

Исправление опечаток

Информационный поиск. Лекция №4



О преподавателях



Александр Бадьин

Руководитель команды предобработки запросов
отдела поиска и голосовых технологий.

- Занимаюсь обработкой запросов в поисковых системах VK
- Люблю C++

Telegram: [@a_badin](https://www.instagram.com/a_badin)

О преподавателях



Арина Косовская

Программист-исследователь в команде ранжирования отдела поиска и голосовых технологий.

- Занимаюсь улучшением качества ранжирования
- Люблю обучать модели (:

Telegram: [@arinakosovskaia](https://www.t.me/arinakosovskaia)

План лекции

1

Существующие
алгоритмы

2

Оценка близости
исправлений

3

Модель зашумленного
канала

4

Детали реализации

5

Современные подходы

Опечатки в запросах

10% - 20% запросов имеют хотя бы одну опечатку

Примеры:

- Орфография
 - Замена буквы подслуш~~е~~но → подслушано
 - Пропуск буквы п~~о~~слушано → подслушано
 - Лишняя буква подслуша~~н~~но → подслушано
 - Перестановка букв Алексан~~рд~~ → Александр

Опечатки в запросах

10% - 20% запросов имеют хотя бы одну опечатку

Примеры:

- Орфография
 - Замена буквы подслуш~~е~~но → подслушано
 - Пропуск буквы п~~о~~слушано → подслушано
 - Лишняя буква подслуша~~н~~но → подслушано
 - Перестановка букв Алексан~~рд~~ → Александр
- Пробелы
 - Пропуск пробела не~~с~~мотрите наверх → не смотрите наверх
 - Лишний пробел не смотрите на ~~а~~ верх → не смотрите наверх

Опечатки в запросах

10% - 20% запросов имеют хотя бы одну опечатку

Примеры:

- Орфография
 - Замена буквы подслуш~~е~~но → подслушано
 - Пропуск буквы п~~о~~слушано → подслушано
 - Лишняя буква подслуша~~н~~но → подслушано
 - Перестановка букв Алексан~~рд~~ → Александр
- Пробелы
 - Пропуск пробела не~~с~~мотрите наверх → не смотрите наверх
 - Лишний пробел не смотрите на ~~а~~ верх → не смотрите наверх
- Транслитерация кудаго → KudaGo
- Раскладка ~~мл~~ → vk

Простой алгоритм исправления опечаток

Простой алгоритм исправления опечаток

1. Составить словарь

Простой алгоритм исправления опечаток

1. Составить словарь
2. Если в запросе существует слово, которого нет в словаре - значит это опечатка

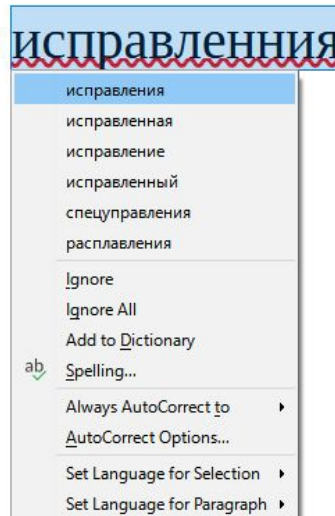
Простой алгоритм исправления опечаток

1. Составить словарь
2. Если в запросе существует слово, которого нет в словаре - значит это опечатка
3. Для исправления предлагаем ближайшее слово из словаря

Простой алгоритм исправления опечаток

Обычно используется в текстовых редакторах

Простой алгоритм исправления



If `bash` is invoked with the name `sh`, it tries to mimic the startup behavior of `histori` as well. When invoked as an interactive login shell, or a non-interactive shell with the file `and ~/.profile`, in that order. The `--noprofile` option may be used to inhibit this the variable `ENV`, expands its value if it is defined, and uses the expanded value as the attempt to read and execute commands from any other startup files, the `--rcfile` option to read any other startup files. When invoked as `sh`, `bash` enters `posix` mode after the st

When `bash` is started in `posix` mode, as with the `--posix` command line option, it follows the `ENV` variable and commands are read and executed from the file whose name is the expan

`Bash` attempts to determine when it is being run with its standard input connected to a r the secure shell daemon `sshd`. If `bash` determines it is being run in this fashion, it rea It will not do this if invoked as `sh`. The `--norc` option may be used to inhibit this beh `rshd` does not generally invoke the shell with those options or allow them to be specifie

If the shell is started with the effective user (group) id not equal to the real user (gr functions are not inherited from the environment, the `SHELLOPTS`, `BASHOPTS`, `CDPATH`, a effective user id is set to the real user id. If the `-p` option is supplied at invocation

DEFINITIONS

1 → posit	5 → pew six	9 → psis	c → posies	g → pow six
2 → six	6 → poo six	0 → pix	d → pa six	h → poise
3 → Po six	7 → posts	a → pox	e → pi six	i → do six
4 → P0 six	8 → poses	b → posits	f → poi six	j → go six

Простой алгоритм исправления опечаток

GNU Aspell

```
srch-devops4 aspell6-ru-0.99f7-1$ echo послушано | aspell -l ru -a  
@(#) International Ispell Version 3.1.20 (but really Aspell 0.60.8.1)  
& послушано 5 0: подслушано, прослушано, послушно, послушало, послушный
```

Hunspell (Chrome, Firefox, LibreOffice, Telegram)

Hunspell 1.7.2

луномосик

& луномосик 1 0: полуночник

наталная карта

& наталная 4 0: латанная, натальная, наталкивая, налетая

натальная

+ натальный

Составление словаря

Составление словаря

- Орфографические словари
 - Низкая полнота
 - Низкая доля ошибок

Составление словаря

- Орфографические словари

- Низкая полнота
- Низкая доля ошибок

АБРАКАДАБРА, -ы, ж. Бессмысленный, непонятный набор слов [первонач.: таинственное персидское слово, служившее спасительным магическим заклинанием].

АБРЕК, -а, м. В период присоединения Кавказа к России: горец, участвовавший в борьбе против царских войск и администрации.

АБРИКОС, -а, род.мн. -ов, м. Южное фруктовое дерево сем. розоцветных, дающее сочные сладкие плоды с крупной косточкой, а также плод его. II прил. абрикосный, -ая, -ое ы абрикосовый, оая, -ое.

Составление словаря

- Орфографические словари
 - Низкая полнота
 - Низкая доля ошибок
- Составить из наборов текстов
 - Средняя полнота
 - Средняя доля ошибок

Составление словаря

- Орфографические словари
 - Низкая полнота
 - Низкая доля ошибок
- Составить из наборов текстов
 - Средняя полнота
 - Средняя доля ошибок

Дегустацию посетили турецкие дистрибьюторы и импортеры, а также руководители крупных ритейл-сетей, таких как Migros A.Ş, Maxx Royal, ABV Alkollü İçecekler и других. Оценить российскую продукцию пришли также турецкие инфлюенсеры с аудиторией более 4 миллионов подписчиков.

Составление словаря

- Орфографические словари
 - Низкая полнота
 - Низкая доля ошибок
- Составить из наборов текстов
 - Средняя полнота
 - Средняя доля ошибок

Дегустацию посетили турецкие дистрибьюторы и импортеры, а также руководители крупных ритейл-сетей, таких как Migros A.Ş, Maxx Royal, ABV Alkollü İçecekler и других. Оценить российскую продукцию пришли также турецкие инфлюенсеры с аудиторией более 4 миллионов подписчиков.

Кринж-комедия

Материал из Википедии — свободной энциклопедии

[\[править \]](#) [\[править код \]](#)

Кринж-комедия, **комедия содрогания** (англ. *Cringe comedy*) — особый **комедийный** жанр, в котором юмор проистекает из чувства **социальной неловкости**. Часто вызывающая передергивание комедия имеет вид **мокьюментари** (псевдодокументалистики), её сюжет развивается в правдоподобной обстановке, например, такой как **рабочее место**, чтобы придать происходящему видимость реальности^[1].

Определение близости между словами

Определение близости между словами

Расстояние Левенштейна - минимальное число операций для преобразования одного слова в другое.

- Замена символа
- Вставка символа
- Удаление символа

Определение близости между словами

Расстояние Левенштейна - минимальное число операций для преобразования одного слова в другое.

- Замена символа
- Вставка символа
- Удаление символа
- Транспозиция символов (Расстояние Дamerau — Левенштейна)

Алгоритм Вагнера — Фишера

Создается таблица $M \times N$

- Первая строка заполняется от $0 \dots N$
- Первый столбец заполняется от $0 \dots M$

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1						
а	2						
р	3						
а	4						
т	5						
о	6						
в	7						

Алгоритм Вагнера — Фишера

Остальные ячейки заполняются выбором минимального значения между:

- Значением верхней ячейки + 1
- Значением левой ячейки + 1
- Значением ячейки, расположенной по диагонали
 - + 1, если буквы в строке и столбце не совпадают
 - + 0, если буквы в строке и столбце совпадают

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1						
а	2						
р	3						
а	4						
т	5						
о	6						
в	7						

Алгоритм Вагнера — Фишера

Остальные ячейки заполняются выбором минимального значения между:

- Значением верхней ячейки + 1
- Значением левой ячейки + 1
- Значением ячейки, расположенной по диагонали
 - + 1, если буквы в строке и столбце не совпадают
 - + 0, если буквы в строке и столбце совпадают

		с	а	м	а	р	а
	0(0)	1(2)	2	3	4	5	6
с	1(2)	0					
а	2						
р	3						
а	4						
т	5						
о	6						
в	7						

Алгоритм Вагнера — Фишера

Остальные ячейки заполняются выбором минимального значения между:

- Значением верхней ячейки + 1
- Значением левой ячейки + 1
- Значением ячейки, расположенной по диагонали
 - + 1, если буквы в строке и столбце не совпадают
 - + 0, если буквы в строке и столбце совпадают

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1	0					
а	2						
р	3						
а	4						
т	5						
о	6						
в	7						

Алгоритм Вагнера — Фишера

Остальные ячейки заполняются выбором минимального значения между:

- Значением верхней ячейки + 1
- Значением левой ячейки + 1
- Значением ячейки, расположенной по диагонали
 - + 1, если буквы в строке и столбце не совпадают
 - + 0, если буквы в строке и столбце совпадают

		с	а	м	а	р	а
	0	1(2)	2(3)	3	4	5	6
с	1	0(1)	1				
а	2						
р	3						
а	4						
т	5						
о	6						
в	7						

Алгоритм Вагнера — Фишера

Остальные ячейки заполняются выбором минимального значения между:

- Значением верхней ячейки + 1
- Значением левой ячейки + 1
- Значением ячейки, расположенной по диагонали
 - + 1, если буквы в строке и столбце не совпадают
 - + 0, если буквы в строке и столбце совпадают

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1	0	1	2	3	4	5
а	2	1	0	1	2	3	4
р	3	2					
а	4						
т	5						
о	6						
в	7						

Алгоритм Вагнера — Фишера

Остальные ячейки заполняются выбором минимального значения между:

- Значением верхней ячейки + 1
- Значением левой ячейки + 1
- Значением ячейки, расположенной по диагонали
 - + 1, если буквы в строке и столбце не совпадают
 - + 0, если буквы в строке и столбце совпадают

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1	0	1	2	3	4	5
а	2	1	0	1	2	3	4
р	3	2					
а	4						
т	5						
о	6						
в	7						

Алгоритм Вагнера — Фишера

Остальные ячейки заполняются выбором минимального значения между:

- Значением верхней ячейки + 1
- Значением левой ячейки + 1
- Значением ячейки, расположенной по диагонали
 - + 1, если буквы в строке и столбце не совпадают
 - + 0, если буквы в строке и столбце совпадают

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1	0	1	2	3	4	5
а	2	1(2)	0(1)	1	2	3	4
р	3	2(3)	1				
а	4						
т	5						
о	6						
в	7						

Алгоритм Вагнера — Фишера

$d(\text{"самара"}, \text{"саратов"}) = 4$

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1	0	1	2	3	4	5
а	2	1	0	1	2	3	4
р	3	2	1	1	2	2	3
а	4	3	2	2	1	2	2
т	5	4	3	3	2	2	3
о	6	5	4	4	3	3	3
в	7	6	5	5	4	4	4

Алгоритм Вагнера — Фишера

$d(\text{“самара”}, \text{“саратов”}) = 4$

$$D(i, j) = \begin{cases} 0, & i = 0, j = 0 \\ i, & j = 0, i > 0 \\ j, & i = 0, j > 0 \\ \min\{ \\ \quad D(i, j - 1) + 1, \\ \quad D(i - 1, j) + 1, \\ \quad D(i - 1, j - 1) + m(S_1[i], S_2[j]) \\ \} & j > 0, i > 0 \end{cases}$$

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1	0	1	2	3	4	5
а	2	1	0	1	2	3	4
р	3	2	1	1	2	2	3
а	4	3	2	2	1	2	2
т	5	4	3	3	2	2	3
о	6	5	4	4	3	3	3
в	7	6	5	5	4	4	4

Алгоритм Вагнера — Фишера

$d(\text{“самара”, “саратов”}) = 4$

Построение редакционного предписания

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1	0	1	2	3	4	5
а	2	1	0	1	2	3	4
р	3	2	1	1	2	2	3
а	4	3	2	2	1	2	2
т	5	4	3	3	2	2	3
о	6	5	4	4	3	3	3
в	7	6	5	5	4	4	4

Алгоритм Вагнера — Фишера

$d(\text{“самара”}, \text{“саратов”}) = 4$

Построение редакционного предписания

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1	0	1	2	3	4	5
а	2	1	0	1	2	3	4
р	3	2	1	1	2	2	3
а	4	3	2	2	1	2	2
т	5	4	3	3	2	2	3
о	6	5	4	4	3	3	3
в	7	6	5	5	4	4	4

Алгоритм Вагнера — Фишера

$d(\text{"самара"}, \text{"саратов"}) = 4$

Построение редакционного предписания:

- ↘ Замена одной буквы на другую
- → Удаление буквы
- ↓ Добавление буквы

		с	а	м	а	р	а
	0	1	2	3	4	5	6
с	1	0	1	2	3	4	5
а	2	1	0	1	2	3	4
р	3	2	1	1	2	2	3
а	4	3	2	2	1	2	2
т	5	4	3	3	2	2	3
о	6	5	4	4	3	3	3
в	7	6	5	5	4	4	4

Минусы простого алгоритма исправления опечаток

Минусы простого алгоритма исправления опечаток

- Большое число кандидатов с одинаковой оценкой **татья**:

статья
татьян
третья
татьяна
статьи
таиться
татьяне
статью
статье
тетя
татьяну
татьяны
стать
таится
платья
катя

Минусы простого алгоритма исправления опечаток

- Большое число кандидатов с одинаковой оценкой **татья**:
- Сложность составления словаря: Д**а**винчик → Дайвинчик

статья
татьян
третья
татьяна
статьи
таиться
татьяне
статью
статье
тетя
татьяну
татьяны
стать
таится
платья
катья

Минусы простого алгоритма исправления опечаток

- Большое число кандидатов с одинаковой оценкой **татья**:
- Сложность составления словаря: Д**а**винчик → Дайвинчик
- Нет учета контекста: ната**л**ная карта -> нательная карта

статья
татьян
третья
татьяна
статьи
таиться
татьяне
статью
статье
тетя
татьяну
татьяны
стать
таится
платья
катья

Модель зашумленного канала

Модель зашумленного канала

orig - оригинальный запрос

fix - исправленный запрос

D - множество всех запросов

$P(\text{fix}|\text{orig})$ - вероятность запроса *fix*, если ввели *orig*

$$\text{fix}^* = \arg \max_{\text{fix} \in D} P(\text{fix}|\text{orig})$$

Модель зашумленного канала

orig - оригинальный запрос

fix - исправленный запрос

D - множество всех запросов

$P(\text{fix}|\text{orig})$ - вероятность запроса *fix*, если ввели *orig*

$$\text{fix}^* = \arg \max_{\text{fix} \in D} P(\text{fix}|\text{orig})$$

$$\text{fix}^* = \arg \max_{\text{fix} \in D} \frac{P(\text{orig}|\text{fix})P(\text{fix})}{P(\text{orig})}$$

Модель зашумленного канала

orig - оригинальный запрос

fix - исправленный запрос

D - множество всех запросов

$P(\text{fix}|\text{orig})$ - вероятность запроса fix, если ввели orig

$$\text{fix}^* = \arg \max_{\text{fix} \in D} P(\text{fix}|\text{orig})$$

$$\text{fix}^* = \arg \max_{\text{fix} \in D} \frac{P(\text{orig}|\text{fix})P(\text{fix})}{P(\text{orig})}$$

$$\text{fix}^* = \arg \max_{\text{fix} \in D} P(\text{orig}|\text{fix})P(\text{fix})$$

Модель зашумленного канала

$$fix^* = \arg \max_{fix \in D} P(orig|fix)P(fix)$$

$P(orig|fix)$ - вероятность получить запрос $orig$, при намерении пользователя ввести fix

$P(fix)$ - вероятность запроса fix

Модель зашумленного канала

$$fix^* = \arg \max_{fix \in D} \underbrace{P(orig|fix)P(fix)}$$

Модель
ошибок

$P(orig|fix)$ - вероятность получить запрос $orig$, при намерении пользователя ввести fix

$P(fix)$ - вероятность запроса fix

Модель зашумленного канала

$$fix^* = \arg \max_{fix \in D} \underbrace{P(orig|fix)}_{\text{Модель ошибок}} \underbrace{P(fix)}_{\text{Языковая модель}}$$

$P(orig|fix)$ - вероятность получить запрос $orig$, при намерении пользователя ввести fix

$P(fix)$ - вероятность запроса fix

Модель ошибок

$$P(orig|fix) \approx \alpha^{-lev(orig,fix)}$$

Модель ошибок

$P(\text{“собранный”} | \text{“собранный”}) \approx P(\text{пропуск “н”})$

Модель ошибок

$P(\text{“собранный”} | \text{“собранный”}) \approx P(\text{пропуск “н”})$

$P(\text{“соьранный”} | \text{“собранный”}) \approx P(\text{замена “б” на “ь”})$

Модель ошибок

$P(\text{“собранный”} | \text{“собранный”}) \approx P(\text{пропуск “н”})$

$P(\text{“соьранный”} | \text{“собранный”}) \approx P(\text{замена “б” на “ь”})$

Имея данные orig \rightarrow fixed:

Модель ошибок

$P(\text{“собранный”} | \text{“собранный”}) \approx P(\text{пропуск “н”})$

$P(\text{“соьранный”} | \text{“собранный”}) \approx P(\text{замена “б” на “ь”})$

Имея данные $\text{orig} \rightarrow \text{fixed}$:

- Получить редакционное предписание по Левенштейну

Модель ошибок

$P(\text{“собранный”} | \text{“собранный”}) \approx P(\text{пропуск “н”})$

$P(\text{“соьранный”} | \text{“собранный”}) \approx P(\text{замена “б” на “ь”})$

Имея данные `orig` \rightarrow `fixed`:

- Получить редакционное предписание по Левенштейну
- Посчитать вероятности ошибок

Модель ошибок

$P(\text{“собранный”} | \text{“собранный”}) \approx P(\text{пропуск “н”})$

$P(\text{“соьранный”} | \text{“собранный”}) \approx P(\text{замена “б” на “ь”})$

Имея данные `orig` \rightarrow `fixed`:

- Получить редакционное предписание по Левенштейну
- Посчитать вероятности ошибок

Источники данных:

- Готовые датасеты
- Поведенческая статистика

Модель ошибок

$P(\text{"собранный"} | \text{"собранный"}) \approx P(\text{пропуск "н"})$

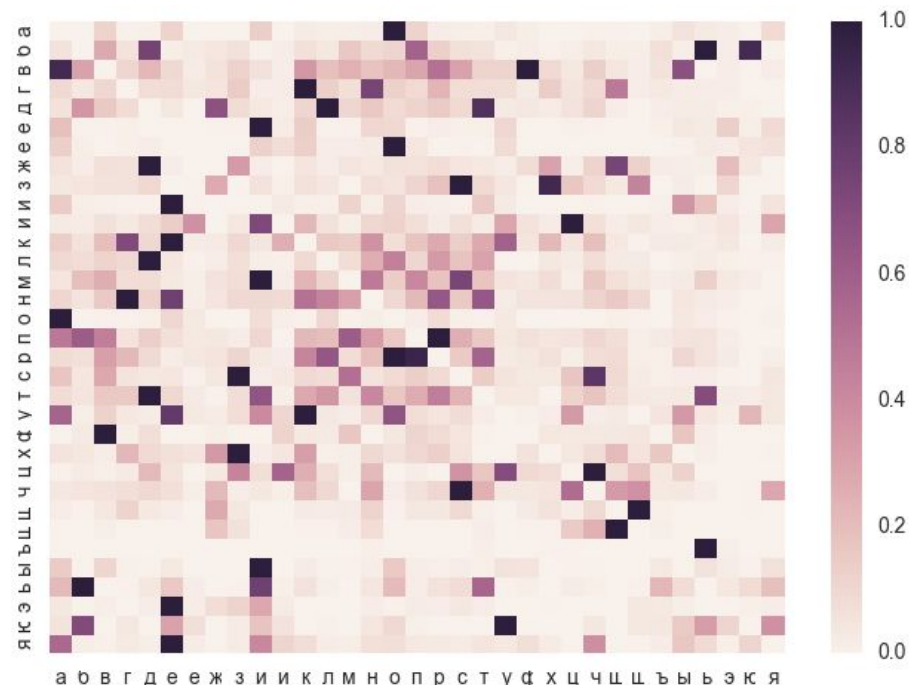
$P(\text{"соьранный"} | \text{"собранный"}) \approx P(\text{замена "б" на "ь"})$

Имея данные orig \rightarrow fixed:

- Получить редакционное предписание по Левенштейну
- Посчитать вероятности ошибок

Источники данных:

- Готовые датасеты
- Поведенческая статистика



Модель ошибок

$$P(orig|fix) \approx P(o_1|f_1)P(o_2|f_2)...P(o_n|f_m)$$

Модель ошибок

$$P(orig|fix) \approx P(o_1|f_1)P(o_2|f_2)...P(o_n|f_m)$$

$$\log(P(orig|fix)) \approx \log(P(o_1|f_1)) + \log(P(o_2|f_2))...\log(P(o_n|f_m))$$

Модель ошибок

$$P(orig|fix) \approx P(o_1|f_1)P(o_2|f_2)...P(o_n|f_m)$$

$$\log(P(orig|fix)) \approx \log(P(o_1|f_1)) + \log(P(o_2|f_2))...\log(P(o_n|f_m))$$


Переход к взвешенному расстоянию Дамерау-Левенштейна

Взвешенное расстояние Левенштейна

Каждая операция имеет свой вес

- Значение верхней ячейки + $P(\text{"вставки"})$
- Значение левой ячейки + $P(\text{"удаления"})$
- Значение ячейки, расположенной по диагонали + $P(\text{"замены"})$

		с	а	м
с				
а				
р				
а				



The diagram shows a 6x5 grid representing a dynamic programming table for the weighted Levenshtein distance between the strings 'с' and 'а'. The columns are labeled with the characters of the second string (с, а, м) and the rows with the characters of the first string (с, а, р, а). The top row and the first column are highlighted in light green. The cells (3,2), (4,2), (4,3), and (5,2) are highlighted in light gray. Three arrows point to the cell (5,2): a red arrow from the left (cell 5,1), a green arrow from above (cell 4,2), and a blue arrow from the top-left (cell 4,1). This illustrates the calculation of the distance for the character 'р' in the first string against the character 'а' in the second string.

Модель зашумленного канала

$$fix^* = \arg \max_{fix \in D} \underbrace{P(orig|fix)}_{\text{Модель ошибок}} \underbrace{P(fix)}_{\text{Языковая модель}}$$

$P(orig|fix)$ - вероятность получить запрос $orig$, при намерении пользователя ввести fix

$P(fix)$ - вероятность запроса fix

Языковая модель

Расчет по частотности:

$$P(fix) \approx \frac{C(fix)}{\sum_q C(q)}$$

Языковая модель

Расчет по частотности:

$$P(fix) \approx \frac{C(fix)}{\sum_q C(q)}$$

Используем разбиение на слова

$$P(fix) = P(w_1 w_2 w_3 \dots w_n) = P(w_1) P(w_2 | w_1) P(w_3 | w_1 w_2) \dots P(w_n | w_1 \dots w_{n-1})$$

Языковая модель

Расчет по частотности:

$$P(fix) \approx \frac{C(fix)}{\sum_q C(q)}$$

Используем разбиение на слова

$$P(fix) = P(w_1 w_2 w_3 \dots w_n) = P(w_1) P(w_2 | w_1) P(w_3 | w_1 w_2) \dots P(w_n | w_1 \dots w_{n-1})$$

При предположении о независимости слов друг от друга:

$$P(fix) \approx P(w_1) P(w_2) \dots P(w_n)$$

Языковая модель

Расчет по частотности:

$$P(fix) \approx \frac{C(fix)}{\sum_q C(q)}$$

Используем разбиение на слова

$$P(fix) = P(w_1 w_2 w_3 \dots w_n) = P(w_1) P(w_2 | w_1) P(w_3 | w_1 w_2) \dots P(w_n | w_1 \dots w_{n-1})$$

Представляем как цепь Маркова

$$P(w_k | w_1 \dots w_{k-1}) \approx P(w_k | w_{k-1})$$

Языковая модель

Расчет по частотности:

$$P(fix) \approx \frac{C(fix)}{\sum_q C(q)}$$

Используем разбиение на слова

$$P(fix) = P(w_1 w_2 w_3 \dots w_n) = P(w_1) P(w_2 | w_1) P(w_3 | w_1 w_2) \dots P(w_n | w_1 \dots w_{n-1})$$

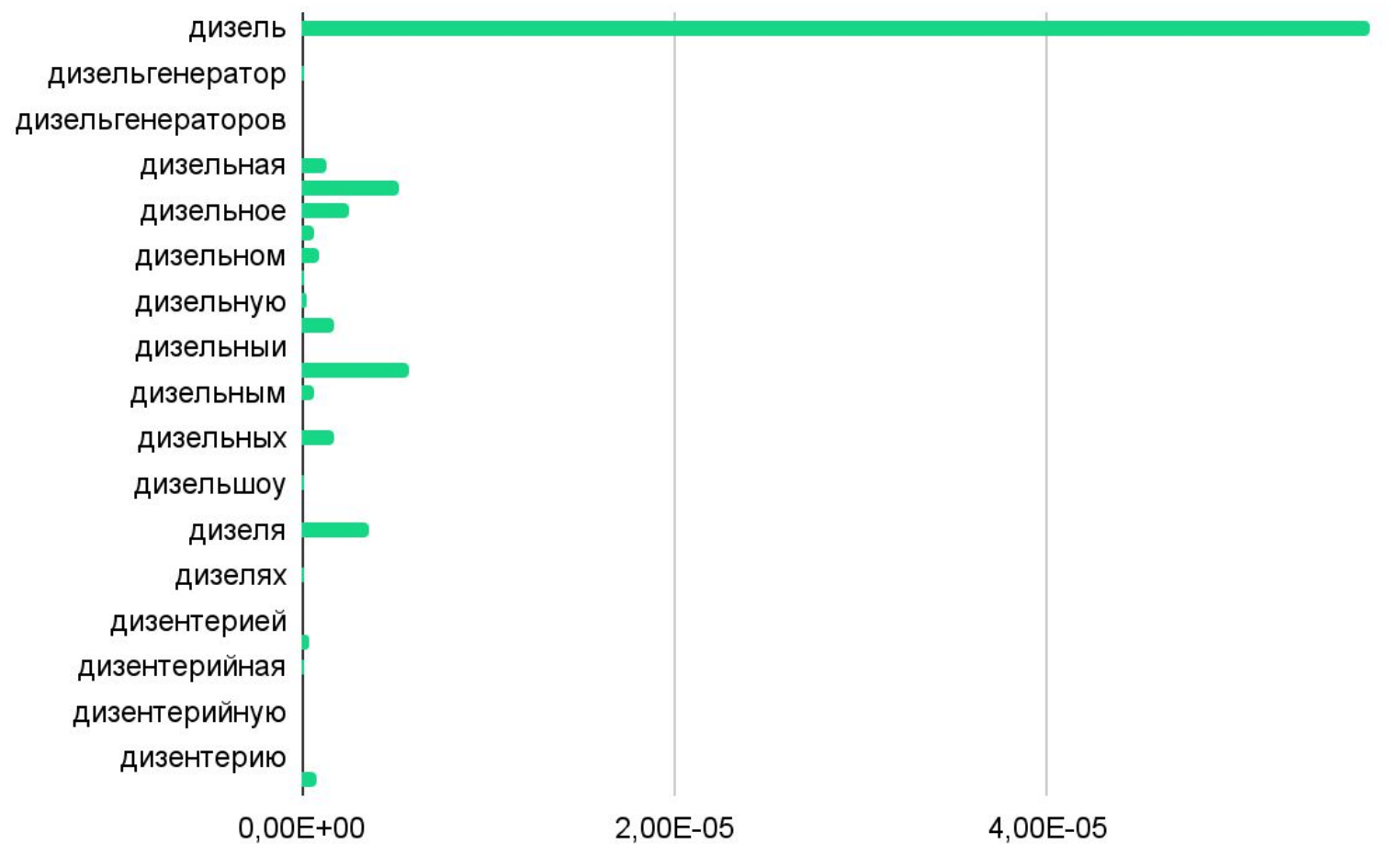
Представляем как цепь Маркова

$$P(w_k | w_1 \dots w_{k-1}) \approx P(w_k | w_{k-1})$$

Биграммная модель:

$$P(fix) \approx P(w_1 | \hat{\ }) P(w_2 | w_1) \dots P(w_n | w_{n-1}) P(\$ | w_n)$$

Сглаживание в языковых моделях



Модель зашумленного канала

$$fix^* = \arg \max_{fix \in D} \underbrace{P(orig|fix)}_{\text{Модель ошибок}} \underbrace{P(fix)}_{\text{Языковая модель}}$$

$P(orig|fix)$ - вероятность получить запрос $orig$, при намерении пользователя ввести fix

$P(fix)$ - вероятность запроса fix

Модель зашумленного канала

$$fix^* = \arg \max_{\underline{fix \in D}} \underbrace{P(orig|fix)}_{\text{Модель ошибок}} \underbrace{P(fix)}_{\text{Языковая модель}}$$

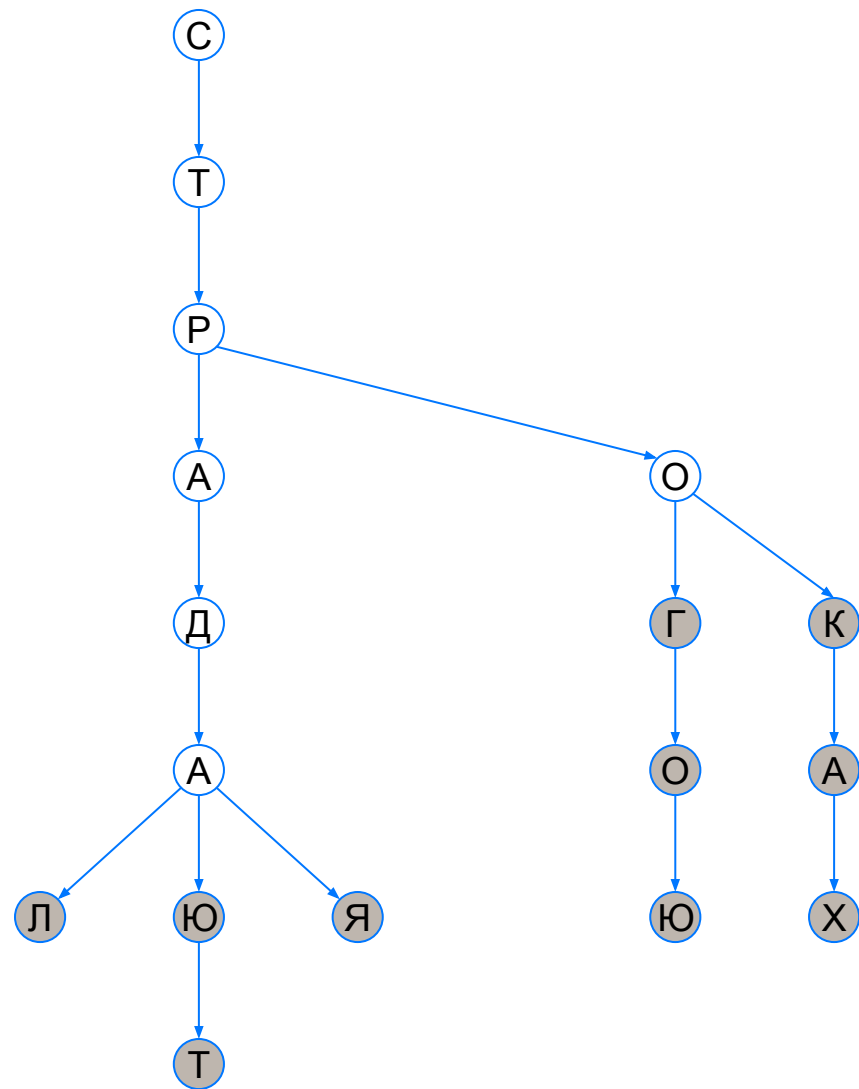
$P(orig|fix)$ - вероятность получить запрос $orig$, при намерении пользователя ввести fix

$P(fix)$ - вероятность запроса fix

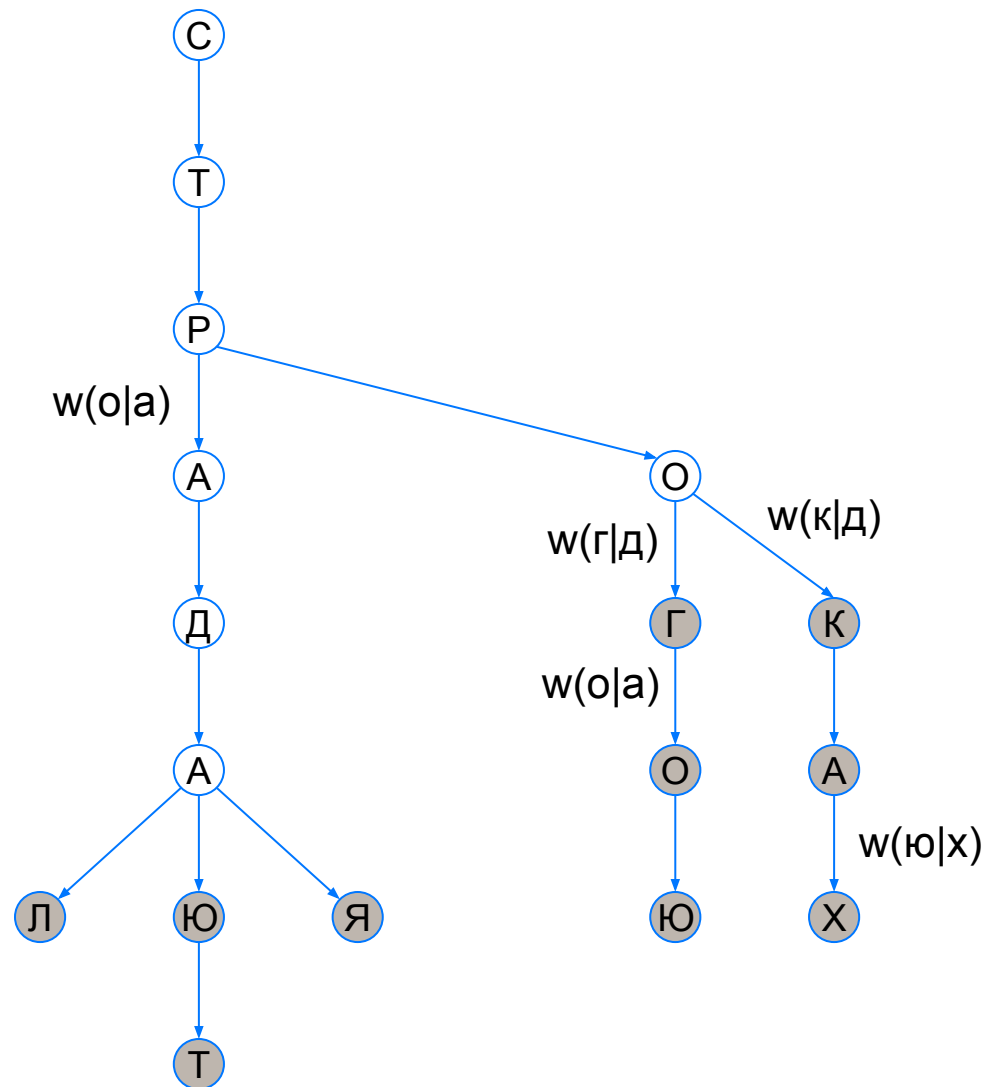
Генерация кандидатов

страдаю	от	бессоницы
страдаю (1.7)	от (4)	бессонницы (1.7)
страдают (-5.5)	вот (8)	бессонница (-4.3)
страдая (-10)	тот (6.1)	бессоннице (-6)
страдае (-12)	о	бессонницу (-7)
страдай (-13)	то	
страда (-13)	од	
строгую (-13)	рот	
страдал (-13)	отт	
строю (-15)	пот	
строгая (-16)	отв	
продаю (-16)	ост	
страдан (-18)	кот	
строгую (-20)	об	
трогаю (-20)		
строга (-21)		
строфа (-21)		
строка (-22)		
сброда (-23)		
строках (-23)		
строфах (-26)		
строкам (-31)		

Префиксное дерево



Префиксное дерево



Генерация кандидатов

страдаю	от	бессоницы
<p>страдаю (1.7)</p> <p>страдают (-5.5)</p> <p>страдаю (-10)</p> <p>страдаю (-12)</p> <p>страдаю (-13)</p> <p>страдаю (-13)</p> <p>строгую (-13)</p> <p>страдал (-13)</p> <p>строю (-15)</p> <p>строгая (-16)</p> <p>продаю (-16)</p> <p>страдан (-18)</p> <p>строгую (-20)</p> <p>трогаю (-20)</p> <p>строга (-21)</p> <p>строфа (-21)</p> <p>строка (-22)</p> <p>сброда (-23)</p> <p>строках (-23)</p> <p>строфах (-26)</p> <p>строкам (-31)</p>	<p>от (4)</p> <p>вот (8)</p> <p>тот (6.1)</p> <p>о</p> <p>то</p> <p>од</p> <p>рот</p> <p>отт</p> <p>пот</p> <p>отв</p> <p>ост</p> <p>кот</p> <p>об</p>	<p>бессонница (1.7)</p> <p>бессонницы (-4.3)</p> <p>бессоннице (-6)</p> <p>бессонницу (-7)</p> <p>одиночество (-20.1)</p> <p>одиночестве (-20.2)</p>

Генерация кандидатов

страдаю	от	бессоницы
<p>страдаю (1.7)</p> <p>страдают (-5.5)</p> <p>страдаю (-10)</p> <p>страдаю (-12)</p> <p>страдаю (-13)</p> <p>страда (-13)</p> <p>строю (-13)</p> <p>строю (-13)</p> <p>страдал (-13)</p> <p>строю (-15)</p> <p>строгая (-16)</p> <p>продаю (-16)</p> <p>страдан (-18)</p> <p>строю (-20)</p> <p>трогаю (-20)</p> <p>строга (-21)</p> <p>строфа (-21)</p> <p>строка (-22)</p> <p>сброда (-23)</p> <p>строках (-23)</p> <p>строфах (-26)</p> <p>строкам (-31)</p>	<p>от (4)</p> <p>от (-4)</p> <p>вот (8)</p> <p>тот (6.1)</p> <p>о</p> <p>то</p> <p>од</p> <p>рот</p> <p>отт</p> <p>пот</p> <p>отв</p> <p>ост</p> <p>кот</p> <p>об</p>	<p>бессонница (1.7)</p> <p>бессонницы (-4.3)</p> <p>бессоннице (-6)</p> <p>бессонницу (-7)</p> <p>одинокчество (-20.1)</p> <p>одинокчестве (-20.2)</p>

Генерация кандидатов

страдаю	от	бессоницы
<p>страдаю (1.7)</p> <p>страдают (-5.5)</p> <p>страдаю (-10)</p> <p>страдаю (-12)</p> <p>страдаю (-13)</p> <p>страдаю (-13)</p> <p>строю (-13)</p> <p>строю (-13)</p> <p>страдал (-13)</p> <p>строю (-15)</p> <p>строгая (-16)</p> <p>продаю (-16)</p> <p>страдан (-18)</p> <p>строую (-20)</p> <p>трогаю (-20)</p> <p>строга (-21)</p> <p>строфа (-21)</p> <p>строка (-22)</p> <p>сброда (-23)</p> <p>строках (-23)</p> <p>строфах (-26)</p> <p>строкам (-31)</p>	<p>от (4)</p> <p>от (-4)</p> <p>вот (8)</p> <p>тот (6.1)</p> <p>о</p> <p>то</p> <p>од</p> <p>рот</p> <p>отт</p> <p>пот</p> <p>отв</p> <p>ост</p> <p>кот</p> <p>об</p>	<p>бессонница (1.7)</p> <p>бессонницы (-4.3)</p> <p>бессоннице (-6)</p> <p>бессонницу (-7)</p> <p>одинокчество (-20.1)</p> <p>одинокчестве (-20.2)</p>

Генерация кандидатов

страдаю	от	бессоницы
<p>страдаю (1.7)</p> <p>страдают (-5.5)</p> <p>страдаю (-10)</p> <p>страдаю (-12)</p> <p>страдаю (-13)</p> <p>страдаю (-13)</p> <p>строю (-13)</p> <p>строю (-13)</p> <p>страдаю (-13)</p> <p>строю (-15)</p> <p>строгая (-16)</p> <p>страдаю (-16)</p> <p>страдаю (-18)</p> <p>строю (-20)</p> <p>страдаю (-20)</p> <p>строга (-21)</p> <p>строфа (-21)</p> <p>строка (-22)</p> <p>сброда (-23)</p> <p>строках (-23)</p> <p>строфах (-26)</p> <p>строкам (-31)</p>	<p>от (4)</p> <p>вот (8)</p> <p>тот (6.1)</p> <p>о</p> <p>то</p> <p>од</p> <p>рот</p> <p>отт</p> <p>пот</p> <p>отв</p> <p>ост</p> <p>кот</p> <p>об</p>	<p>бессонница (1.7)</p> <p>бессонницы (-4.3)</p> <p>бессоннице (-6)</p> <p>бессонницу (-7)</p> <p>одинокчество (-20.1)</p> <p>одинокчестве (-20.2)</p>

Ускорение поиска кандидатов

Ускорение поиска кандидатов

$$Index[word_{typed}] \rightarrow [c_1, c_2, \dots]$$

Ускорение поиска кандидатов

$$Index[word_{typed}] \rightarrow [c_1, c_2, \dots]$$

$$F(x), F(word_{typed}) = F(c_1) = F(c_2) \dots = F(c_n)$$

Ускорение поиска кандидатов

$$Index[word_{typed}] \rightarrow [c_1, c_2, \dots]$$

$$F(x), F(word_{typed}) = F(c_1) = F(c_2) \dots = F(c_n)$$

Soundex одинаковый индекс для строк, имеющих схожее звучание

Ускорение поиска кандидатов

$$Index[word_{typed}] \rightarrow [c_1, c_2, \dots]$$

$$F(x), F(word_{typed}) = F(c_1) = F(c_2) \dots = F(c_n)$$

Soundex одинаковый индекс для строк, имеющих схожее звучание

$$F_s(\text{quadrobists}) = F_s(\text{quadrobers}) = F_s(\text{quadrobeasts}) = \text{Q361}$$

Ускорение поиска кандидатов

$$Index[word_{typed}] \rightarrow [c_1, c_2, \dots]$$

$$F(x), F(word_{typed}) = F(c_1) = F(c_2) \dots = F(c_n)$$

Soundex одинаковый индекс для строк, имеющих схожее звучание

$$F_s(\text{quadrobists}) = F_s(\text{quadrobers}) = F_s(\text{quadrobeasts}) = \text{Q361}$$

$$Index[\text{Q361}] \rightarrow [\text{quadrobists}, \text{quadrobers}, \text{quadrobics}, \dots]$$

Ускорение поиска кандидатов

$$Index[word_{typed}] \rightarrow [c_1, c_2, \dots]$$

$$F(x), F(word_{typed}) = F(c_1) = F(c_2) \dots = F(c_n)$$

Soundex одинаковый индекс для строк, имеющих схожее звучание

$$F_s(\text{quadrobists}) = F_s(\text{quadrobers}) = F_s(\text{quadrobeasts}) = \text{Q361}$$

$$Index[\text{Q361}] \rightarrow [\text{quadrobists}, \text{quadrobers}, \text{quadrobics}, \dots]$$

Работает не для всех опечаток

Symspell

Symspell

Symspell множество индексов, полученных удалением буквы

<https://seekstorm.com/blog/1000x-spelling-correction/>

Symspell

Symspell множество индексов, полученных удалением буквы

снрил

сериал

Symspell

Symspell множество индексов, полученных удалением буквы

снрил

сериал

снрил

нил

нри

нрил

нрл

рил

сил

сни

снил

снл

снр

снри

снрл

сри

срил

срл

Symspell

Symspell множество индексов, полученных удалением буквы

снрил

сериал

снрил

еиал

нил

ерал

нри

ериа

нрил

ериал

нрл

ерил

рил

риал

сил

сеал

сни

...

снил

сриа

снл

сриал

снр

срил

снри

снрл

сри

срил

срл

Symspell

Symspell множество индексов, полученных удалением буквы

снрил

сериал

снрил

еиал

нил

ерал

нри

ериа

нрил

ериал

нрл

ерил

рил

риал

сил

сеал

сни

...

снил

сриа

снл

сриал

снр

срил

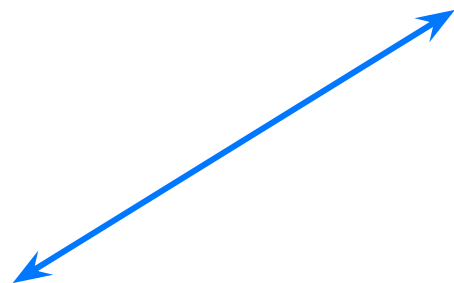
снри

снрл

сри

срил

срл



Дополнительные улучшения

- Ранжирование кандидатов
- Разбиение слов
- Склейка слов
- Транслитерация
- Переключение раскладки

Материалы

- Speech and Language Processing
Dan Jurafsky and James H. Martin
<https://web.stanford.edu/~jurafsky/slp3/>
- How to Write a Spelling Corrector
<https://www.norvig.com/spell-correct.html>
- 1000x Faster Spelling Correction algorithm
<https://seekstorm.com/blog/1000x-spelling-correction/>

Современные методы исправления опечаток

Проблемы простых методов и способы улучшения

- Не учитывают контекст слова с ошибкой или “рассматривают” недостаточно большое окно.
Требуют тяжелые словари.

Проблемы простых методов и способы улучшения

- Не учитывают контекст слова с ошибкой или “рассматривают” недостаточно большое окно. Требуют тяжелые словари.
- Вместо словарей, можно использовать предобученные контекстные представления слов, чтобы понимать значение слова в контексте всего предложения
- Чтобы работать с неизвестными словами или опечатками на уровне символов, можно использовать в том числе посимвольные эмбединги.

NeuSpell

Идея: использовать предобученные эмбединги для захвата контекста, комбинируя их с посимвольными архитектурами.

Spelling correction systems in NeuSpell (Word-Level Accuracy / Correction Rate)

	Synthetic		Natural		Ambiguous	
	WORD-TEST	PROB-TEST	BEA-60K	JFLEG	BEA-4660	BEA-322
ASPELL (Atkinson, 2019)	43.6 / 16.9	47.4 / 27.5	68.0 / 48.7	73.1 / 55.6	68.5 / 10.1	61.1 / 18.9
JAMSPELL (Ozinov, 2019)	90.6 / 55.6	93.5 / 68.5	97.2 / 68.9	98.3 / 74.5	98.5 / 72.9	96.7 / 52.3
CHAR-CNN-LSTM (Kim et al., 2015)	97.0 / 88.0	96.5 / 84.1	96.2 / 75.8	97.6 / 80.1	97.5 / 82.7	94.5 / 57.3
SC-LSTM (Sakaguchi et al., 2016)	97.6 / 90.5	96.6 / 84.8	96.0 / 76.7	97.6 / 81.1	97.3 / 86.6	94.9 / 65.9
CHAR-LSTM-LSTM (Li et al., 2018)	98.0 / 91.1	97.1 / 86.6	96.5 / 77.3	97.6 / 81.6	97.8 / 84.0	95.4 / 63.2
BERT (Devlin et al., 2018)	98.9 / 95.3	98.2 / 91.5	93.4 / 79.1	97.9 / 85.0	98.4 / 92.5	96.0 / 72.1
SC-LSTM						
+ ELMO (input)	98.5 / 94.0	97.6 / 89.1	96.5 / 79.8	97.8 / 85.0	98.2 / 91.9	96.1 / 69.7
+ ELMO (output)	97.9 / 91.4	97.0 / 86.1	98.0 / 78.5	96.4 / 76.7	97.9 / 88.1	95.2 / 63.2
+ BERT (input)	98.7 / 94.3	97.9 / 89.5	96.2 / 77.0	97.8 / 83.9	98.4 / 90.2	96.0 / 67.8
+ BERT (output)	98.1 / 92.3	97.2 / 86.9	95.9 / 76.0	97.6 / 81.0	97.8 / 88.1	95.1 / 67.2

Источник: <https://arxiv.org/pdf/2010.11085.pdf>

NeuSpell

- Обучается как sequence labeling, где каждый токен размечается исходным (если он правильный) или его исправлением (если неправильный).
- Для каждого токена модель предсказывает **распределение вероятностей** над всеми словами в словаре
- Чтобы собрать обучающий датасет, авторы создают синтетические наборы данных с ошибками, искажая правильные тексты: меняли/переставляли/удаляли символы, заменяли слова на их частые неправильные написания, заменяли символы на неправильные в зависимости от контекста.

SAGE

Идея: научить модель различать преднамеренные и непреднамеренные ошибки. Обучить для этого модель генерировать правильное предложение с помощью transformer-like архитектуры.

Идея: случайные зашумления данных не похожи на то, как реально ошибаются люди, они не помогут модели выучить орфографию и грамматику

Источник: <https://arxiv.org/pdf/2308.09435.pdf>
<https://habr.com/ru/companies/sberdevices/articles/763932/>

SAGE: генерация данных

По параллельному корпусу (предложение и его коррекция) собираются статистики для:

- 1) количества ошибок на предложение
- 2) типа ошибки (удаление/вставка/замена символа или пробела)
- 3) абсолютная и относительная (относительно предложения) позиции ошибки в предложении.

Статистика приводится к дискретному распределению, и получается распределение числа ошибок, их типов и позиций (внутри каждого типа). По этому распределению добавляются ошибки в “чистый” текст.

Источник: <https://arxiv.org/pdf/2308.09435.pdf>
<https://habr.com/ru/companies/sberdevices/articles/763932/>

SAGE: обучение

Предобучение:

- 1) Собрать большой корпус чистых текстов
- 2) Вставить ошибки с помощью статистического метода
- 3) Seq2Seq обучение: на вход модели подается зашумленное предложение, она должна сгенерировать корректное
- 4) Дополнительно энкодер учится предсказывать тип и позицию ошибки в исходном предложении

Дообучение:

- 5) Дообучить модель на датасетах с человеческими ошибками

SAGE

Model	RUSpellRU			MultidomainGold			MedSpellChecker			GitHubTypoCorpusRu		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Yandex.Speller	83.0	59.8	69.5	52.9	51.4	52.2	80.6	47.8	60.0	67.7	37.5	48.3
JamSpell	42.1	32.8	36.9	25.7	30.6	28.0	24.6	29.7	26.9	49.5	29.9	37.3
Hunspell	31.3	34.9	33.0	16.2	40.1	23.0	10.3	40.2	16.4	28.5	30.7	29.6
gpt-3.5-turbo-0301												
With Punctuation	55.8	75.3	64.1	33.8	72.1	46.0	53.7	66.1	59.3	43.8	57.0	49.6
W/O Punctuation	55.3	75.8	63.9	30.8	70.9	43.0	53.2	67.6	59.6	43.3	56.2	48.9
gpt-4-0314												
With Punctuation	57.0	75.9	65.1	34.0	73.2	46.4	54.2	67.7	60.2	44.2	57.4	50.0
W/O Punctuation	56.4	76.2	64.8	31.0	72.0	43.3	54.2	69.4	60.9	45.2	58.2	51.0
text-davinci-003												
With Punctuation	55.9	75.3	64.2	33.6	72.0	45.8	48.0	66.4	55.7	45.7	57.3	50.9
W/O Punctuation	55.4	75.8	64.0	31.2	71.1	43.4	47.8	68.4	56.3	46.5	58.1	51.7
M2M100-1.2B	88.8	71.5	79.2	63.8	61.1	62.4	78.8	71.4	74.9	47.1	42.9	44.9

Источник: <https://arxiv.org/pdf/2308.09435.pdf>
<https://habr.com/ru/companies/sberdevices/articles/763932/>

JamSpell: классический ML

Авторы добавляют две модели градиентного бустинга на решающих деревьях (CatBoost):

- Бинарный классификатор: модель решает, есть ошибка в слове или нет
- Регрессор: выбирает лучший вариант исправления слова

В качестве признаков они используют:

- Частота слова
- Наличие его в словаре; длина слова
- Частоты n-грам (2, 3)
- Частоты близлежащих слов с расстоянием 3 и 4
- Предсказание на основе n-грам
- Редакционное расстояние

Фреймворки

Классические:

- [HunSpell](#)
- [SymSpell](#)
- [DeepPavlov](#)

Продвинутые:

- [YaSpeller](#)
- [NeuSpell](#)
- [SAGE](#)
- [JamSpell](#)



Спасибо
за внимание!