ISSN 2708-2032
e-ISSN 2708-2040

INTERNATIONAL UNIVERSITY

INTERNATIONAL JOURNAL OF INFORMATION & COMMUNICATION TECHNOLOGIES

Volume 2, Issue 2
June, 2021

ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
MINISTRY OF EDUCATION AND SCIENCE OF THE REPUBLIC OF KAZAKHSTAN

INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES

МЕЖДУНАРОДНЫЙ ЖУРНАЛ ИНФОРМАЦИОННЫХ И КОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

ХАЛЫҚАРАЛЫҚ АҚПАРАТТЫҚ ЖӘНЕ КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАР ЖУРНАЛЫ

Том 2, Выпуск 2
Июнь, 2021

Издание зарегистрировано Министерством информации и общественного развития Республики Казахстан. Свидетельство о постановке на учет № KZ82VPY00020475 от 20.02.2020 г.

Журнал зарегистрирован в Международном центре по регистрации сериальных изданий ISSN (ЮНЕСКО, г. Париж, Франция)

Выходит 4 раза в год.

УЧРЕДИТЕЛЬ:
АО «Международный университет информационных технологий»

ISSN 2708-2032 (print)
ISSN 2708-2040 (online)

СОДЕРЖАНИЕ

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ИНЖЕНЕРИЯ ЗНАНИЙ

ИНФОКОММУНИКАЦИОННЫЕ СЕТИ И КИБЕРБЕЗОПАСНОСТЬ

ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ

ЦИФРОВЫЕ ТЕХНОЛОГИИ В ЭКОНОМИКЕ И МЕНЕДЖМЕНТЕ

CONTENTS

SOFTWARE DEVELOPMENT AND KNOWLEDGE ENGINEERING

INFORMATION AND COMMUNICATION NETWORKS AND CYBERSECURITY

SMART SYSTEMS

DIGITAL TECHNOLOGIES IN ECONOMICS AND MANAGEMENT

МАЗМҰНЫ

БАҒДАРЛАМАЛЫҚ ҚАМТАМАНЫ ӘЗІРЛЕУ ЖӘНЕ БІЛІМ ИНЖЕНЕРИЯСЫ

АҚПАРАТТЫҚ ЖӘНЕ КОММУНИКАЦИЯЛЫҚ ЖЕЛІЛЕР ЖӘНЕ КИБЕРҚАУІПСІЗДІК

ИНТЕЛЛЕКТУАЛДЫ ЖҮЙЕЛЕР

ЭКОНОМИКА ЖӘНЕ БАСҚАРУДАҒЫ САНДЫҚ ТЕХНОЛОГИЯЛАР

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ИНЖЕНЕРИЯ ЗНАНИЙ

INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES
ISSN 2708–2032 (print)
ISSN 2708–2040 (online)
Vol. 2. Is. 2. Number 06 (2021). Pp. 09–18
Journal homepage: https://journal.iitu.edu.kz
https://doi.org/10.54309/IJICT.2021.06.2.001
УДК 004.021

Бактаев А.Б.*, Мукажанов Н.К.

Международный университет информационных технологий, Алматы, Казахстан

АЛГОРИТМ РЕШЕНИЯ ЗАДАЧИ ПО ИСПРАВЛЕНИЮ ОПЕЧАТОК В ТЕКСТЕ, ПРИМЕНЯЕМЫЙ В ПОИСКОВЫХ СИСТЕМАХ С ПОДДЕРЖКОЙ КАЗАХСКОГО ЯЗЫКА

Аннотация. В статье описывается подход решения и применения алгоритма исправления опечаток или ошибок в словах с поддержкой казахского языка, которые допустили люди при вводе текста. Алгоритм разрабатывается с помощью расстояния Дамерау – Левенштейна и редакционного предписания для поисковой системы. Для точности и релевантности поиска, требуется точные ключевые запросы. В спешке люди могут забыть переключить раскладку, допустить орфографические или другие виды ошибок при наборе текста, и нельзя допустить чтобы это повлияло на качество поиска информации.

Ключевые слова: исправление опечатка в тексте, поисковые запросы, ошибки в запросах, расстояние Дамерау — Левенштейна, редакционное предписание.

Введение

Количество информационных ресурсов, таких как новостные сайты, блоги, площадки электронной коммерции с онлайн продажами, электронные энциклопедии, непрерывно растет. Существует мнение, что поисковая машина для сайта не является важным элементом, заслуживающим внимания и траты ресурсов. Возможно, это правда в каких-то частных ситуациях. Однако в большинстве случаев поисковая машина играет важную роль в жизненном цикле сайта. Данная работа посвящена одной из главных частей поисковой системы — предварительной обработке текстового запроса для улучшения релевантности поиска. Необходимость предварительной обработки запроса перед выполнением поиска обусловлена технологической особенностью поискового процесса: в большинстве случаев поисковые машины используют полнотекстовый поиск, поэтому чем точнее выполнен запрос, тем точнее будет результат.

Принцип предварительной обработки текста можно реализовать с помощью алгоритма редакционного предписания. Также можно применить алгоритм расстояния Левенштейна для определения разности между двумя последовательностями символов. Требуется база данных существующих слов, чтобы вычислительная система понимала, какие слова существуют в естественном языке. Естественный язык включает в себя более тысячи различных языков со своими правилами, синтаксисом и т.д. За основу возьмем три языка: казахский, русский, английский. Люди при наборе текста допускают чаще всего ошибки следующего характера:

- самый распространенный вариант — проявление человеческого фактора — опечатка, которую возможно допустить где угодно. Например, «сегдня сонлечный день». В данном предложении ошибки в словах сегодня и солнечный;
- слова из русского языка, набранные латинскими буквами, или наоборот. Забыли сменить раскладку клавиатуры. Например, «Privet» или «Ghbdtn» вместо «Привет»;
- казахский текст, написанный в русской раскладке. Например, «ынгайлы» вместо «ыңғайлы», или «коркем ан» вместо «көркем ән».

Основная концепция решения задачи по исправлению опечаток

Чтобы получить нужную информацию, найти ответ, необходимо правильно сформулировать поисковой запрос. Результат поиска напрямую зависит от введенного для поиска текста. Текст является набором слов, а слова — последовательностью символов, которая имеет определенное значение. Поскольку мыслительная деятельность протекает быстрее физических действий, при наборе текста пользователи могут допустить ошибку или опечатку. Наша цель — по возможности найти и исправить эти ошибки, чтобы результат поиска всегда был более релевантным.

Предположим, что имеется база данных с миллионами слов и одно слово на вход для того, чтобы найти корректный исходный вариант. Поиск совпадения путем сопоставления длины нужного слова с каждым словом в базе данных, высчитывая сходство между s_1 и s_2, — слишком дорогая и долгая операция. Извлечь данные из памяти быстрее, чем выполнять трудоемкие вычисления при каждом запросе. Все дело в реализации данного функционала. Вместо того чтобы перебирать процент совместимости с каждым словом в базе данных, можно предварительно подготовить набор возможных вариантов с помощью алгоритма редакционного предписания и поискать совпадения в проиндексированной базе данных. Если результаты от предыдущих действий не удовлетворительны, можно расширить диапазон с помощью смены раскладки между кириллицей и латиницей, воспользоваться снова алгоритмом редакционного предписания и попытаться найти совпадения в базе данных уже с другой раскладкой.

Перебирая возможные варианты с помощью редакционного предписания, получаем в результате кандидатов для исправления. Для определения одного варианта, нуждающегося в корректировке, можно использовать множество вариантов разных метрик и подходов. В данной работе сравним теорию вероятности с алгоритмом расстояния Левенштейна.

Алгоритм

Дистанция редактирования между двумя строками (редакционное расстояние) — это наименьшее количество операций вставки, удаления и замены одного символа на другой для превращения одной строки в иную [1].

Расстояние Дамерау-Левенштейна — это расширение еще одной операцией алгоритма расстояния Левенштейна. Операция называется транспозицией, перестановка рядом расположенных символов местами. Алгоритм хорошо работает с однобайтными кодировками, но на некоторых языках допустимо описание текстов только с использованием кодировок с переменной длиной символов, например Юникод [2]. Он выявил, что при наборе текста 4/5 ошибок являются транспозициями [3].

Редакционное предписание — это порядок действий для получения из одной строки второй наикратчайшим образом. Чаще всего действия обозначаются так, как указано в таблице «Обозначения операций редакционного предписания» (таблица 1) [1]. Для операций вставки (insert) используются буквы из алфавита языка, на котором требуется найти слово.

Таыблица 1 – Обозначения операций редакционного предписания

Операция	Полное название на английском	Полное название на русском
D	Delete	удалить
I	Insert	вставить
R	Replace	заменить
T	Transpose	транспозиция
M	Match	совпадение

К примеру, для строк «СӘЛЕМ» и «САЕМ» можно построить преобразование как показано в таблице «Пример работы редакционного предписания» (таблица 2) [3]. Первые

символы совпадают, обозначаем буквой M, что означает Match (совпадение) согласно таблице 1. Следующие операции — замена, вставка, затем снова два совпадения.

Таблица 2 – Пример работы редакционного предписания

Операция	**M**	**R**	**I**	**M**	**M**
Корректное слово	С	Ә	Л	Е	М
Слово с ошибкой	С	А	…	Е	М

Также из слова СӘЛЕМ можно построить разные варианты с ошибками, используя редакционное предписание (рисунок 1).

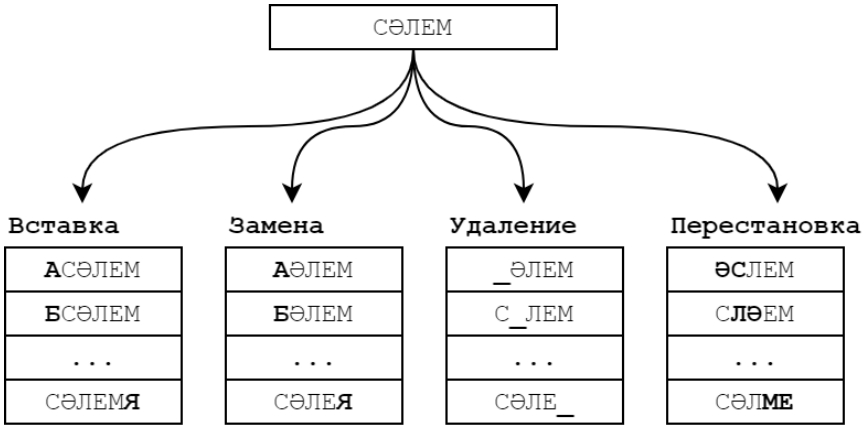

Рисунок 1 - Демонстрация работы редакционного предписания

Можем считать, что все ω неотрицательны: если две последовательные операции можно заменить одной, это не ухудшает общую цену (к примеру, заменить символ x на y, а потом с y на z не лучше, чем сразу x на z) [1].

Очевидно, справедливы следующие утверждения:

$d(s_1,s_2) \geq ||s_1| - |s_2||$

$d(s_1,s_2) \leq \omega(\varepsilon,b) * |s_1| + \omega(a,\varepsilon) * |s_2|$

$d(s_1,s_2) = 0 \leftrightarrow s_1 = s_2$

где $d(s_1,s_2)$ — расстояние с подстановкой между s_1 и s_2, $|s|$ — длина строки s [1].

Основаным минусом является то, что алгоритм требует $O(M * N)$ операций и точно такую же память. Чем длиннее строка, тем дольше операция будет выполняться. Это значит, что потребуется примерно 40 гигабайт памяти для сравнения файлов длиной в 10^5 строк [2]. Учитывая, что областью применения является поисковая система, долгое выполнение не приемлемо. По этой причине мы выставим ограничения по длине каждого слова. Предположительно она будет равна длине самого длинного слова в базе данных.

Определение кандидата с помощью теорий вероятности — по каждому найденному слову-кандидату ищем частоту повторения в базе данных и делим его на количество всех слов в базе данных, и возвращаем кандидата, где получили максимальное значение.

Смена раскладки текста — это преобразование текста с кириллицы на латиницу или наоборот в соответствии с раскладкой на клавиатуре (рисунок 2).

Допустимо наличие нескольких раскладок для одного письменного языка. Например, имеются раскладки ЙЦУКЕН и фонетическая (ЯВЕРТЫ) для русского языка. QWERTY, Дворака и Colemak для английского языка [4].

Клавиатурная раскладка казахского языка разрабатывалась на основе русской раскладки и закреплена стандартом РСТ КазССР 903-90. Раскладка казахского, русского и английского (рисунок 2) [4].

Рисунок 2 - Клавиатурная раскладка казахского, русского и английского языка

Для смены раскладки создадим коллекцию, которая представляет из себя пару ключ-значения (таблица 3).

Таблица 3 – Коллекция ключ-значения для смены раскладки

Latin	q	w	e	r	t	y	u	i	o	p	[	]
Cyrillic	й	ц	у	к	е	н	г	ш	щ	з	х	ъ
Latin	Q	W	E	R	T	Y	U	I	O	P	{	}
Cyrillic	Й	Ц	У	К	Е	Н	Г	Ш	Щ	З	Х	Ъ
Latin	a	s	d	f	g	h	j	k	l	;	'	
Cyrillic	ф	ы	в	а	п	р	о	л	д	ж	э	
Latin	A	S	D	F	G	H	J	K	L	:	"	
Cyrillic	Ф	Ы	В	А	П	Р	О	Л	Д	Ж	Э	
Latin	z	x	c	v	b	n	m	,	.			
Cyrillic	я	ч	с	м	и	т	ь	б	ю			
Latin	Z	X	C	V	B	N	M	<	>			
Cyrillic	Я	Ч	С	М	И	Т	Ь	Б	Ю			
Latin	@	#	$	%	*	(	)	_	+			
Cyrillic	Ә	І	Ң	Ғ	Ү	Ұ	Қ	Ө	h			
Latin	2	3	4	5	8	9	0	-	=			
Cyrillic	ә	і	ң	ғ	ү	ұ	қ	ө	h			

Предварительная обработка текста (корректировка текста) с поддержкой казахского языка решается с помощью комплекса алгоритмов и подходов для решения задачи (рисунок 3). Данный подход учитывает смену раскладки клавитатуры, различные варианты опечаток и т.д.

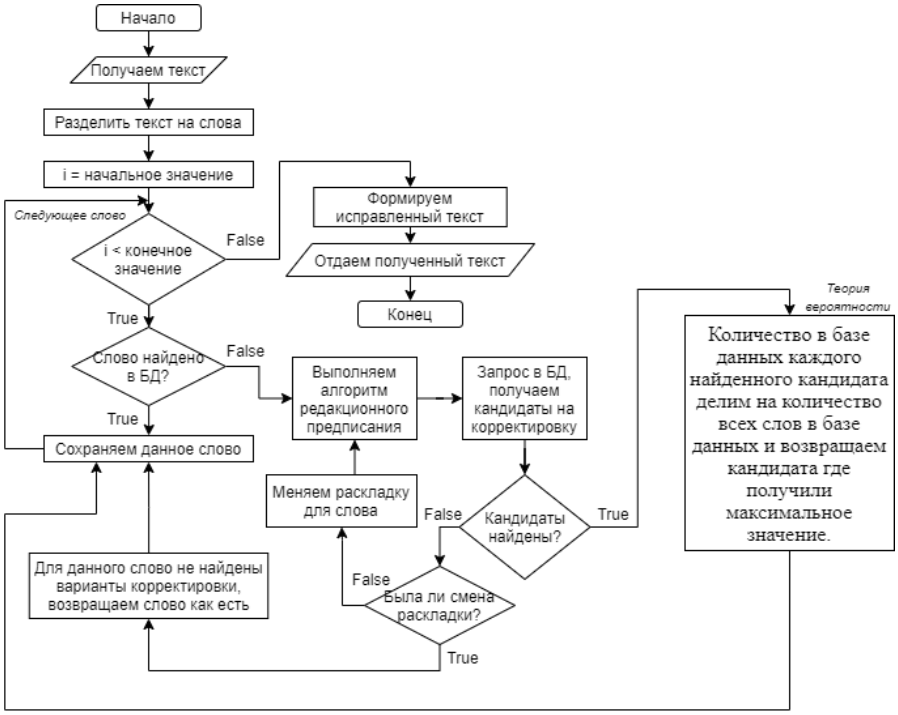

Рисунок 3 - Блок-схема алгоритма исправления корректировки текста

Шаги алгоритма корректировки текста (рисунок 3):

1) входной текст разделяем на отдельные слова;
2) начинаем перебирать каждое слово;
3) сначала пробуем найти соответствие в базе данных без каких-либо изменений, чтобы не совершать лишних операций в случае, если слово изначально верное;
4) если находим слово в базе данных, добавляем в итоговую коллекцию;
5) если совпадения в базе данных по этому слову не найдено, выполняется алгоритм редакционного предписания, на выходе получаем набор возможных вариантов, по которым ищем кандидатов в базе данных слов;
6) если кандидаты не найдены, проверяем, была ли ранее смена раскладки, например, с английского на русский, казахский или наоборот. Если была, меняем раскладку, тем самым расширив диапазон поиска кандидатов на корректировку. Далее возвращаемся на предыдущий шаг, снова выполняем редакционное предписание и пытаемся получить кандидатов на корректировку;
7) если после применения алгоритма редакционного предписания количество кандидатов равно нулю, исходное слово возвращаем в качестве корректного слова, так как в базе данных отсутствует данное слово и кандидаты на его корректировку;
8) если кандидаты найдены и при этом количество кандидатов больше одного, возникает новая задача: определить, какой вариант из подобранных кандидатов подходит на корректировку. У данной задачи есть различные способы решения. В данной статье рассмотрим теорию вероятности, алгоритм расстояния Дамерау-Левенштейна;
9) по полученным результатам слова из коллекций конкатенируем и возвращаем в качестве предварительно обработанного текста для поисковой системы.

Сделать выбор из полученных кандидатов на корректировку — задача непростая. Данную проблему можно решить с помощью различных метрик и даже с помощью теории вероятности. Вопрос в том, какую цель преследовать: скорость или точность?

Анализ методов

Если сравнить теорему Байеса и расстояние Левенштейна, то отдать предпочтение какому-либо одному методу сложно. Попробуем сравнить точность и время выполнения (таблица 4).

Таблица 4 – Сравнение расстояния Левенштейна и теории вероятности в практическом применении на языке программирования python

Метрика Левенштейна	**Теория вероятности**
Text: кайрлы **Elapsed time:** 0.2219 seconds **Результат:** қайырлы	**Text:** кайрлы **Elapsed time:** 0.2248 seconds **Результат:** жайлы
Text: интеркт **Elapsed time:** 0.2866 seconds **Результат:** интернет	**Text:** интеркт **Elapsed time:** 0.2897 seconds **Результат:** интернет
Text: кайырлы кун алматы каласы **Elapsed time:** 0.0004 seconds **Результат:** қайырлы **Elapsed time:** 0.0002 seconds **Результат:** қайырлы күн **Elapsed time:** 0.0002 seconds **Результат:** қайырлы күн алматы **Elapsed time:** 0.0003 seconds **Результат:** қайырлы күн алматы баласы	**Text:** кайырлы кун алматы каласы **Elapsed time:** 0.0003 seconds **Результат:** қайырлы **Elapsed time:** 0.0002 seconds **Результат:** қайырлы күн **Elapsed time:** 0.0002 seconds **Результат:** қайырлы күн алматы **Elapsed time:** 0.0003 seconds **Результат:** қайырлы күн алматы баласы

Тестирования метода подбора кандидатов

Множество вариантов зависит от базы данных слов. Чем больше слов, тем больше вариантов, и тем сложнее определить точно, что имел в виду пользователь.

Рассмотрим проблемы обоих методов. Ниже представлен список из введенных для корректировки слов и их варианты. Например, для слова *карай* нашлось 5 кандидатов на казахском языке. По метрике расстояния Левенштейна, у всех вариантов метрика вычислила одинаковое расстояние «1 — на один символ каждый из них отличается от исходного». Выбор в данном случае будет очевидно случайным. Теорией вероятности в данном случае будет учитываться, насколько часто используется в базе данных это слово. Если частота выше, чем у остальных кандидатов, алгоритм подберет то слово, которое используется чаще всего.

Например варианты:

Карай - {'арай', 'қадай', 'жасай', 'асай', 'тарай', 'атай', 'жарай', 'маңай', 'санай', 'малай', 'шавай', 'тақай', 'калай', 'ламай', 'шалай', 'сағай', 'сарай', 'адай', 'апай', 'жанай', 'алай', 'надай', 'абай', 'кабак', 'талай', 'ағай', 'қарай'}

Сут - {'суы', 'сәт', 'сот', 'суыт', 'сат', 'су', 'сүт', 'вут'}

Калам - {'қалам', 'далам', 'балам', 'салам', 'шалам', 'алам'}

Кате - {'кете', 'катер', 'күте', 'қате', 'кафе'}

Каласы - {'қаласы', 'баласы', 'наласы'}

Карай - {'сарай', 'арай', 'тарай', 'қарай', 'жарай'}

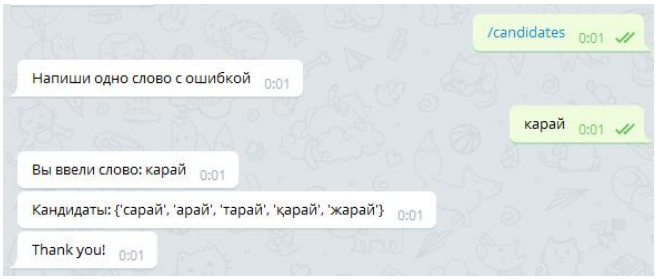

Рисунок 4 - Бот для корректировки текста (подбор кандидатов)

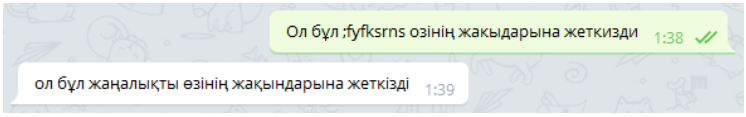

Рисунок 5 - Бот для корректировки текста (исправление предложения)

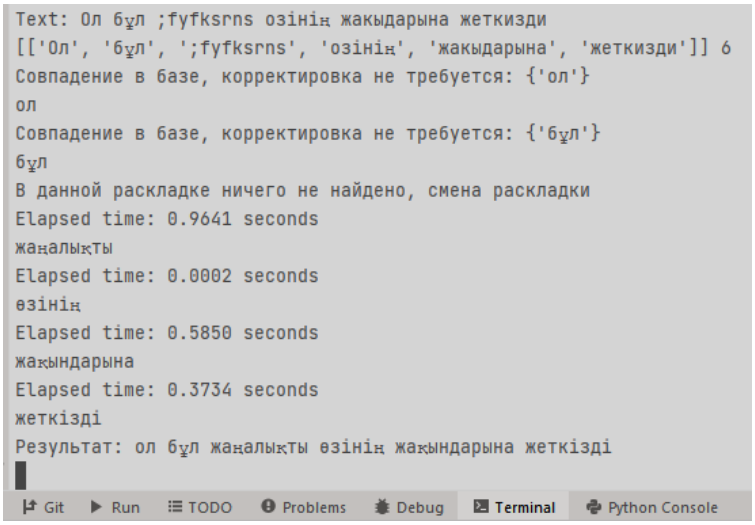

Рисунок 6 - Консоль и время выполнения корректировки каждого слова

Заключение

Проведенная работа

В данной статье были описаны основные подходы для решения задачи корректировки текста на казахском языке. В частности, расширение казахскими буквами редакционного предписания и его применение, расширение базы казахскими словами, анализ и сравнение метрики Левенштейна и теории вероятности, смена раскладки текста, демонстрация примера на блок-схеме и результатов экспериментов с казахскими словами и предложениями, написанными на языке python.

Результаты

После сравнения теории вероятности и расстояния Левенштейна можно понять, что расстояние Левенштейна дает более релевантный результат, чем теория вероятности. Однако, какую бы метрику не использовали для определения, разницей между словами *интернет* и *интернат* будет один символ, что делает определение достаточно сложным. Для улучшения точности дополнительно нужно учитывать контекст запроса. Соответственно, в базе данных нужно построить контекстные зависимости для слов.

Планы на будущее

Опечатки можно исправлять методами нечеткого поиска и линейным поиском [5]. На сегодняшний день не существует алгоритмов, которые смогли бы с точностью в 100 %

определить, какое слово имел в виду человек, но к этому нужно стремиться. Возможно, применение машинного обучения в сфере лингвистики поможет сделать определение того, что имел в виду человек в своем тексте, более точным.

Чаще всего применяются различные виды структур нейронных сетей для повышения точности распознавания, которые требуют сложного длительного обучения и значительной обучающей выборки [6]. Распознование некоторых текстов может быть ограничено по сложности, по алфавиту, по расположению, однако качество корректировки зависит от качества обучения нейронной сети [6]. Но есть и обратная сторона этой проблемы — скорость и цена процесса распознавания ошибок в тексте. Однако разработчики Яндекса показали на практике, применив CatBoost, разработанный их компанией, что использование машинного обучения увеличивает точность корректировки [7].

Для улучшения данного функционала как части поисковой системы можно применить OLAP (online analytical processing) по данным поисковых запросов людей, совместно с машинным обучением, а также нормализацию текста с помощью NLP.

В то же время база слов должна постоянно автоматизированно снабжаться новыми словами, так как появляются новые слова и сленг. Кроме того, нужно учитывать переход на латинский алфавит казахского языка, перевод базы данных казахских слов на латиницу для работы алгоритма, описанного в этой статье.

СПИСОК ЛИТЕРАТУРЫ

1. Бадалов М. И., Мирзамов А. М. Редакционное расстояние с подстановкой //toshkent shahridagi turin politexnika universiteti. – 2017. – С. 304.
2. Сидоркина И. Г., Килеев В. В. Кодировка символов переменной длины в алгоритме Дамерау–Левенштейна //Вестник Чувашского университета. – 2013. – №. 3.
3. Расстояние Левенштейна. [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Расстояние_Левенштейна. (дата обращения: 23.04.2021)
4. Раскладка клавиатуры. [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Раскладка_клавиатуры. (дата обращения: 23.04.2021)
5. Бондаренко А. О. Библиотека для исправления опечаток с учетом контекста: дис. – Сибирский федеральный университет, 2020.
6. Сапаров А. Ю., Бельтюков А. П., Маслов С. Г. Уточнение результатов распознавания математических формул с использованием расстояния Левенштейна //Вестник Удмуртского университета. Математика. Механика. Компьютерные науки. – 2020. – Т. 30. – №. 3. – С. 513-529.
7. Салахутдинова К. И., Лебедев И. С., Кривцова И. Е. Алгоритм градиентного бустинга деревьев решений в задаче идентификации программного обеспечения //Научно-технический вестник информационных технологий, механики и оптики. – 2018. – Т. 18. – №. 6.

REFERENCES

1. Badalov M. I., Mirzamov A. M. Redakcionnoe rasstoyanie s podstanovkoi //toshkent shahridagi turin politexnika universiteti. – 2017. – S. 304.
2. Sidorkina I. G., Kileev V. V. Kodirovka simvolov peremennoi dliny v algoritme Damerau–Levenshteina //Vestnik Chuvashskogo universiteta. – 2013. – №. 3.
3. Rasstoyanie Levenshtejna. [Electronic resource] URL: https://ru.wikipedia.org/wiki/Расстояние_Левенштейна. (date of the application: 23.04.2021)
4. Raskladka klaviatury. [Electronic resource] URL: https://ru.wikipedia.org/wiki/Раскладка_клавиатуры. (date of the application: 23.04.2021)
5. Bondarenko A. O. Biblioteka dlya ispravleniya opechatok s uchetom konteksta : dis. – Sibirskij federalnyi universitet, 2020.

6. Saparov A. U., Beltyukov A. P., Maslov S. G. Utochnenie rezultatov raspoznavaniya matematicheskih formul s ispolzovaniem rasstoyaniya Levenshteina //Vestnik Udmurtskogo universiteta. Matematika. Mekhanika. Komputernye nauki. – 2020. – T. 30. – №. 3. – S. 513-529.
7. Salahutdinova K. I., Lebedev I. S., Krivsova I. E. Algoritm gradientnogo bustinga dereviev reshenii v zadache identifikacii programmnogo obespecheniya //Nauchno-tekhnicheskii vestnik informacionnyh tekhnologii, mekhaniki i optiki. – 2018. – T. 18. – №. 6.

Бактаев А.Б., Мукажанов Н.К.

Қазақ тілін қолдайтын іздеу жүйелерінде қолданылатын мәтіндегі жаңылыстарды түзету бойынша есептерді шешу алгоритмі

Аңдатпа. Қазақ тілін қолдайтын, мақалада мәтінді енгізу кезінде сөздердегі қателермен жаңылыстарды түзету тәсілдері және алгоритмі сипатталған. Алгоритм Дамерау-Левенштейн қашықтығымен және іздеу жүйесіне арналған редакциялық нұсқаулық арқылы жасалады. Іздеудің дәлдігі мен өзектілігі үшін нақты негізгі сұраулар қажет. Асығыста адамдар пернетақта орналасуын ауыстыруды, теру кезінде орфографиялық немесе басқа да қателіктердің пайда болуы мүмкін, бірақ бұл ақпаратты іздеу сапасына әсер етуге жол бермеу керек.

Түйінді сөздер: Мәтіндегі қатені түзету, іздеу сұраулары, сұраулардағы қателер, Дамерау – Левенштейн қашықтығы, редакциялық нұсқаулық.

Baktayev A.B., Mukazhanov N.K.

Algorithm for solving the problem of correcting typos with search engines supporting the Kazakh language

Abstract. This article describes an approach to solving and applying an algorithm for correcting typos and spelling mistakes which people make when entering a text in the Kazakh language.

The algorithm is developed using the Damerau-Levenshtein distance and editorial prescription for search system (engine). An exact keyword (query) is required for more accurate and relevant search. In a hurry, people tend to forget to switch the keyboard layout, make spelling or other types of mistakes when entering a text, and this should not be allowed to affect the quality of information retrieval.

Keywords: correction of typos in text, search queries, query errors, Damerau-Levenshtein distance, editorial prescription.

Авторлар туралы мәлімет:

Бактаев Айдос Бакдаулетович, бакалавр, «Компьютерлік инженерия және ақпараттық қауіпсіздік» кафедрасының магистранты, Халықаралық ақпараттық технологиялар университеті.

Мукажанов Нуржан Какенович, PhD, «Компьютерлік инженерия және ақпараттық қауіпсіздік» кафедрасының ассистент-профессоры, Халықаралық ақпараттық технологиялар университеті.

Сведения об авторах:

Бактаев Айдос Бакдаулетович, бакалавр, магистрант кафедры «Компьютерная инженерия и информационная безопасность», Международный университет информационных технологий.

Мукажанов Нуржан Какенович, PhD, ассистент-профессор кафедры «Компьютерная инженерия и информационная безопасность», Международный университет информационных технологий.

About the authors:

Aidos B. Baktayev, master student, Department of Computer Engineering and Information Security, International Information Technology University.

Nurzhan K. Mukazhanov, PhD, Assistant-Professor, Department of Computer Engineering and Information Security, International Information Technology University.

INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES

МЕЖДУНАРОДНЫЙ ЖУРНАЛ ИНФОРМАЦИОННЫХ И КОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

ХАЛЫҚАРАЛЫҚ АҚПАРАТТЫҚ ЖӘНЕ КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАР ЖУРНАЛЫ

Ответственный за выпуск	Есбергенов Досым Бектенович
Редакторы	Далабаева Айсара Касымбековна Джоламанова Балия Джалгасбаевна Медведев Евгений Юрьевич
Компьютерная верстка	Туратауова Айжаркын Ахметовна
Компьютерный дизайн	Туратауова Айжаркын Ахметовна

Подписано в печать 26.06.2021 г.
Тираж 500 экз. Формат 60x84 1/16. Бумага тип.
Уч.-изд.л. 10.1. Заказ №165

Издание Международный университет информационных технологий
Издательский центр КБТУ, Алматы, ул. Толе би, 59