Занятие 3

Информационный поиск

Катя Герасименко

Зимняя олимпиадная школа МФТИ

05.01.2019

Проект

Рекомендательная система новостей

Новости разбиты по темам – 20 тем

Можно считать, что мы рекомендуем похожие новости, но идеи расширения критериев приветствуются.

Можно за бонусные баллы:

- собрать / дособрать / доразметить данные, чтобы воплотить интересную идею
- расширить критерии релевантности новости
- написать интерфейс

Задача

1. По запросу на естественном языке (или его подобии – *купить цветы дешево москва*) найти релевантные документы из нашего корпуса
2. Предоставить выдачу документов, отсортированную по релевантности

Расширение понятия

На самом деле всё, о чем будет рассказано, можно расширить до задачи «найти по тексту похожие тексты», что может пригодиться вам в проекте.

Как найти релевантные документы?

1. По совпадающим словам: обратный индекс
2. Учитывать семантику: векторное представление текстов

Как отранжировать список?

- Самое простое – просуммировать значения **TF-IDF** слов запроса для каждого документа
- **Okapi BM25** + улучшения – текстовая метрика, основана на TF и IDF

Есть метрики, которые не зависят от запроса

- PageRank
- HITS

Если есть **оценки асессоров** – машинное обучение на разных признаках, включая эти (point-wise ranking, pair-wise ranking, разные метрики для оценки ранжирования (см. nDCG))

Поиск по совпадающим словам

Когда делаем систему:

1) предобработка документов
2) обратный индекс – для каждого слова сохраняем его наличие / частоту в разных документах

Когда ищем:

1) предобработка запроса
2) для каждого слова в запросе смотрим, в каких документах оно есть и с какой частотой

Обратный индекс

Индекс – для каждого документа знаем, какие слова в нем есть.

Doc1: i, like, cats

Doc2: dogs, are, better, than, cats

Обратный индекс – для каждого слова знаем, в каких документах оно есть.

i: Doc1
like: Doc1
cats: Doc1, Doc2

dogs: Doc2
are: Doc2
better: Doc2
than: Doc2

TF-IDF

t – отдельное слово, d – документ, D – корпус

TF — term frequency

$$\mathrm{tf}(t,d) = \frac{n_t}{\sum_k n_k}$$

n_t — количество вхождений t в d

$\sum_k n_k$ — общее число слов в d.

IDF — inverse document frequency

$$\mathrm{idf}(t,D) = \log \frac{|D|}{|\{d_i \in D \mid t \in d_i\}|}$$

$|D|$ — количество документов в корпусе

$|\{d_i \in D \mid t \in d_i\}|$ — количество документов в корпусе, в которых встречается t

Продвинутый IDF для Okapi BM25 (будет на след. слайде)

$$\mathrm{idf}(t,D) = \log \frac{|D| - n(t) + 0.5}{n(t) + 0.5}, \text{где } n(t) = \{d_i \in D \mid t \in d_i\}$$

Если слово встречается больше чем в половине документов, логарифм отрицательный, что нехорошо

1) игнорировать, обнулять
2) поставить нижний порог IDF

TF-IDF

TF-IDF

$$\text{tf}-\text{idf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

Сакральный смысл – если слово часто встречается в одном документе, но в целом по корпусу встречается в небольшом количестве документов, у него высокий TF-IDF

Okapi BM25

$$\text{score}(d, Q) = \sum_{i=1}^{n} \text{idf}(q_i) \cdot \frac{\text{qf}(q_i, d) \cdot (k_1 + 1)}{\text{qf}(q_i, d) + k_1 \cdot (1 - b + b \cdot \frac{|\text{d}|}{\text{avgdl}})}$$

продвинутый

Q – запрос, q_i – отдельное слово в запросе

d – конкретный документ

qf(q_i, d) – частота q_i в d

avgdl – средняя длина документа в корпусе

k_1, b – свободные параметры, их обычные значения:

$k_1 \in [1.2, 2]$

b = 0.75

Что хорошо, а что плохо

Хорошо:

- просто и быстро считать
- может сработать когда мало данных

Плохо:

- это bag-of-words – мы кладем слова в один мешок. Нет учета контекста, нет семантики, ничего нет, кроме тех слов что мы ему дали

Поиск с учетом семантики

1. Для каждого документа создать вектор фиксированной длины, отражающий смысл документа
   - word2vec и другие эмбеддинги
   - методы тематического моделирования
2. Для запроса вычислить такой же вектор
3. Сравнивать близость векторов (например, косинусная близость, cosine similarity)
4. Ранжировать по близости

Поиск с учетом всего

Когда есть какая-либо информация о похожести / релевантности текстов, можно эту информацию использовать в качестве ответов и обучить что-нибудь supervised с любыми признаками.

У поисковиков – данные о поведении пользователей (клики; время, проведенное на странице, etc.)

Как это сейчас

Поиск документов по запросу с помощью всего, что можно, включая семантические вектора и нейросети

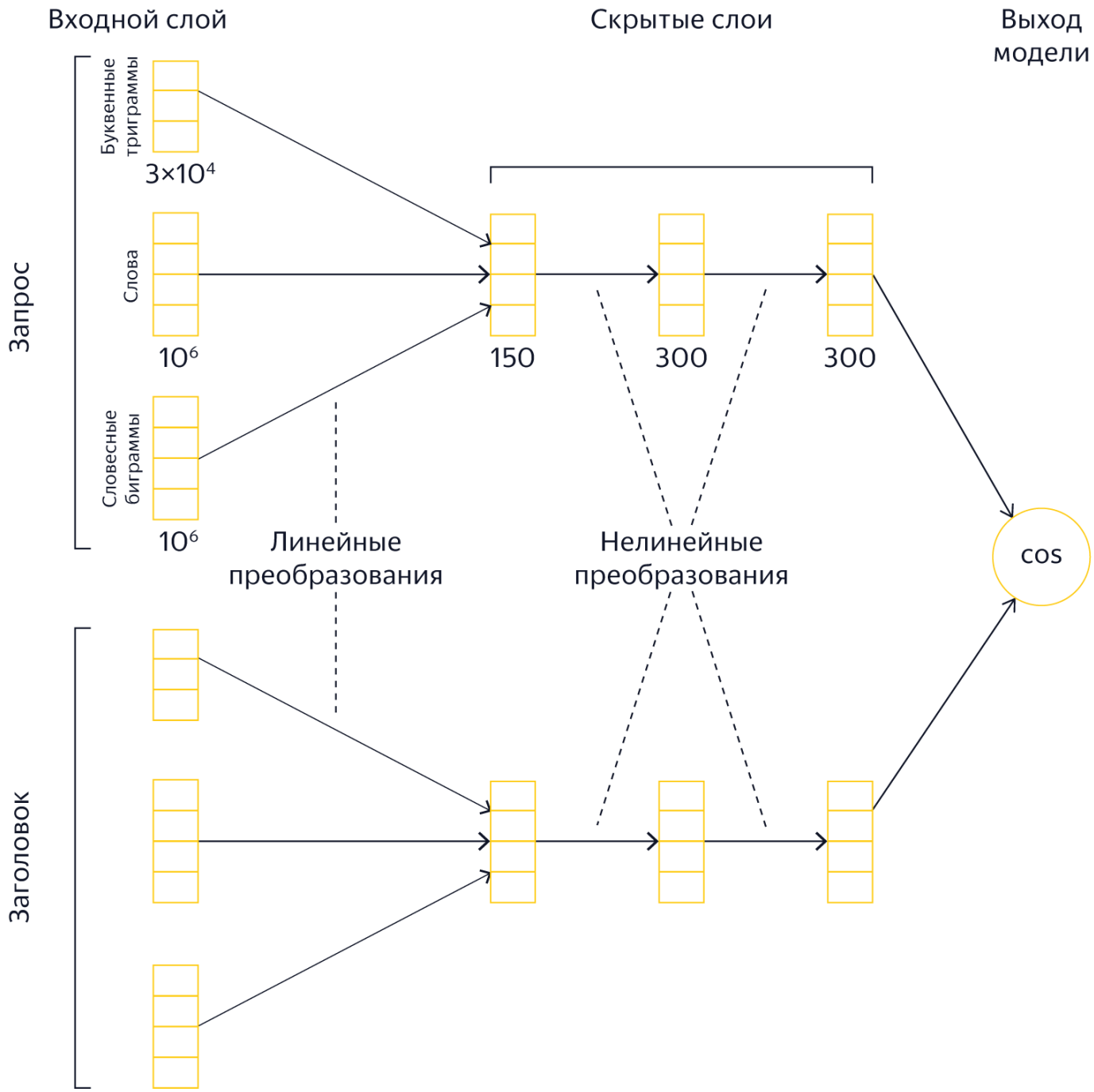
+ учет поведения пользователей

-> Яндекс – «Королёв» (август 2017)

Ранжирование документов с помощью огромной формулы ранжирования и МО

+ асессоры / краудсорсинг

-> Яндекс – «Матрикснет» (с 2009)


Входной слой
Скрытые слои
Выход модели
Запрос
Буквенные триграммы
3×10⁴
Слова
10⁶
Словесные биграммы
10⁶
150
300
300
Линейные преобразования
Нелинейные преобразования
cos
Заголовок