Сформируем отношения R_1, R_2 для связок лексем множества L^{NF} и их рейтинговые распределения ξ_{R_1}, ξ_{R_2}:

$$R_1 = L^{NF},$$
$$R_2 = \left\{ (b_1, b_2) \mid b_1, b_2 \in L^{NF}; seq(b_1, b_2) \right\}.$$

4.4. Словарные структуры данных. Опишем следующую структуру данных — словарь D^{corpus}, который будет впоследствии применяться для ограничения области лексем, подлежащих корректировке:

$$D^{corpus} = L^{lp} \cap \left(D^{gen} \cup D^{spec} \right),$$

где D^{gen} — словарь общих слов русского языка, D^{spec} — словарь специфических терминов конкретной предметной области документов (словарь имен собственных, фамилий, географических наименований и т.п.).

5. Генерация корректировок. Пусть последовательность $S = \{s_1, s_2, ..., s_m\}$ — упорядоченный по порядку следования набор лексем, полученных в результате распознавания отдельного изображения документа.

Тогда $S^N = \{s_1, s_2, ..., s_n\}$ — результат нормализации последовательности S.

Задача генерации корректировок состоит из двух подзадач:

1. Отбор полного множества слов-кандидатов W_i^C из всего множества слов L^W для корректировки каждой лексемы s_i последовательности S^N.

2. Ранжирование отобранных слов-кандидатов W_i^C и выявление среди них наиболее вероятных $\hat{W}_i$.

В общем виде эту задачу можно представить следующим образом:

$$\hat{W} = \left\{ \arg\max_w^k \left(G(w,s)\right) \mid w \in W^C, W^C \subset L^W, s \in S^N, G(w,s) \in [0..1] \right\},$$
$$1 \le |\hat{W}| \le k,$$

где $G(w,s)$ — степень достоверности того, что результатом корректировки лексемы s является слово w; W^C — множество всех отобранных слов-кандидатов; $\arg\max_w^k \left(G(w,s)\right)$ — список размером k слов w, при которых функция $G(w,s)$ принимает свои максимальные значения.

Разделим все множество лексем S^N на множество лексем S^{err}, подлежащих корректировке, и множество лексем S^{cor}, которые будем считать корректно распознанными:

$$S^N = S^{err} \cup S^{cor},\ S^{err} = S^N \setminus S^{cor}.$$

В область лексем S^{cor}, не подлежащих корректировке, отнесем лексемы, для которых найдено соответствие в словаре D^{corpus} или длина которых меньше порогового значения φ:

$$S^{cor} = \left\{ s \mid s \in S^N; \left(len(s) \le \varphi \vee s \in D^{corpus} \right) \right\}.$$

Тогда корректно распознанные лексемы S^{cor} будут обладать следующим свойством:

$$\forall s \in S^{cor} \rightarrow w = s \rightarrow R(w,s) = 1,\ W^C = \{s\}.$$

Таким образом, задача процесса генерации корректировок сокращается до формирования списка наиболее вероятных слов-кандидатов $\hat{W}$ из множества L^W для каждого элемента множества лексем S^{err}.

5.1. Отбор слов-кандидатов. Основой метода анаграмм [14,17] является предварительно сформированная хэш-таблица анаграмм $H^{anagram}$. Ключом таблицы является «плохая» хэш-функция, которая выдает числовое значение одинаковое для всех слов, являющихся анаграммами. Значениями таблицы выступают соответствующие ключу списки лексем.

Значение хэш-функции будем называть анаграммным ключом. Каждому символу c_i используемого алфавита $A = c_1,...,c_{|A|}$, сопоставим уникальный числовой идентификатор из кодовой таблицы UTF-8. Значение хэш-функции для слова $w = c_1,...,c_{|w|}$ вычисляется по следующей формуле:

$$hash(w) = \sum_{i=1}^{|w|} \text{int}(c_i)^n ,$$

где $\text{int}(c)$ — значение числового кода символа c в кодовой таблице UTF-8, а n — значение степени, в которую возводится числовой код. Возведение числового кода в степень n необходимо для устранения появления коллизий, в работе [14] установлено достаточное значение $n = 5$.

Введем ряд определений для описания отбора слов-кандидатов по методу анаграмм.

Ключевое слово — слово, требующее корректировки, для которого происходит отбор слов-кандидатов на замену.

Алфавит A_W^k, содержащий n-граммы всех символов и знаков, входящих в состав списка слов W, с добавлением символа пробела в начало и конец каждого слова:

$$A_W^k = \{ngrams(sws, n) \mid w \in W, 1 \le n \le k\},$$

где s — символ пробела, $ngrams(w, n)$ — функция, возвращающая список n-грамм слова w.

Строгий алфавит A_W^{k-}, содержащий n-граммы всех символов и знаков, входящих в состав списка слов W, без добавления символа пробела:

$$A_W^{k-} = \{ngrams(w, n) \mid w \in W, 1 \le n \le k\}.$$

Поисковый алфавит X_W^k, состоящий из анаграммных ключей символьных n-грамм списка слов W:

$$\mathrm{X}_W^k = \{hash(\delta) \mid \delta \in A_W^k\}.$$

Ключевой алфавит Φ_W^k, состоящий из анаграммных ключей символьных n-грамм ключевого слова w:

$$\Phi_W^k = \{hash(\delta) \mid \delta \in A_{\{w\}}^{k-}\}.$$

Отметим, что ключевой алфавит строится на основе строгого алфавита.

Главной задачей метода анаграмм является нахождение списка слов-кандидатов W^C, схожих по написанию с заданным ключевым словом. Схожие слова могут быть получены путем вставки, удаления, замены или перестановки символов [18].

Выделим основную характеристику хэш-функции:

$$hash(w) = \sum_{i=1}^{|w|} hash(c_i) = hash(c_1 \ldots c_{k-1}) + hash(c_k) + hash(c_{k+1} \ldots c_{|w|}),$$
$$\forall 1 \le k \le |w|,$$

из которой следует, что общее значение может быть рассчитано, как сумма значений элементов произвольной длины.

Данная характеристика позволяет определить следующие четыре пути по извлечению всего набора схожих слов для ключевого слова w:

1. Перестановка. Для извлечения всех слов, полученных путем перестановки символов, достаточно однократной выборки из хэш-таблицы по анаграммному ключу исходного слова:

$$hash(слово) = hash(слоов) = hash(волос).$$

2. Удаление. Каждый символ или символьную последовательность ключевого алфавита Г необходимо вычесть из анаграммного ключа ключевого слова:

$$hash(сллово) - hash(л) = hash(слово).$$

3. Вставка. Каждый символ или символьную последовательность поискового алфавита П необходимо прибавить:

$$hash(слво) + hash(о) = hash(слово).$$

4. Замена. Каждый символ или символьную последовательность ключевого алфавита Г необходимо вычесть, а каждый символ или символьную последовательность поискового алфавита П прибавить:

$$hash(слого) - hash(г) + hash(в) = hash(слово).$$

На практике, операция «замена» является общей для оставшихся трех операций:

a) для получения операции «удаление» достаточно прибавить 0 и вычесть элемент ключевого алфавита;

a) для получения операции «вставка» достаточно вычесть 0 и прибавить элемент поискового алфавита;

b) для получения операции «перестановка» достаточно ничего не вычитать и не прибавлять.

Рассмотрим псевдокод алгоритма отбора слов-кандидатов по методу анаграмм (листинг 1):

```
#w – ключевое слово
#ldLimit – пороговое значение расстояния Левенштейна
#searchAlph – поисковый алфавит
#keyAlph – ключевой алфавит
Function anagramSearch(w, ldLimit, searchAlph, keyAlph) {
W^C ← ∅;
R^C [] ← ∅;
keyWordHash ← hash(w);
for all ck ∈ keyAlph do
    for all cs ∈ searchAlph do
        simulatedWordHash ← keyWordHash + cs - ck;
        W' ← getWords(simulatedWordHash);
        for all w' ∈ W' do
            if LD(w, w') ≤ ldLimit then
                W^C ← W^C ∪ w';
                R^C[w'] = R^C[w'] + 1;
            end if
        end for
    end for
end for
return W^C;
}
```

Листинг 1. Алгоритм отбора слов-кандидатов по методу анаграмм

Функция $getWords(simulatedWordHash)$ возвращает список слов-анаграмм по хэшу $simulatedWordHash$ из хэш-таблицы анаграмм $H^{anagram}$.

Между каждым выбранным словом-кандидатом и ключевым словом вычисляется расстояние Левенштейна, и если оно больше заданного порога $ldLimit$, то слово-кандидат исключается из результирующего списка.

В результате, некоторые слова могут быть получены несколько раз, чем чаще слово было получено, тем выше его достоверность. Количество повторений сохраняется для каждого слова в ассоциативном массиве R^C.

Рассмотрим особенности адаптации метода анаграмм в данной работе.

Алфавиту A_W^k установим значение $k = 2$, а множество $W = L^{lp}$. Введем дополнительное ограничение на построение алфавита A_W^2 такое, что в его состав могут входить только те символы и символьные последовательности, из которых формируются корректные слова:

$$C = \{a..я\} \bigcup \{-, s\},$$

$$A_W^2 = C \bigcup \{c_1, c_2 \mid seq(c_1, c_2) \in sws; w \in W; c_1, c_2 \in C\},$$

где s — пробельный символ. Добавление пробельного символа в начало и конец слова w необходимо для обеспечения возможности корректировки ошибочно объединенных слов.

В поисковый алфавит и ключевой алфавит добавим нули. Таким образом, основная формула для вычисления слов-кандидатов для ключевого слова w выглядит следующим образом:

$$simulatedWordHash \leftarrow keyWordHash + cs - ck,$$

$$cs \in \{0\} \cup X_W^2, ck \in \{0\} \cup \Phi_{\{w\}}^2.$$

5.2 Ранжирование слов-кандидатов. После получения множества слов-кандидатов W^C, необходимо определить вероятность каждого из них. Применим для этого двухшаговую модель ранжирования:

1) на первом шаге производится оценка каждого слова-кандидата, отобранного методом анаграмм, и упорядочивание множества W^C по убыванию оценки;

2) на втором шаге для каждого из ν слов-кандидатов вычисляется финальный ранг $G(w,s)$.

Шаг 1. Оценка соответствия слова-кандидата w для замены лексемы s [2]:

$$score_A(s,w) = \ln(f(w)) \times (|w| - LD(s,w)) \times r(w),$$

где $f(w) = \xi_{L^W}(w)$ — частота повторения слова w во всем корпусе слов L^W; $|w|$ — длина слова w в символах; $LD(s,w)$ — расстояние Левенштейна между словами s и w; $r(w) = R^C[w]$ — количество повторений слова w в ходе отбора методом анаграмм.

Вычисление оценки $score_A(s,w)$ основывается на двух предположениях.

1. Исходное множество слов L^W, из которого выбираются слова-кандидаты, может содержать ошибки распознавания, так как оно было получено из корпуса распознанных документов. Вследствие этого, будем считать, что слова, обладающие наибольшей частотой повторения $f(w)$, наиболее вероятно являются корректными и будем отдавать им предпочтение при ранжировании слов-кандидатов.

2. Чем меньше расстояние Левенштейна $LD(s,w)$ между исходной лексемой и словом-кандидатом, тем более предпочтительным будем его считать.

Таким образом, на данном шаге производится вычисление оценки $score_A(s,w)$ для каждого слова-кандидата из множества W^C и формируется упорядоченное по убыванию вычисленной оценки множество:

$$\vec{W}^C = \{w \mid w \in W^C, score_A(s,w_i) \geq score_A(s,w_{i+1}), 1 \leq i \leq |W^C|\}.$$

Шаг 2. Вычисление финального ранга.

Вычисление финального ранга $G(w,s)$ произведем для ν слов-кандидатов, используя следующую систему оценки:

$$G(w,s) = \frac{score_A(s,w)}{\sum_{i=1}^{\nu} score_A(s,w)} \times P(w),$$

где ν — число, определяющее максимальное количество слов-кандидатов для нормализации оценки степени достоверности $score_A(s,w)$; $P(w)$ — статистическая вероятность нахождения слова-кандидата w в тексте на позиции лексемы s.

Выразим $P(w_i)$ через языковую модель следующим образом:

$$P(w_i) = P(w_i \mid w_{1,i-1}),$$

где $P(w_i \mid w_{1,i-1})$ — вероятность появления слова w_i при наличии предшествующей ему последовательности слов $w_1, w_2, ..., w_{i-1}$.

В случае языковой модели, основанной на биграммах, данное определение можно упростить:

$$P(w_i) = P(w_i \mid w_{1,i-1}) = \frac{f(w_{i-1}, w_i)}{f(w_{i-1})},$$

где $f(w_{i-1})$ — частота повторения слова, $f(w_{i-1}, w_i)$ — частота повторения биграммы (w_{i-1}, w_i).

В данной работе, вместо одного слова w_{i-1} выступает множество слов-кандидатов $\vec{W}_{i-1}^C$, а информация о частоте повторения слов и биграмм получается из рейтинговых распределений отношений связок лексем в нормальной форме ξ_{R_1}, ξ_{R_2}.

Формула расчета вероятности принимает вид:

$$P(w_i^k) = \frac{\sum_{j=1}^{\min(\nu, |\vec{W}_{i-1}^C|)} f(w_{i-1}^j, w_i^k)}{\sum_{j=1}^{\min(\nu, |\vec{W}_{i-1}^C|)} f(w_{i-1}^j)},$$

$$1 \le k \le |\vec{W}_i^C|,$$

$$f(w_{i-1}^j) \leftarrow \xi_{R_1}(morph(w_{i-1}^j)),$$

$$f(w_{i-1}^j, w_i^k) \leftarrow \xi_{R_2}(morph(w_{i-1}^j), morph(w_i^k)),$$

где w_i^k — k-ое по порядку слово-кандидат на замену лексемы s_i, w_{i-1}^j — j-ое по порядку слово-кандидат на замену лексемы s_{i-1}.

6. Способ оценки качества результатов корректировки. Оценку качества результатов корректировки можно проводить путем сравнения характеристик результатов распознавания различных OCR систем с использованием автоматической корректировки и без нее.

Характеристики вычисляются путем сопоставления результата распознавания с ранее подготовленным эталонным текстом и сбором следующего ряда метрик [19]: коэффициент распознанных символов, коэффициент распознанных слов, точность в символах, точность в словах, точность в словах без учета порядка, словарная точность.

7. Заключение. Предложенный в работе метод предназначается для автоматической корректировки результатов массового оптического распознавания документов.

Отбор слов-кандидатов для корректировки осуществляется по специальным глоссариям, построенным на основе частотных характеристик повторений слов и словосочетаний со всего корпуса распознанных материалов. Использование таких глоссариев, позволяет производить корректировку текстов различных предметных областей, содержащих узкоспециализированную терминологию, имена собственные, географические наименования и т.п.

Финальное ранжирование отобранных слов-кандидатов производится с учетом контекста и основывается на результатах статистического n-грамм анализа всего корпуса текста в нормальной форме. Реализация функции перевода слова в нормальную форму на основе морфологических предсказаний позволяет генерировать нормальные (базовые) формы даже для слов русского языка, отсутствующих в словаре.

При корректировке корпусов большого объема существенное значение имеет скорость выполнения обработки. В данной работе для увеличения скорости производится корректировка только слов с некорректным написанием, также не подвергаются корректировке слова с малым количеством символов. Данный подход не может обеспечить корректировку слов, присутствующих в словаре, но некорректно примененных в текущем контексте.

Литература

1. *Смирнов С.В.* Подсистема массового распознавания изображений архивных документов // Труды СПИИРАН. 2012. Выпуск 3(22). С. 234–248.
2. *Kai N.* Unsupervised Post-Correction of OCR Errors // Hannover: Leibniz University. 2010.
3. *Kukich K.* Techniques for automatically Correcting Words in Text // ACM computing survey Computational Linguistic. 1992. vol. 24. no. 4. pp. 377–439.
4. *Mailburg M.* Comparative Evaluation of Techniques for Word Recognition Improvement by Incorporation of Syntactic Information // 4th International Conference Document Analysis and Recognition (ICDAR ‘97). 1997. p. 784.

5. *Beitzel S., Jensen E., Grossman D.* A Survey of Retrieval Strategies for OCR Text Collections // Proc. of 2003 Symposium on Document Image Understanding Technology. 2003.
6. *Chen D., Mao J., Mohiuddin K.* An Efficient Algorithm for Matching a Lexicon with a Segmentation Graph // Fifth International Conference on Document Analysis and Recognition. 1999. pp. 543-546.
7. *Mays E., Damerau F.J., Mercer R.L.* Context Based Spelling Correction // Inf. Process. Manage. 1991. vol. 27. no. 5. pp. 517–522.
8. *Fossati D., Barbara Di Eugenio.* A Mixed Trigrams Approach for Context Sensitive Spell Checking // Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing '07). 2007. pp. 623–633.
9. *Шоломов Д.Л., Постников В.В., Марченко А.А., Усков А.В.* Пост-обработка результатов OCR распознавания, использующая частично определенный синтаксис // Труды ИСА РАН. 2005. Т.16. С. 146 163.
10. *Смирнов С.В.* Методы автоматической постобработки результатов распознавания в задачах оцифровки архивных документов // Информационно-измерительные и управляющие системы. 2013. №9. С. 22–32.
11. *Philips L.* The Double Metaphone Search Algorithm // C/C++ Users J. 2000. vol. 18. no. 6. pp. 38–43.
12. *Pollock J., Zamora A.* Automatic Spelling Correction in Scientific and Scholarly Text // Commun. ACM. 1984. vol. 27. no. 4. pp. 358–368.
13. *Левенштейн В.* Двоичные коды с исправлением выпадений, вставок и замещений символов // Доклады Академий Наук СССР. 1965. Т. 163. № 4. С. 845-848.
14. *Reynaert M.* Text Induced Spelling Correction // Proceedings of the 20th international conference on Computational Linguistics (COLING '04). 2004. pp. 834.
15. Автоматическая обработка текста. URL: www.aot.ru (дата обращения 10.05.2014).
16. *Сокирко А.В.* Морфологические модули на сайте www.aot.ru // Материалы конференции «Диалог-2004». 2004.
17. *Reynaert M.* Corpus-Induced Corpus Clean-up // Fifth International Conference on Language Resources and Evaluation (LREC '2006). 2006.
18. *Damerau F.J.* A technique for computer detection and correction of spelling errors // Commun. ACM. 1964. vol. 7. no. 3. pp. 171–176.
19. *Смирнов С.В.* Критерии оценки качества результатов оптического распознавания // Сборник материалов XVI Международной научно-практической конференции «Перспективы развития информационных технологий». Новосибирск. 2013. С. 33–38.

References

1. Smirnov S.V. [Subsystem of mass image recognition of archival documents]. *Trudy SPIIRAN – SPIIRAS Proceedings*. 2012. vol. 3 (22). pp. 234 248. (In Russ.).
2. Kai N. Unsupervised Post-Correction of OCR Errors. Hannover: Leibniz University. 2010.
3. Kukich K. Techniques for automatically Correcting Words in Text. ACM computing survey Computational Linguistic. 1992. vol. 24. no. 4. pp. 377–439.
4. Mailburg M. Comparative Evaluation of Techniques for Word Recognition Improvement by Incorporation of Syntactic Information. 4th International Conference Document Analysis and Recognition (ICDAR '97). 1997. p. 784.
5. Beitzel S., Jensen E., Grossman D. A Survey of Retrieval Strategies for OCR Text Collections. Proc. of 2003 Symposium on Document Image Understanding Technology. 2003.
6. Chen D., Mao J., Mohiuddin K. An Efficient Algorithm for Matching a Lexicon with a Segmentation Graph. Fifth International Conference on Document Analysis and Recognition. 1999. pp. 543-546.

7. Mays E., Damerau F.J., Mercer R.L. Context Based Spelling Correction. Inf. Process. Manage. 1991. vol. 27. no. 5. pp. 517–522.
8. Fossati D., Barbara Di Eugenio. A Mixed Trigrams Approach for Context Sensitive Spell Checking. Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing '07). 2007. pp. 623–633.
9. Sholomov D.L., Postnikov V.V., Marchenko A.A., Uskov A.V. [Post-processing of OCR recognition results using partially defined syntax]. *Trudy ISA RAN – ISA RAS Proceedings*. 2005. vol. 16. pp. 146–163. (In Russ.).
10. Smirnov S.V. [Methods of post-correction of OCR errors in problems of automatic processing of archival documents]. *Informatsionno-izmeritelnye i upravlyayushchie sistemy – Information-measuring and Control Systems*. 2013. vol. 9. pp. 22–32. (In Russ.).
11. Philips L. The Double Metaphone Search Algorithm. C/C++ Users J. 2000. vol. 18. no. 6. pp. 38–43.
12. Pollock J., Zamora A. Automatic Spelling Correction in Scientific and Scholarly Text. Commun. ACM. 1984. vol. 27. no.4. pp. 358–368.
13. Levenshtein V.I. [Binary Codes Capable of Correcting Deletions, Insertions, and Reversals]. *Doklady Akademij Nauk SSSR – Reports of the USSR Academy of Sciences*. 1965. vol. 163. no. 4. pp. 845–848. (In Russ.).
14. Reynaert M. Text Induced Spelling Correction. Proceedings of the 20th international conference on Computational Linguistics (COLING '04). 2004. pp. 834
15. *Avtomaticheskaja obrabotka teksta* [Automatic text processing]. Available at: www.aot.ru (accessed 10.05.2014). (In Russ.).
16. Sokirko A.V. [Morphological modules on site www.aot.ru]. *Materialy konferencii «Dialog-2004»* [Proceedings of conference "Dialog-2004"]. 2004. (In Russ.).
17. Reynaert M. Corpus-Induced Corpus Clean-up. Fifth International Conference on Language Resources and Evaluation (LREC '2006). 2006.
18. Damerau F.J. A technique for computer detection and correction of spelling errors. Commun. ACM. 1964. vol. 7. no. 3. pp. 171–176.
19. Smirnov S.V [Criteria for evaluating the quality of results OCR]. *Sbornik materialov XVI Mezhdunarodnoj nauchno-prakticheskoj konfe-rencii «Perspektivy razvitija informacionnyh tehnologij»* [Proceedings of the 16th International scientific-practical conference "Prospects of the development of information technologies"]. Novosibirsk. 2013. pp. 33–38. (In Russ.).

Смирнов Сергей Владимирович — начальник сектора, Санкт-Петербургское государственное унитарное предприятие «Санкт-Петербургский информационно-аналитический центр». Область научных интересов: автоматическая обработка изображений документов и текстов. Число научных публикаций — 7. serge.smir@gmail.com; Транспортный пер., д. 6а, г. Санкт-Петербург, 191040, РФ; р.т. +7(812)764-3957, факс +7(812)764-9548.

Smirnov Sergey Vladimirovich — head of sector, St. Petersburg State Unitary Enterprise "St. Petersburg Information and Analytical Centre". Research interests: automatic processing of document images and texts. The number of publications — 7. serge.smir@gmail.com; 6a Transportniy per., St. Petersburg, 191040, Russia; office phone +7(812)764-3957, fax +7(812)764-9548.

РЕФЕРАТ

Смирнов С.В. **Корректировка ошибок оптического распознавания на основе рейтинго-ранговой модели текста.**

В проектах по оцифровке документов культурного наследия основную массу составляют исторические документы. Результаты, получаемые на выходе систем распознавания необходимо подвергать последующей автоматической постобработке с целью коррекции допущенных ошибок.

Существующие методы в большей степени предназначены для обработки современных текстов и не подходят в чистом виде для обработки исторических текстов, содержащих большое количество специализированных терминов, имен собственных, географических наименований и т.п.

Предложенный в работе подход автоматической корректировки результатов массового оптического распознавания документов использует алгоритм нахождения минимального расстояния между словами (расстояние Левенштейна [13]) и алгоритм поиска схожих слов методом анаграмм [14].

Процесс корректировки результатов распознавания разделяется на три этапа: подготовка структур данных, генерация корректировок и формирование результата.

В ходе предварительного этапа подготовки структур данных производится сбор статистической информации по всему корпусу распознанных документов, формируется целый ряд словарей и хэш-таблиц, содержащих необходимые данные для этапа генерации корректировок.

Этап генерации корректировок является основным этапом обработки, на котором для каждого ошибочно распознанного слова формируются списки слов-кандидатов на замену, отсортированные по рейтингу.

Отбор слов-кандидатов для корректировки осуществляется по специальным глоссариям, построенным на основе частотных характеристик повторений слов и словосочетаний со всего корпуса распознанных материалов. Использование таких глоссариев, позволяет производить корректировку текстов различных предметных областей, содержащих узкоспециализированную терминологию, имена собственные, географические наименования и т.п.

Финальное ранжирование отобранных слов-кандидатов производится с учетом контекста и основывается на результатах статистического n-грамм анализа всего корпуса текста в нормальной форме. Реализация функции перевода слова в нормальную форму на основе морфологических предсказаний позволяет генерировать нормальные (базовые) формы даже для слов русского языка, отсутствующих в словаре.

При корректировке корпусов большого объема существенное значение имеет скорость выполнения обработки. В данной работе для увеличения скорости производится корректировка только слов с некорректным написанием, также не подвергаются корректировке слова с малым количеством символов. Данный подход не может обеспечить корректировку слов, присутствующих в словаре, но некорректно примененных в текущем контексте.

SUMMARY

Smirnov S.V. **OCR error correction based on rating-rank model of text.**

Historical documents are the most popular in projects on digitization of cultural heritage documents. Results obtained at the output of the recognition systems need to be subjected to the subsequent automatic post-processing for the correction of errors.

Existing methods mostly intended for processing and modern texts do not fit in pure form for the treatment of historically texts containing a large number of specialized terminology, proper names, geographical names, etc.

The proposed approach uses an algorithm for finding the minimum distance between words (Levenstein distance [13]) and the search algorithm of similar words using Anagram method [14].

The correction process of recognition results is divided into three stages: preparation of data structures, generation of corrections and result formation.

During the preliminary phase of the data structures collection of statistical information on all case recognized documents are formed, a number of dictionaries and hash tables containing the necessary data for the generation step adjustments are produced.

Generating step is a major milestone, which forms the word candidate lists for replacement of the errors, sorted by rating.

Selection of word candidates for corrections performed by special glossary based on the basis of the frequency characteristics of repetitions of words and phrases from around the body recognized materials. The use of such glossaries, allows adjustment of texts of various domains containing highly specialized terminology, proper names, place names, etc.

The final ranking of the selected word candidates is made taking into account the context and based on the results of the statistical n-gram analysis of the entire body of text in normal form. Implementation of word normalization is based on morphological predictions and can generate normal (base) form even for Russian words, which are absent in the dictionary.

Speed of processing is very important during the processing of large volume of data. To increase the velocity only words with incorrect spelling are corrected, also words with a small number of characters are not corrected. This approach cannot correct the words which are present in dictionary but have a wrong interpretation in the context.