

## Оглавление

Введение .....	1
<b>Структура работы.....</b>	<b>2</b>
Глава I. Теоретические основания .....	4
<b>Неунифицированные геотеги .....</b>	<b>5</b>
<b>Цели и методы .....</b>	<b>6</b>
Глава II. Обзор проблемы.....	7
<b>Классификация неунифицированных геотегов .....</b>	<b>8</b>
Глава III. Алгоритм для автоматической унификации геотегов.....	11
<b>Транслитерация .....</b>	<b>11</b>
<b>Исправление орфографических ошибок (опечаток).....</b>	<b>13</b>
<b>Метод N-грамм .....</b>	<b>19</b>
<b>Выбор алгоритма .....</b>	<b>22</b>
<b>Словарь соответствий.....</b>	<b>22</b>
<b>Финальный алгоритм .....</b>	<b>23</b>
Заключение .....	25
Список источников и литературы .....	26
Приложение .....	28

## Введение

Региональная разметка является важной составляющей для многих больших корпусов текстов. Однако, большой объём выгружаемой информации создаёт необходимость её унификации. Унификация тех или иных данных – направление, связанное непосредственно с процессом обработки естественного языка. В данной работе рассматривается региональная разметка в Генеральном Интернет-Корпусе Русского Языка (ГИКРЯ).

*Актуальность* работы заключается в том, что на данный момент в сегменте структурных атрибутов ГИКРЯ многие существующие пользовательские локации не унифицированы, а значит не попадают в выборку по запросу местоположения.

*Цель работы* заключается в выявлении основных проблем и путей их решения, возникающих при работе с региональной разметкой в Генеральном Интернет-Корпусе Русского Языка (ГИКРЯ).

Для достижения поставленной цели формулируются следующие *задачи*:

- Описать существующий алгоритм присваивания тексту геотега.
- Описать основные проблемы, связанные с невозможностью соотнесения геотега с нужной локацией и, как следствие, потерей метки.
- Предложить пути решения указанных проблем для унификации нераспознанных локаций с высокой полнотой и точностью.

*Материалом*, на котором проводится исследование, служит Генеральный Интернет-корпус Русского Языка (ГИКРЯ), а именно – список всех существующих в нём пользовательских локаций.

*Гипотеза*, выдвинутая в рамках данной работы, заключается в возможности создания качественного алгоритма на основе словарей и метрик, таблиц соотношения.

*Новизна* работы обусловлена малой изученностью рассматриваемой

области, отсутствием общепринятых словарей и правил для унификации геотегов, а также непосредственно самим материалом исследования, который представляет собой актуальные на сегодняшний день пользовательские названия мест проживания.

Процесс исследования осуществлялся *поэтапно*:

1. С сервера ГИКРЯ была произведена выгрузка всех пользовательских названий для местоположений
2. Производилась ручная разметка полученного списка:  
унифицированные локации отделялись от тех, которым была присвоена метка NA (Not Available)
3. Выявлялся некоторый ряд проблем, связанных с возможностью унификации таких тегов.
4. Рассматривались пути решения этих проблем
5. Создавались таблицы соответствий для унифицированных и неунифицированных вариантов.

### Структура работы

Работа состоит из трех глав, введения, заключения, приложений и списка литературы. Первая глава посвящена рассмотрению теоретических оснований для данной работы: структуры ГИКРЯ и значимости региональной разметки. Вводятся такие понятия как «унифицированные» и «неунифицированные» геотеги. Во второй главе представлен общий взгляд на проблему, список неунифицированных геотегов систематизируется по общим признакам, вычисляются такие параметры, как общее число вариантов пользовательских названий и количество их вхождений. В третьей главе поэтапно описывается алгоритм решения проблем унификации геотегов. Внимание акцентируется на методах, используемых для унификации, а именно на написании правил транслитерации, возможных вариантах системы проверки орфографии и таблиц соотношения собственно пользовательских

геотегов и их синонимов.

## Глава I. Теоретические основания

### О проекте Генерального Интернет-корпуса Русского языка

Генеральный Интернет-корпус Русского Языка (ГИКРЯ) – мегакорпус, созданный при помощи полностью автоматической технологии сбора и разметки текстов из Рунета и основанный на современных достижениях компьютерной лингвистики. По состоянию на май 2016 года корпус включает в себя материалы крупнейших ресурсов Рунета: Новостей, ВКонтакте, Живого Журнала, Блогов Мейл.ру, — а также Журнального Зала. Объём корпуса: 20 миллиардов слов, из них 18,6 миллиардов — из социальных сетей.

Помимо собственно лингвистической разметки в ГИКРЯ осуществляется метатекстовая разметка. Для каждого текста из соцсетей хранятся время и место его написания, URL, интернет-жанр (блог, новости и т.д.), а также год, место рождения автора, пол автора.

**Таблица состава данных**

Ресурс	URL	Имя автора	Год рождения	Пол	Родной город	Город нахождения	Год написания текста
ВК	-	-	+	+	+	+	+
ЖЖ	+	-	+	+	-	+	+
ЖЗ	+	-	-	+	-	-	+
БМР	+	-	+	+	-	+	+
<b>Новости:</b>							
Лента	+	-	-	-	-	-	+
Росбалт	+	-	-	-	-	-	+
Риа	+	-	-	-	-	-	+

В таблице состава данных видно, что такой метаатрибут, как город нахождения автора, присваивается текстам из социальных сетей: ВКонтакте, Живого Журнала, Блогов Мейл.ру.

**Объектом** исследования в данной работе является региональная метатекстовая разметка ГИКРЯ. Она необходима для лексикографических, социолингвистических, диалектологических исследований. К примеру, для изучения региональной вариативности той или иной лексемы.

### Неунифицированные геотеги

Большой размер корпуса, хоть и позволяет получать статистические данные о языке, порождает в свою очередь несколько проблем. Первая из них заключается в том, что помимо просто сбора, при построении корпуса необходимо провести некоторую обработку данных. Рассмотрим далее обработку данных, связанную с метаатрибутами.

При выгрузке текстов из соцсетей вся метайнформация, в том числе и геолокация автора собирается автоматически со страниц сайтов, сделанных по одному и тому же шаблону. Здесь стоит отметить, что под «геолокацией» автора подразумевается то, что он написал в графе «местоположение»/«регион» на своей странице в том или ином ресурсе, откуда была произведена выгрузка текста. При попадании текста из соцсети «ВКонтакте» в графу попадает что-то одно: или город нахождения или родной город. По умолчанию это – город нахождения, но если он не указан, то родной город. Однако, здесь исследователь сталкивается со следующей проблемой: многие авторы указывают некорректное местоположение. Используют перифраз (общепринятый или употребляемый в определенных кругах), сокращения; пишут названия населённых пунктов по-английски и/или с ошибками. Все эти варианты остаются не унифицированы и оттого не попадают в выборку запроса по локации. Назовём такие некорректные данные **неунифицированными геотегами**. Они и будут **предметом** данного исследования.

Таким образом, подобные локации могут серьёзно повлиять на результаты лексикологического или социолингвистического исследования, использующего региональную разметку. Более того, неунифицированные данные создают огромный «хвост» на гистограмме частотности всех локаций. При этом, полностью лишать тексты таких важных тегов не кажется целесообразным.

## Цели и методы

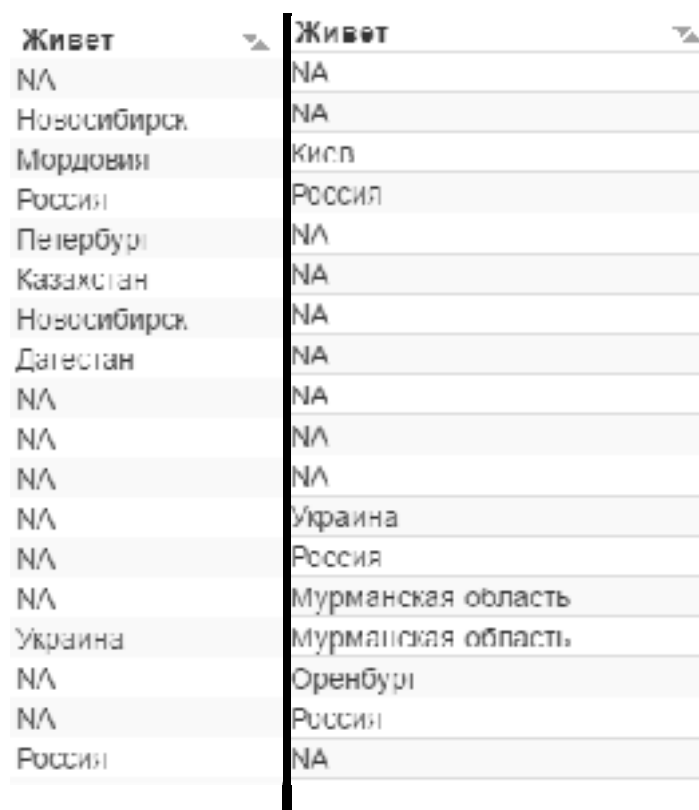
В данной работе я ставлю своей **целью** проанализировать все варианты пользовательских названий местоположений и составить правила автоматического переформулирования неунифицированных геотегов.

Описания цельного алгоритма для выполнения данной задачи до этого не было представлено. Хотя поисковые системы некоторых сайтов справляются с задачей поиска по метке, не входящей в стандартный список регионов.

Например, в социальной сети «ВКонтакте» при поиске «ЕКБ» будут показаны пользователи, указавшие в своём профайле город проживания Екатеринбург.

## Глава II. Обзор проблемы

Первым этапом моей работы стал обзор всех существующих пользовательский названий местоположений. С сервера ГИКРЯ была произведена выгрузка всех существующих названий местоположений, указанных пользователями в соцсетях. Всего таких локаций было 58304. В том случае, когда региональную информацию не удаётся извлечь автоматически, атрибуту «loc» присваивается метка NA (not available). Унифицированными будем считать названия местоположений, которые можно найти в словаре. В своей работе я использую список регионов на русском языке во всех падежах из словаря ABBYY COMPRENO. Мною был проанализирован весь список. Затем – геотеги, которым была присвоена метка NA. Основная работа была проделана именно с этим списком.



Живёт	Живёт
НА	НА
Новосибирск	НА
Мордовия	Кипр
Россия	Россия
Петербург	НА
Казахстан	НА
Новосибирск	НА
Дагестан	НА
НА	НА
НА	НА
НА	НА
НА	Украина
НА	Россия
НА	Мурманская область
Украина	Мурманская область
НА	Оренбург
НА	Россия
Россия	НА

Рисунок 1

Рисунок 2

На рисунке 1 примеры меток местоположения автора в выдаче ГИКРЯ по запросу «сладкий апельсин», на рисунке 2 – примеры меток местоположения



автора в выдаче ГИКРЯ по запросу «жимолость». Как мы видим, больше половины тегов в случайной выборке имеют отметку NA.

Пользовательские названия, успешно присвоенные атрибуту «loc» в поиске на материале ГИКРЯ, далее не будут рассматриваться. Хотя и здесь есть свои задачи. Например, необходима иерархия локаций по административным единицам (населённый пункт, регион, страна). Это даст пользователю возможность поиска по локациям любого административного уровня. Оставим эту тему для другого исследования.

### **Классификация неунифицированных геотегов**

Оставшиеся неунифицированные геотеги было решено систематизировать по общим признакам. Эта работа была проведена вручную. Каждая метка из списка неунифицированных отмечалась в зависимости от того, что мешало ей быть найденной в словаре.

Причины тому могли быть следующие:

- 1) Пользователь соцсети указал название города не на русском языке:

*Moscow, Moskau*

Так как программа, выгружающая данные со страниц пользователей в корпус, распознает только латиницу и кириллицу, все подобные варианты были на латинской раскладке. В этот же пункт следует отнести и латинскую транслитерацию:

*Moskva, Sankt-Petersburg*

Так как используемый “эталонный” словарь был на русском языке, такие метки не могли быть обнаружены. Подобные названия встречаются в 75067 употреблениях.

- 2) В названии региона присутствует орфографическая ошибка:

*масква, калиниград, екатерибург + г. новосибирск*

- 3) Пользователь использовал общепринятое (но не на официальном уровне) сокращение названия своего региона:

*МСК, НСК, СПб, НН*

- 4) Метка локации содержит альтернативные названия или перифраз для обозначения региона:

*Город на Неве, Столица, Питер*

- 5) В поле местоположения пользователь указал нечто, не поддающееся определению как локация. Например, цифры и другие небуквенные символы:

*2:5020, 7034, #####*

Здесь следует отметить, что в некоторых случаях использование цифр в геотегах можно установить как местоположение, а именно в ситуациях, когда эти цифры — автомобильные коды регионов России. Но понять, является ли тот или иной набор цифр зашифрованным в автомобильном коде названием региона представляется возможным только для самых частотных из подобных употреблений или для локаций, в которых отдельно отмечен этот факт: *10 region*

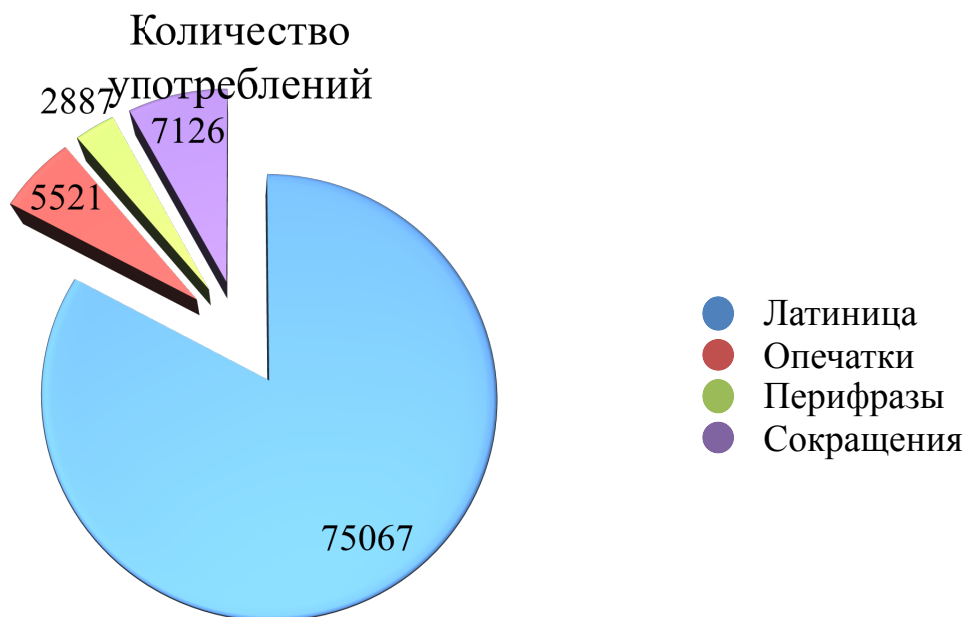
Помимо этого, достаточно часто пользователь будто “не определился” с собственным местоположением и указал сразу несколько регионов в одном поле:

*Новосибирск-Бийск, Москва\_Рязань,*

Так как нельзя точно установить, в каком именно из указанных городов находится автор, унификация подобных локаций не может быть произведена. Отнесённые к этому пункту варианты локаций представляют собой большинство из всего списка неунифицированных, но имеют наименьшее количество вхождений: по 1-2 употребления на слово. Так, локации с числом вхождений, равным 1 составляют абсолютное большинство в общем списке. Их количество — 40256 (при общем числе локаций в 58304).

Итак, варианты локаций из пункта 5 не могут быть чётко идентифицированы, поэтому исключаются из дальнейшего анализа.

Ниже представлен график соотношения разных типов оставшихся неунифицированных российских геотегов по количеству употреблений.



Как видно из диаграммы, наибольшее количество употреблений приходится на транслитерированные варианты написания населённого пункта. А именно, 75067 употреблений, что составляет 82,88% от общего числа всех выбранных для дальнейшего анализа вхождений. На втором месте сокращения — 7126 употреблений или 7,85%. Затем следуют варианты с орфографическими ошибками: 5521 употребление — 6,08%. Реже всего можно встретить перифраз: 2887 вхождений или 3,181%.

## Глава III. Алгоритм для автоматической унификации геотегов

### Транслитерация

Как было показано в предыдущей главе, наибольший процент неунифицированных геотегов приходится на варианты написания локации на латинице. Отсюда следует вывод, что первым шагом в разработке алгоритма для их унификации должна стать программа транслитерации.

Транслитерация — перевод одной графической системы алфавита в другую, то есть передача букв одной письменности буквами другой. Компьютерная транслитерация — способ автоматического преобразования слов в одном языке в их фонетические эквиваленты на другом языке [Oh, Choi, Isahara, 2006] .

Большинство подходов к проблеме транслитерации стремятся смоделировать преобразования одновременно на графическом и фонетическом уровне.

[Knight and Graehl: 1997; Stalls and Knight: 1998; Al-Onaizan and Knight: 2002]

Один из таких подходов и был выбран мной для преобразования геотегов, написанных на латинице. Он основывается на применении таблиц замены знаков латинской системы письма знаками кириллической системы письма [Knight and Graehl: 1997]. В качестве таблицы соотношений был выбран ГОСТ 16876-71.

Кириллица ▼	Латиница ▼
а	a
б	b
в	v
г	g
д	d
е	e
ё	jo
ж	zh
з	z
и	i
й	j
к	k
л	l
м	m
н	n
о	o
п	p
р	r
с	s
т	t
у	u
ф	f
х	kh
ц	c
ч	ch
ш	sh
щ	shh
ъ	"
ы	y
ь	'
э	eh
ю	ju
я	ja

## Исправление орфографических ошибок (опечаток)

После того, как все подлежащие унификации геотеги представлены в кириллице, мы можем снова сравнивать их с эталонным словарём для выявления соответствий. Но и после этого шага не все полные соответствия могут быть обнаружены. Например, после транслитерации тега *Moscau* на кириллицу с помощью алгоритма, описанного в предыдущем параграфе, получаем вариант *Мосцау*. Помимо этого, в нашем списке есть 189 вариантов пользовательских названий с общим числом употреблений 5521, в которых допущена орфографическая ошибка. Перед нами встаёт необходимость исправления этой ошибки.

В широком смысле выявлением орфографических ошибок и поиском возможных вариантов их исправления занимаются системы исправления опечаток (spell-checkers). Согласно работе [Kukich: 1992] можно выделить следующие проблемы, связанные с исправлением опечаток:

- 1) Выявление несуществующих слов: обнаружение орфографических ошибок, которые приводят к несуществующим словам.
- 2) Исправление таких слов изолированно, без учёта контекста.
- 3) Контекстно-зависимое обнаружение и исправление ошибок.

Автоматически сложно определить ошибку, если в результате ошибки/опечатки или её машинного исправления получилось слово, которое можно найти в словаре, но не передающее значение, которое имел в виду автор. Здесь, безусловно, нужно обращаться к контексту.

Выявление несуществующих слов обычно происходит путём поиска близких слов по словарю. Ранее предполагалось, что такие словари для проверки должны иметь небольшой объём, так как большие словари содержат в том числе и очень редкие слова. Те, в свою очередь, могут быть похожи на слово с ошибкой. Например, ошибки в написании английских слов *very* и *won't* могут распознаться как редкие слова *veery* («бурый короткоклювый дрозд») и *wont* («обыкновение, привычка»). Практика показывает, что опечатки

действительно могут «спрятаться» за редкими словами. Однако, большие словари приносят больше пользы, нежели вреда. С помощью маркирования редких слов можно избежать принятия за них ошибки. [Damerau, Mays, 1989] Это особенно актуально для вероятностных систем проверки орфографии, учитывающих частотность слова. Таким образом, современные программы проверки правописания используют словари большого объёма.

Вернёмся к нашему списку геотегов, содержащих ошибку. Проанализировав весь список можно понять, что все варианты имеют однозначное исправление, не зависящее от словарного окружения, это даёт нам право предполагать, что они могут быть исправлены в автоматическом режиме.

Сначала следует определить, какие типы ошибок встречаются в пользовательских локациях, чтобы понять, с помощью какого алгоритма автоматически их находить и исправлять.

Почти 80% ошибок правописания можно разделить на ошибки вставки, удаления, замены или перестановки символов, а также комбинации этих типов ошибок [Damerau, 1964].

Вернёмся к нашему списку геотегов, неунифицированных из-за наличия опечатки в написании. Все слова из этого списка содержат ту или иную ошибку из приведённых выше типов. А именно:

- 1) Вставка: «лишние» буквы или символы в названии населённого пункта, а так же пометка «г.» («город») рядом с этим названием:

*Массква* (14 употреблений), *Донецьк* (29 употреблений) *г. Магадан* (13 употреблений) *г. Барнаул* (11 употреблений)

- 2) Удаление: отсутствие нужных букв или символов:

*Санкт Петербург* (33 употребления), *Санкт-Петерург* (8 употреблений) *Моска* (11 употреблений), *Калиниград* (12 употреблений)

3) Замена нужных букв или символов на другие:

*Москоу* (89 употреблений), *Мозгва* (12 употреблений), \

4) Перестановка букв:

*Новосибиркс* (4 употребления)

5) Сочетание ошибок типов 1-4:

*Маськва* (18 употреблений), *г.махочкала* (7 употреблений), *Иркуцк* (7 употреблений)

В рассматриваемом списке не встречалось больше трёх ошибок в одном слове. Но следует учесть, что корпус постоянно пополняется новыми текстами, а значит и новыми локациями. Теоретически, в них может быть совершено и большее количество ошибок.

Итак, нам требуется корректор, который может находить и исправлять 3 (и более) ошибки представленных типов.

Существует несколько подходов к созданию систем автоматической проверки орфографии. Один из самых популярных — решение Питера Норвига [Norvig, 2007]:

- 1) Слово проверяется по словарю
- 2) Если совпадений не обнаружено, то генерируются все однобуквенные ошибки
- 3) Проверка по словарю
- 4) Если совпадения вновь не выявлены, генерируются двубуквенные ошибки
- 5) Проверка по словарю
- 6) Если совпадений не обнаружено, повторение цикла с добавлением букв в ошибках



Под генерацией ошибок здесь понимается следующее: для проверяемого слова создается набор слов, которые получаются применением уже описанных ранее 4-х операций: удаление, вставка, замена и перестановка. К примеру, ошибки замены 'а' на 'о' в слове 'масква' дадут следующие слова: *оасква, москва, маоква, масова, маскоа, маскво*. Ясно, что такое делается для всех букв алфавита. Подобным образом генерируются варианты слов для вставки, удаления и перестановки.

Разумеется, при таком подходе нельзя гарантировать полного исправления всех ошибок. Например, возвращаясь к списку геотегов, если нам дано слово *оск*, то правильным будет слово «омск» или «орск»? Именно поэтому на следующем шаге применяется вероятностный (или стохастический) подход. Будем считать, что мы пытаемся выбрать такое слово  $c$  из всех возможных слов-исправлений так, что вероятность появления именно слова  $c$  при данном слове  $w$  будет максимальна:

$$\operatorname{argmax}_c P(c|w)$$

Согласно теореме Байеса выражение, записанное выше, эквивалентно следующему выражению:

$$\operatorname{argmax}_c P(w|c) P(c) / P(w)$$

Поскольку  $P(w)$  одинакова для всех  $c$  мы можем отбросить  $P(w)$ , что даст нам:

$$\operatorname{argmax}_c P(w|c) P(c)$$

Здесь:

$P(c)$  – вероятность появления слова  $c$  (частотность употребления  $c$ ). Эта вероятность обусловлена самим языком (точнее моделью языка). Иначе говоря,  $P(c)$  определяет как часто  $c$  встречается в текстах на русском языке.  $P(\text{«привет»})$  будет достаточно высока, тогда как

$P(\text{«элеутерокок»})$  будет меньше, а  $P(\text{«шнвроправ»})$  будет около нуля.

$P(w|c)$  – вероятность того, что автор опечатался и написал  $w$ , хотя имел в виду  $c$ . Эта вероятность обусловлена частотностью тех или иных ошибок в языке (и называется моделью ошибок языка).

$\text{argmax}_c$  – оператор, осуществляющий поиск такого допустимого  $c$ , которое бы максимизировало условную вероятность появления  $w$  при данном  $c$ .

Такой алгоритм достаточно прост в осуществлении. В оригинальной статье [Norvig, 2007] приведена его реализация на языке Python в 21 строку кода.

Время работы данного алгоритма зависит от числа  $k$  ошибок и от размера алфавита  $A$ , и в случае использования бинарного поиска по словарю составляет:

$$O((m|A|)^k \cdot m \cdot \log n)$$

Например, при  $k = 1$  и слова длиной в 7 букв (например, «Барнаул») в русском алфавите множество ошибочных слов будет размером около 450, то есть будет необходимо сделать 450 запросов к словарю.

Но уже при  $k = 2$  размер такого множества будет составлять более 115 тысяч вариантов. При этом не нужно забывать еще и о том, что для каждого из таких слов необходимо провести поиск на точное совпадение в словаре. А также необходимо построение модели ошибок языка, подходящей для нашего контекста – российских географических локаций. Таким образом, такой подход хоть и применяется во многих поисковых системах, для нашей задачи кажется не самым подходящим.

Как было сказано выше, для того, чтобы выявить ошибку, необходимо сравнивать слова со словарём. Этот шаг уже проделывался в пункте 1, когда нужно было отделить унифицированные геотеги от неунифицированных, эта же идея реализована в алгоритме Норвига. Но в этих случаях поиск основывается на полном соответствии. Когда речь идёт об алгоритмах проверки правописания, возникает идея искать также «похожие» слова. Эта идея осуществляется в так называемых системах нечёткого поиска. Задача таких систем может быть сформулирована следующим образом: по

заданному слову найти в тексте или словаре размера **n** все слова, совпадающие с этим словом (или начинающиеся с этого слова) с учетом **k** возможных различий. Что значит «похожие слова»? Для выявления схожести слов необходимо ввести некоторую метрику (функцию, определяющую расстояние в метрическом пространстве). Основными метриками для такого параметра являются:

- Расстояние Хэмминга
- Расстояние Левенштейна
- Расстояние Дамерау-Левенштейна
- Метод N-грамм

Рассмотрим каждую из них подробнее.

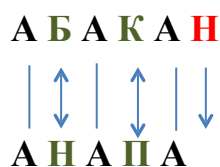
**Расстояние Хэмминга** – число позиций, в которых соответствующие символы двух слов одинаковой длины различны. В более общем случае расстояние Хэмминга применяется для строк одинаковой длины любых  $q$ -ичных алфавитов [Norouzi et al., 2012a].

Очевидно, что слова из нашего списка, ошибки в которых нужно исправить, состоят из разного количества символов. Так как расстояние Хэмминга служит метрикой различия объектов одинаковой размерности, его применение для системы проверки правописания российских географических локаций невозможно.

**Расстояние Левенштейна** — это минимальное количество операций редактирования, которые нужно применить к первой строке чтобы получить вторую. В классическом варианте рассматриваются три операции: вставка, удаление и замена.[Левенштейн, 1965]

**Расстояние Дамерау-Левенштейна** модификация классического расстояния Левенштейна, в которой к операциям вставки, удаления и замены символов, определенных в расстоянии Левенштейна добавлена операция перестановки символов (транспозиция). [Levenshtein, 1965; Damerau, 1964]

Например, чтобы получить из слова *Абакан* слово *Анапа* необходимо совершить две операции замены и одну операцию удаления:



Вычисление определённого таким образом расстояния редактирования может быть произведено с помощью метода, использующего динамическое программирование [Wagner. Fisher, 1974]. Алгоритм имеет сложность  $O(MN)$ , где  $M$  и  $N$  – длины сравниваемых строк, а для нахождения значения расстояния требуется вычислить  $MN$  элементов так называемой матрицы динамического программирования. Сложность вычисления расстояния Левенштейна–Дамерау имеет квадратичный порядок относительно размера строк. [Бойцов Л.М. , 2004]. Значительное количество работ посвящено созданию более эффективных алгоритмов. Предложенные процедуры вычисления можно условно разделить на две категории. Первую категорию составляют алгоритмы, использующие метод отсечений [E. Ukkonen , 1985]. В их основе лежит алгоритм динамического программирования, но для определения расстояния редактирования не требуется вычислять все  $MN$  элементов матрицы. Вторую категорию составляют алгоритмы, основанные на эффективном использовании битовых операций [E.W. Myers, 1998].

### Метод N-грамм

Сравниваемые строки разделяются на подстроки длины  $N$  ( $N$ -граммы), затем осуществляется сравнение наборов подстрок. Исходя из количества совпавших подстрок, можно сделать выводы об их похожести или непохожести.

Данный метод основан на предположении, что при малых искажениях слова должны обладать достаточным количеством общих  $N$ -грамм. Суть метода заключается в следующем: если строка  $V$  «похожа» на строку  $U$ , то у них должны быть какие-либо общие подстроки. Слова в словаре разбиваются на такие  $N$ -граммы. Неунифицированный геотег также разбивается на  $N$ -

граммы, и для каждой из них производится последовательный перебор списка слов, содержащих такую подстроку. [J. R. Ullmann, 1977]

Критериями совпадения в данном случае могут служить следующие величины:

- Количество совпавших подстрок с учетом длин сравниваемых строк;
- Количество совпавших подстрок с учетом их позиций в сравниваемых строках;
- Количество совпавших подстрок с учетом их позиций в сравниваемых строках и длин сравниваемых строк;
- Количество совпавших подстрок с учетом допустимого интервала отклонения их позиций в сравниваемых строках и допустимого отклонения длин сравниваемых строк.

Функция сравнения составляет все возможные комбинации подстрок с длиной вплоть до указанной  $N$  и подсчитывает их совпадения. Число совпадений, разделённое на число вариантов – коэффициент схожести строк для фиксированного  $N$  и выдаёт в качестве результата работы функции, затем берётся среднее значение для всех коэффициентов.

Где ;

$\text{len}(S)$  – длина строки  $S$ ;

$\text{Match}(S_1, S_2, i)$  – сумма совпадений всех подстрок длиной  $i$  из  $S_1$  в строке  $S_2$

Наиболее оптимальным считается деление на подстроки длины  $N = 2$  (биграммы). Рассмотрим пример, где в качестве аргументов заданы две строки *Республика Татарстан* и *Татарстан Республ.* и  $N=2$

	До нормализации	После нормализации	Длина
Эталон	Республика Татарстан	РЕСПУБЛИКА ТАТАРСТАН	20

Рабочая строка	Татарстан Республ.	ТАТАРСТАН РЕСПУБЛ	17
----------------	--------------------	-------------------	----

Здесь нормализация – удаление небуквенных символов и капитализация обеих строк.

Эталон	Рабочая строка
РЕ	ТА
ЕС	АТ
СП	ТА
ПУ	АР
УБ	РС
БЛ	СТ
ЛИ	ТА
ИК	АН
КА	Н
А	Р
Т	РЕ
ТА	ЕС
АТ	СП
ТА	ПУ
АР	УБ
РС	БЛ
СТ	
ТА	
АН	

Итак, при  $i=1$

$r=17/20=0.85$ ,

$i=2$

$$r = 14/19 = 0.73$$

$R = 0.79 = 79\%$  - достаточно высокий результат.

### **Выбор алгоритма**

Таким образом, были рассмотрены основные подходы к созданию системы проверки орфографии: решение Питера Норвига, представляющее собой статистическую языковую модель на основе больших баз данных; системы нечёткого поиска, использующие метрики (расстояние Хэмминга, расстояние Левенштейна, расстояние Дамерау-Левенштейна, Метод N-грамм). В нашем частном случае разработки алгоритма унификации геотегов модель Норвига не кажется подходящей. Для осуществления системы нечёткого поиска самыми подходящими метриками стали расстояние Дамерау-Левенштейна и метод N-грамм. Дальнейший выбор алгоритма должен осуществляться на основе практического сравнительного анализа эффективности этих двух подходов и в данной работе представлен не будет.

### **Словарь соответствий**

Следующий шаг на пути к унификации геотегов – решения вопроса о сокращениях и перифразах. Таких вариантов пользовательских названий в существующей базе – 336 с общим числом вхождений 10013. Эти названия не могут быть идентифицированы по «схожести». Значит, перед нами встаёт задача о создании таблицы соответствий «неунифицированный геотег» – «унифицированный геотег». Стоит отметить, что в имеющемся списке сокращений и вариативных названий «унифицированными метками» будут города с большим населением (начиная от 500 000 человек). Чем больше населённый пункт, тем больше и вариантов его неунифицированного названия. При этом среди двух самых крупных регионов – Москвы и Санкт-Петербурга по числу возможных названий «побеждает» всё же Петербург: для его обозначения пользователи используют 72 различные метки, тогда как для Москвы – 54.

Для начала вручную было проделано составление списка для самых частотных сокращений. Функция переводит строку в нижний регистр, происходит транслитерация по алгоритму, описанному в начале главы, и удаляет все небуквенные символы – пробелы, точки, дефисы. Таким образом, между унифицированным и неунифицированным вариантом есть промежуточный, что уменьшает количество итераций при поиске соответствий. Из самых частотных 99 неунифицированных сокращений осталось 30 промежуточных, переходящих всего в 11 «правильных». В приложении (1) представлена таблица таких сокращений.

Похожий вид работы был проделан и для других устоявшихся названий того или иного населённого пункта. Но здесь уже не учитывался возможный промежуточный вариант.

### **Финальный алгоритм**

В данной главе были предложены решения для каждой конкретной проблемы невозможности унификации того или иного геотега.

Объединение каждого пункта даёт нам финальный алгоритм унификации геотегов в ГИКРЯ:

1. Информация, указанная пользователем соцсети в графе «местоположение» и автоматически выгружаемая в корпус, проверяется по словарю. В данной работе предлагается использовать словарь всех названий городов и регионов ABBYY COMPRENO из-за его полноты и содержательности.
2. Если метка написана на латинице, выполняется алгоритм транслитерации на кириллицу, описанный выше.
3. Если обнаружено соответствие, метка становится унифицированной, а значит присваивается тексту.



4. Если совпадений не обнаружено, выполняется программа исправления правописания, основанная на алгоритме «нечёткого поиска» - N-граммной модели или вычисления расстояния Дамерау-Левенштейна.
5. После исправления возможных ошибок метка снова проверяется в эталонном словаре.
6. Если обнаружено соответствие, метка становится унифицированной, а значит присваивается тексту.
7. Если совпадения не выявлены, то метка ищется в таблице возможных сокращений.
8. Если обнаружено соответствие, метка становится унифицированной, а значит присваивается тексту.
9. Если совпадений не обнаружено, метка сравнивается в таблице возможных вариантов названий.
10. Если обнаружено соответствие, метка становится унифицированной, а значит присваивается тексту.
11. Если совпадений не обнаружено, метке присваивается статус NA

## Заключение

Унификация геотегов – важная задача как для ГИКРЯ, так и для других систем, где учитывается местоположение пользователя. В данной работе был проделан анализ всех существующих пользовательских названий для местоположений, на основе этого анализа были классифицированы признаки, по которым геотег не может унифицироваться. В третьей главе были предложены правила для автоматической унификации неопределённых локаций, базирующиеся на признаках, выделенных во второй главе.

В ходе выполнения практической части работы для решения возникших вышеперечисленных сложностей при унификации геотегов в ГИКРЯ были составлены таблицы соотношений, позволяющие искать и соотносить вариативные названия для регионов с унифицированными вариантами. Стоит отметить, что предложенные правила учитывают постоянную пополняемость ГИКРЯ новыми текстами с новыми геотегами. Таблицы сокращений благодаря промежуточным вариантам учитывают максимальное количество возможных пользовательских названий. Таблицы вариативных названий при необходимости могут дополняться новыми значениями. В итоге была доказана выдвинутая в самом начале работы гипотеза, которая заключалась в возможности создания алгоритма на основе словарей и метрик.

Перспективами данной работы можно считать оценку практической применимости предложенных правил, экспериментальное доказательство выбора наиболее подходящего алгоритма для системы проверки орфографии с учётом специфики контекста географических названий, создание алгоритма, вводящего иерархическую структуру пользовательских локаций.

## Список источников и литературы

1. Jong-Hoon Oh, Hitoshi Isahara. 2006. Mining the Web for Transliteration Lexicons: Joint-Validation Approach. *IEEE/WIC/ACM Intl. Conf. on Web Intelligence*. 254–261.
2. K. Knight and J. Graehl. 1997. Machine Transliteration *Proc. of the Conference of the Association for Computational Linguistics (ACL)*. 54–76.
3. Y. Al-Onaizan, K. Knight. 2002 Named Entity Translation: Extended Abstract. *Proc. HLT*. 14–26
4. Kukich, K. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys (CSUR)*. 24. 12–18
5. Damerau F. J. 1964 A technique for computer detection and correction of spelling errors. *Communications of the Association for Computing Machinery*. Vol. 7, No. 3. 171–176.
6. Damerau F. J., Mays E. 1989 An examination of undetected typing errors. *Information Processing and Management*. Vol. 25, No. 6. 659–664.
7. Peter Norvig. 2007. How to Write a Spelling Corrector. <http://norvig.com/spell-correct.html> (26.04.16)
8. Mohammad Norouzi, David J. Fleet, and Ruslan Salakhutdinov. 2012. Hamming Distance Metric Learning. *Advances in Neural Information Processing Systems (NIPS)*. 25. 1070–1078.
9. В. И. Левенштейн. 1965 Двоичные коды с исправлением выпадений, вставок и замещений символов. *Доклады Академий Наук СССР*. 163.4:845–848.
10. R.A. Wagner and M.J. Fisher. 1974. The String to String Correction Problem. *Journal of the ACM*. 21(1). 168–173.

11. Байтин. 2004. Классификация и экспериментальное исследование современных алгоритмов нечеткого словарного поиска. *Труды конференции RCDL*. 89–102.
12. E. Ukkonen. 1985. Finding approximate patterns in strings,  $O(k * n)$  time. *Journal of Algorithms*. Volume 6. 132-137.
13. E.W. Myers. 1998. A Fast Bit-Vector Algorithm for Approximate String Matching Based on Dynamic Programming. *Journal of the ACM (JACM)*. Volume 46(3). 395 – 415.
14. J. R. Ullmann. 1977. A binary  $n$ -gram technique for automatic correction of substitution, deletion, insertion and reversal errors in words. *Computer Journal*. 20(2).141-147.
- 15.— ГОСТ 16876-71 Правила транслитерации букв кирилловского алфавита буквами латинского алфавита. Переиздание январь 1981 г. с изменениями 1 и 2, утвержденными в декабре 1973 г. и в марте 1980 г., — М.: Издательство стандартов, 1981.

## Приложение

Таблица унификации сокращений

Неунифицированный вариант	Промежуточный вариант	Унифицированный вариант
Питер	питер	Санкт-Петербург
питер		
Piter		
piter		
Петербург	петербург	
СПб		
Спб		
СПБ		
С-Пб		
спб		
СПб		
Спб		

СПБ	спб	
СПб.		
С-Пб.		
с-пб		
SPb		
Spb		
spb		
SPB		
S-Pb		
С-Пб		
С-Петербург	спетербург	
С.-Петербург		
С-Петербург		
С.Петербург		

С. Петербург		
St-Pete	стпете	
Saint Peters	сэинтпетерс	
Saint-P.	сЭИНТП	
Saint-P		
Saint-P		
St.-Petersbourg	стпетерсбург	
St.Petersbourg		
St.P	стп	
St-P		
Мск		Москва
мск		
МСК		
МСК		

Mck.	MCK	
Msk		
msk		
MSK		
MsK		
msc	MCЦ	
Msc		
Mck	MЦK	
MO	MO	
MOW	MOY	
MOW		
Mow		
Mos	MOC	
mscw	MCЦB	



Н.Новгород	нновгород	Нижний Новгород
Н. Новгород		
NNovgorod		
НН	нн	
Н.Н.		
NN		
Екб	екб	Екатеринбург
ЕКБ		
ЕКВ		
ekb		
Ekb		
ЕКВ		
екб		
Е-бург		

Ебург	ебург	
E-burg		
Eburg		
Ё-бург	ёбург	
Ектб	ектб	
Н-ск	НСК	Новосибирск
Нск		
НСК		
НСК		
Nsk		
nsk		
NSK		
N-sk		
Новосиб	новосиб	

Novosib	НОВОСИО	
Новоросс	новоросс	Новороссийск
Р-н-Д	рнд	Ростов-на-Дону
RND		
RnD		
РнД		
Наб. Челны	набчелны	Набережные Челны
Чебы	чебы	Чебоксары
Крск	крск	Красноярск
krsk		
kzn	кзн	Казань
Kzn		
KZN		