Word Mover's Distance

Ксения Вальчук, Анастасия Царькова

НИУ ВШЭ

21 марта 2017 г.

Постановка задачи

Рассмотрим задачу определения степени схожести текстов.
Примеры применения:

- Классификация текстов (новостей)
- Поисковое ранжирование
- Рекомендации книг

Способы представления документов. BOW

• Bag of words(BOW) – упрощенное представление текста без учета грамматики и порядка слов, но сохраняя кратность.

Пример: John likes to watch movies and watch tv. – ['John', 'likes', 'watch', 'movies', 'tv']

BOW = [1, 1, 2, 1, 1]

Способы представления документов. TF-IDF

• TF-IDF (term frequency-inverse document frequency) – статистическая мера, для оценки важности слова в контексте документа, из некоторой коллекци документов D.
Для каждой пары (слово, текст) (t, d) вычисляется величина

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

,где

$$\text{tf}(t, d) = \frac{n_{td}}{\sum_{t \in d} n_{td}}$$

– количество вхождений слова t в отношении к общему числу слов,

$$\text{idf}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

– инверсия частоты, с которой слово встречается в документах коллекции.

word2vec

Модель, которая сопоставляет каждому слову его векторное представление путем максимизации логарифма вероятности нахождения слова рядом со своими соседями

$$max \frac{1}{T} \sum_{t=1}^{T} \sum_{j \in nb(t)} \log p(w_j | w_t)$$

- Имеет простую архитектуру
- Способна быстро обучаться на огромных датасетах
- Обучается сложным зависимостям
  Пример: vec(Einstein) - vec(scientist) + vec(Picasso) $\approx$ vec(painter)

Способы представления документов. Недостатки

Пусть есть текст, который состоит из n уникальных слов. Представим его в виде нормированного мешка слов (nBOW).

$$d \in \mathbb{R}^n, \; d_i = \frac{c_i}{\sum_{j=1}^{n} c_i}$$

где c_i – количество раз, которое слово i встречается в данном тексте. Так как d – нормирован, представим его как точку в $(n-1)$ мерном симплексе.

Способы представления документов. Недостатки

Рассмотрим пример предложений, которые не имеют общих слов, но подразумевают схожую информацию:

- D = Obama speaks to the media in Illinois.
- D' = The President greets the press in Chicago.

$w = [' Obama', ' speaks', ' media', ' Illinois', ' President', ' greets', ' press', ' Chicago']$, тогда получаем мешки слов

$$d = [1/8, 1/8, 1/8, 1/8, 0, 0, 0, 0]$$

$$d' = [0, 0, 0, 0, 1/8, 1/8, 1/8, 1/8]$$

Word Mover's Distance

Word Mover's Distance

WMD определяет расстояние между двумя документами как оптимальную стоимость перемещения слов из одного документа в другой с помощью векторного представления слов.

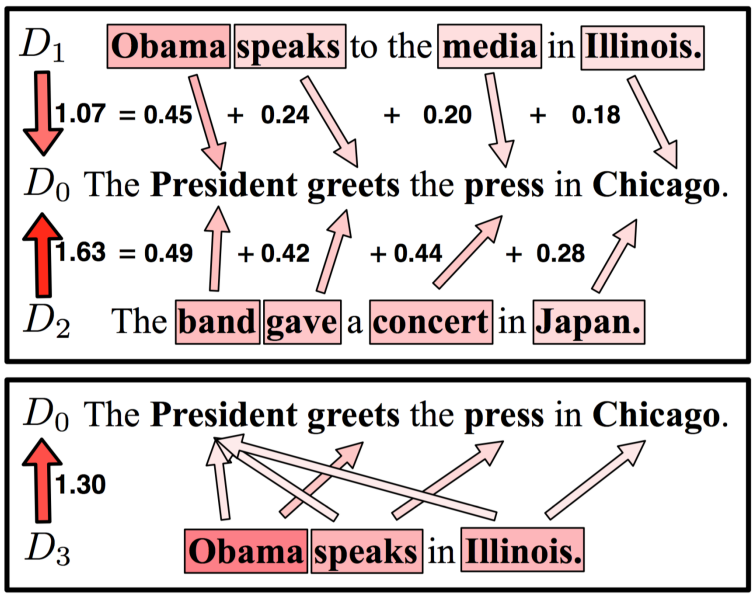
Traveling cost

Пусть у нас есть матрица $X \in \mathbb{R}^{dxn}$, где $x_i \in \mathbb{R}^d$ – векторное представление i-ого слова конечной длины.
Тогда 'семантическое' расстояние между словами i и j определим как

$$c(i,j) = \|x_i - x_j\|_2$$

Flow matrix

Пусть у нас есть два текста D и D' с нормированными мешками слов d, d' соответственно.
Определим матрицу перемещений $T \in \mathbb{R}^{n \times n}$, где $T_{ij} \geq 0$ – 'количество' слова i из d которое переходит в слово j из d'. При этом должно выполняться $\sum_j T_{ij} = d_i$ и $\sum_i T_{ij} = d'_j$



Оптимизационная задача

Таким образом мы можем определить расстояние между текстами как минимальную совокупную стоимость перехода требуемую для того чтобы перевести d в d'.
Формально данная задача минимизации стоимости перехода из d в d' записывается следующим образом:

$$\min_{T \geq 0} \sum_{i,j} T_{ij} \|x_i - x_j\|_2$$

$$\sum_{j=1}^{n} T_{ij} = d_i \ \forall i \in 1, ..., n$$

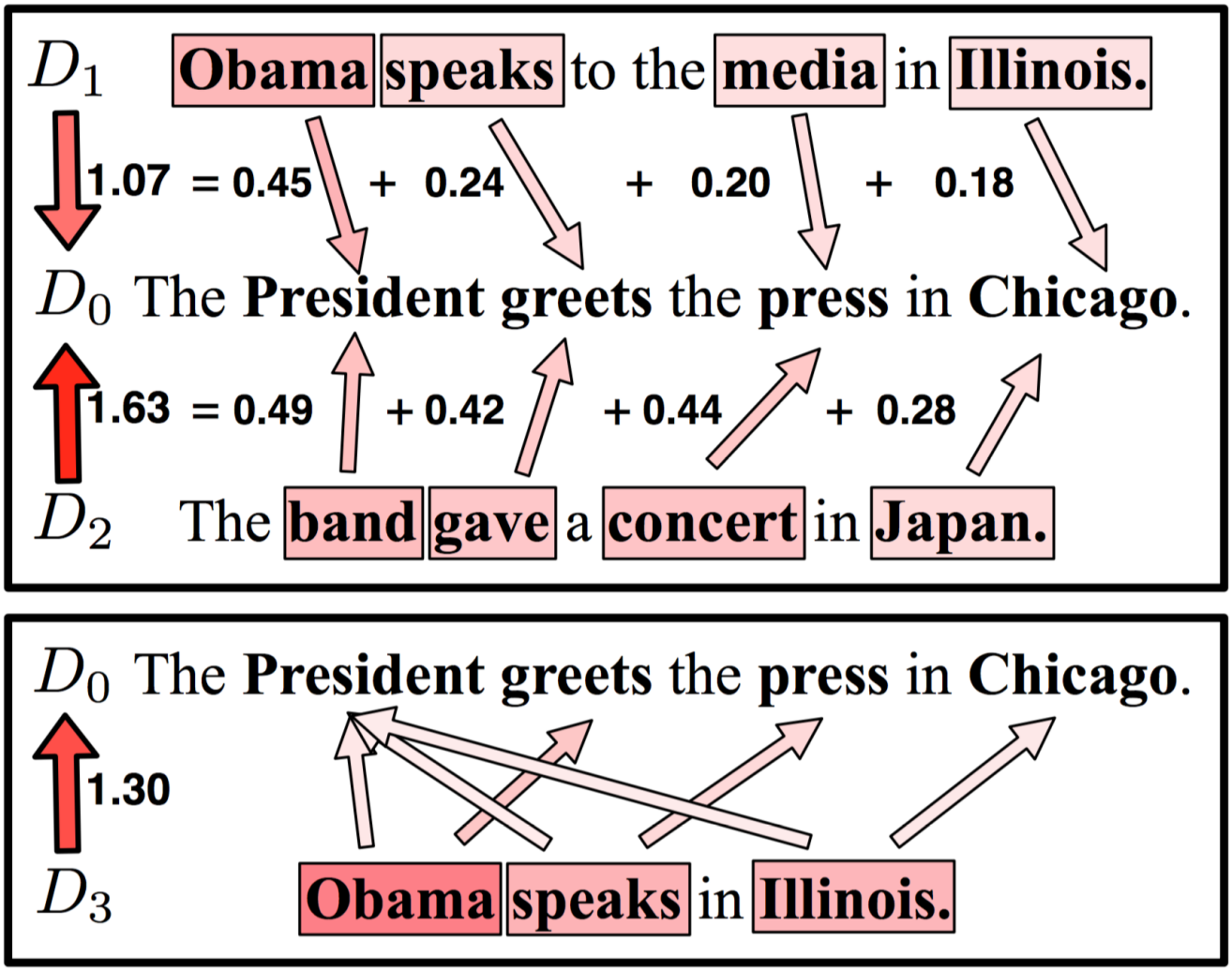
$$\sum_{i=1}^{n} T_{ij} = d_j \ \forall j \in 1, ..., n$$

Word Mover's Distance

Свойства метрики:

- Не имеет гиперпараметров, проста для понимания и использования
- Демонстрирует высокие показатели метрики accuracy на реальных задачах классификации текстов

Пример работы метрики



Сложность

Решение данной задачи оптимизации имеет сложность $O(p^3 logp)$, где p – количество различных слов в тексте. В больших датасетах количество уникальных слов велико – что делает вычисление метрики долгим.
Поэтому надо как то оценивать WMD, что бы уменьшить сложность алгоритма.

Word centroid distance

Попробуем оценить значение метрики снизу:

$$\sum_{i,j=1}^{n} T_{ij} c(i,j) = \sum_{i,j=1}^{n} T_{ij} \|x_i - x'_j\|_2 = \sum_{i,j=1}^{n} \| T_{ij}(x_i - x'_j)\|_2 \geq$$

$$\geq \| \sum_{i,j=1}^{n} T_{ij}(x_i - x'_j)\|_2 = \| \sum_{i=1}^{n} \left(\sum_{j=1}^{n} T_{ij} \right) x_i - \sum_{j=1}^{n} \left(\sum_{i=1}^{n} T_{ij} \right) x'_j \|_2 =$$

$$= \| \sum_{i=1}^{n} d_i x_i - \sum_{j=1}^{n} d_j x'_j \|_2 = \|Xd - Xd'\|_2$$

Сложность вычисления данной метрики $O(dp)$, где d – размерность пространства word2vec.

Relaxed word moving distance

Несмотря на то, что WCD можно посчитать эффективно по времени, она не очень хорошо оценивает нашу метрику.
Ослабим условия задачи

$$\min_{T \geq 0} \sum_{i,j} T_{ij} c(i,j)$$

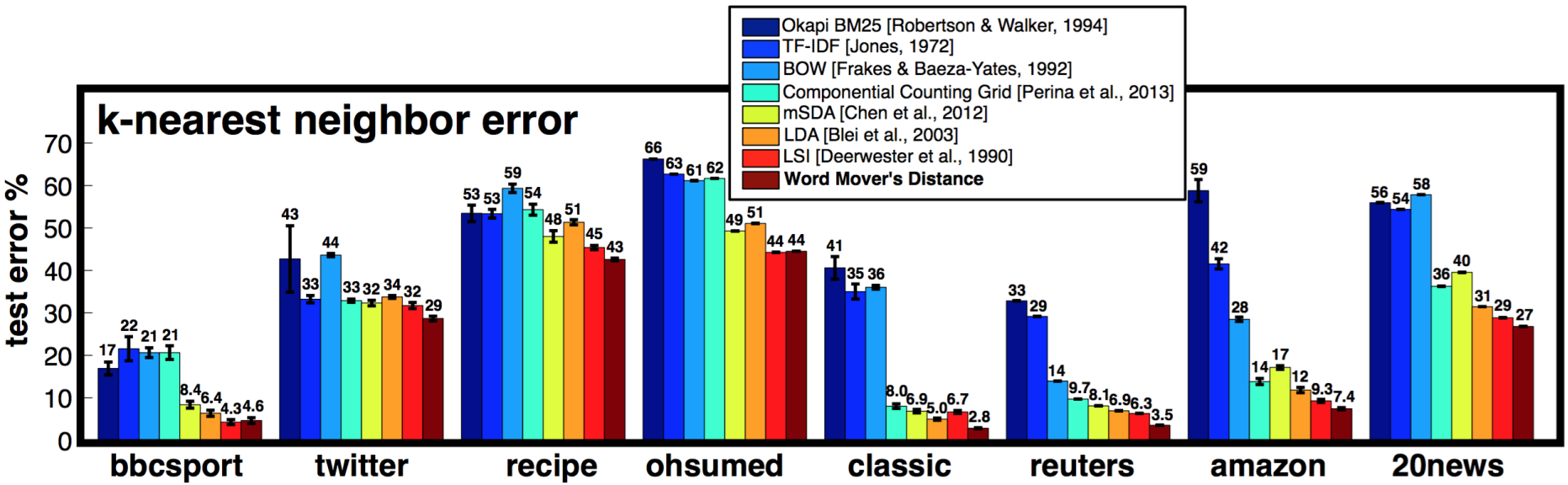
$$\sum_{j=1}^{n} T_{ij} = d_i \forall i \in 1, ..., n$$

Решение полученной задачи оптимизации:

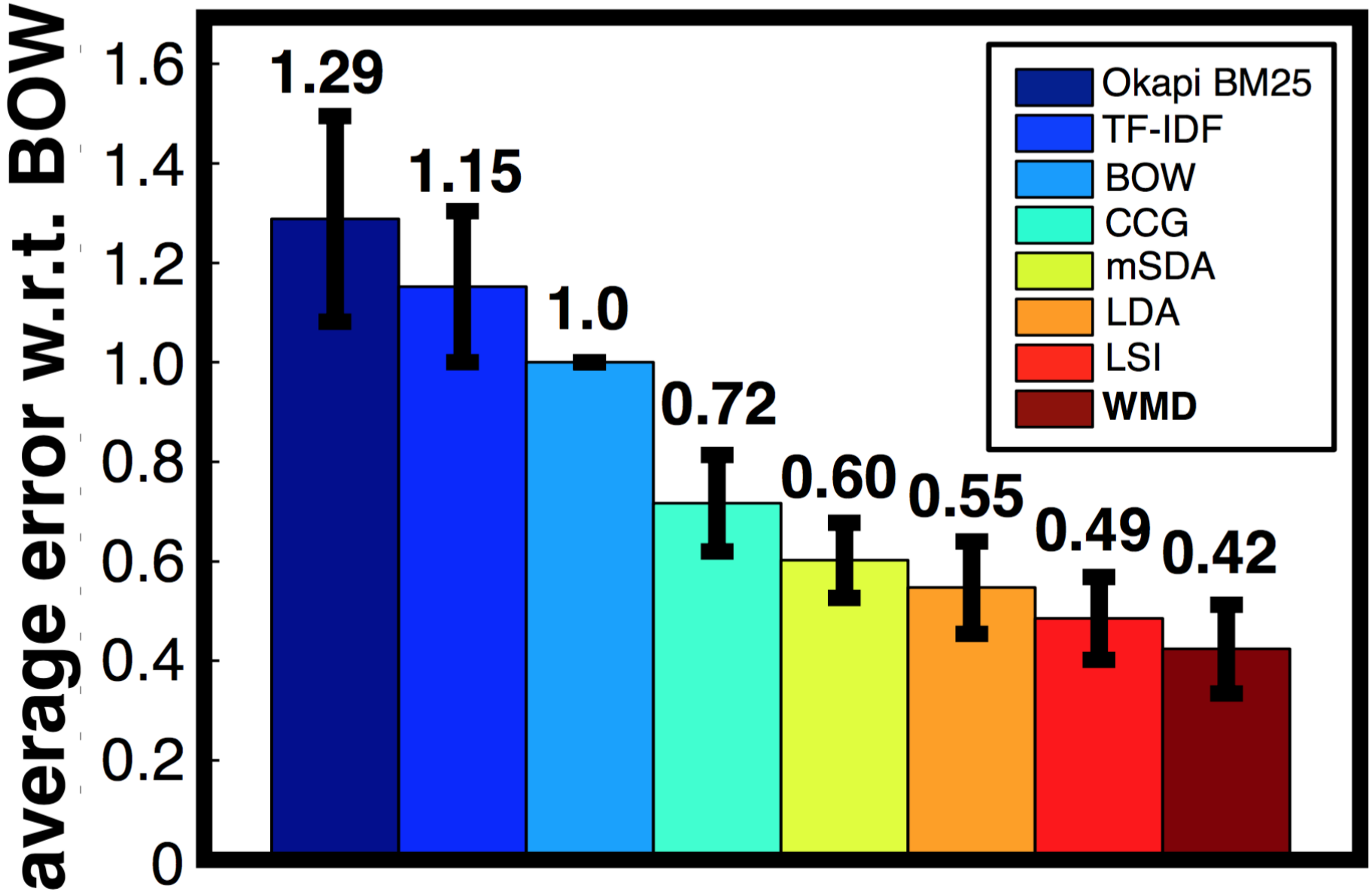
$$T^*_{ij} = \begin{cases} d_i \ \textit{if} \ j = \text{argmin}_j c(i,j) \\ 0, \ \textit{otherwise} \end{cases}$$

$$\sum_j T_{ij} c(i,j) \geq \sum_j T_{ij} c(i,j^*) = c(i,j^*) \sum_j T_{ij} = c(i,j^*) d_i = \sum_j T^*_{ij} c(i,j)$$

Классификация документов



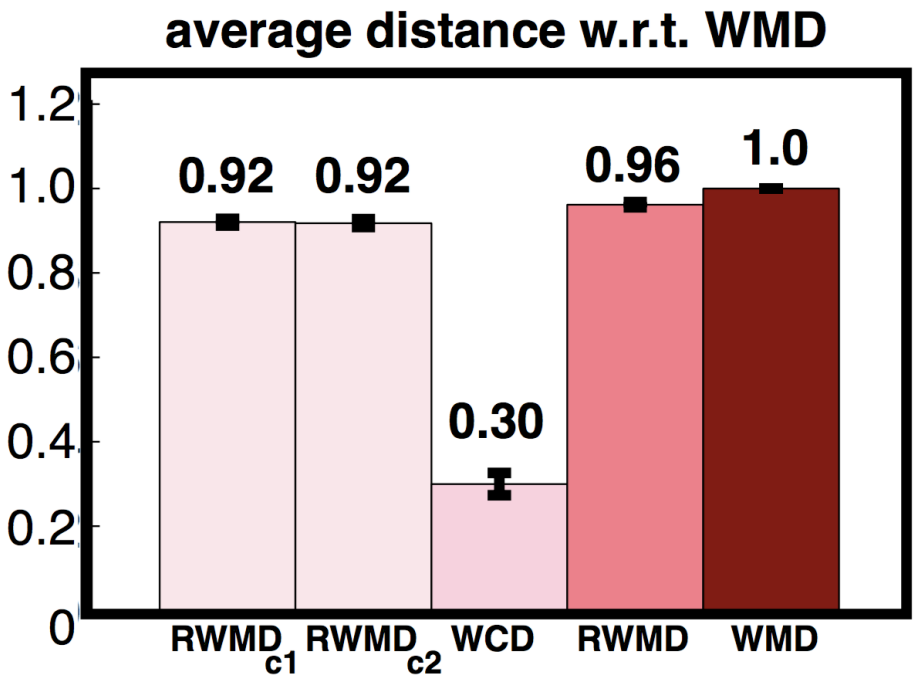
Классификация документов

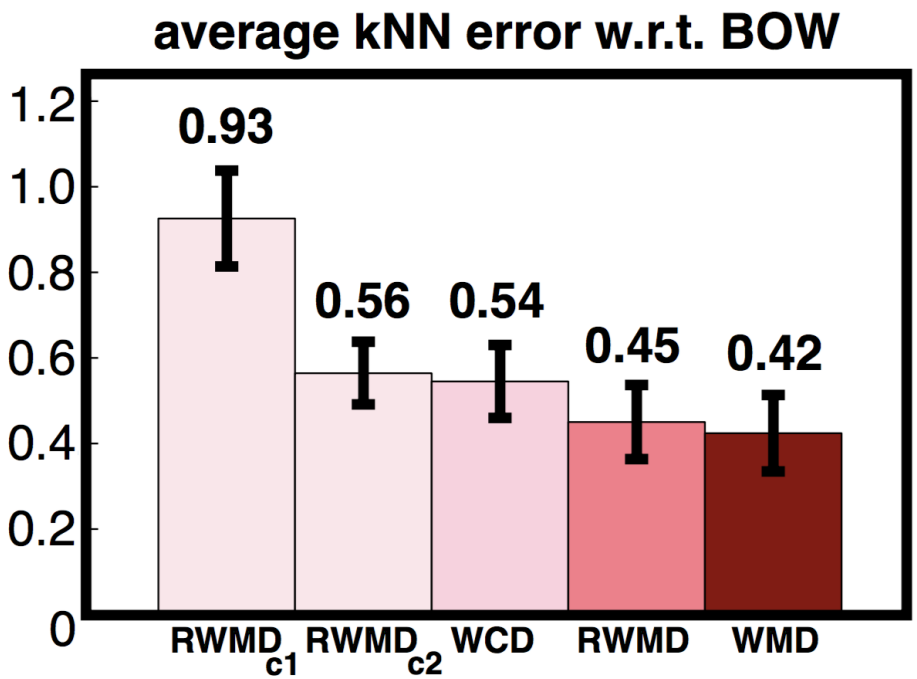


Оптимизация kNN

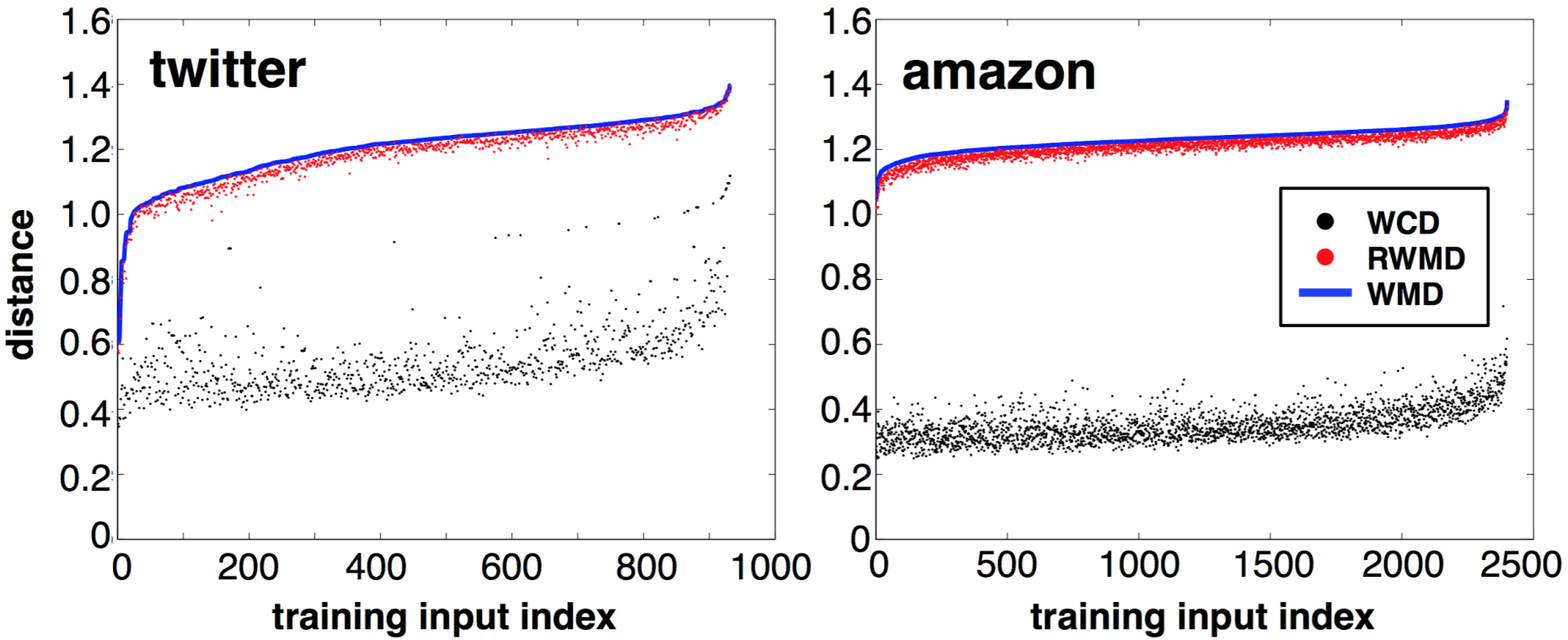
Алгоритм:
1. Сортим все документы по возрастанию метрики WCD
2. Вычисляем WMD для первых k документа из предыдущего шага
3. Проходимся по каждому документу с k по k + m и если RWMD данного документа меньше WMD k-ого, то добавляем его в k ближних, считает для него WMD, сортим все k + 1 и отбираем из них лучший.
4. Повторяем шаг 3 пока не перестанем находить документ лучший k-ого
ч

Оптимизация kNN

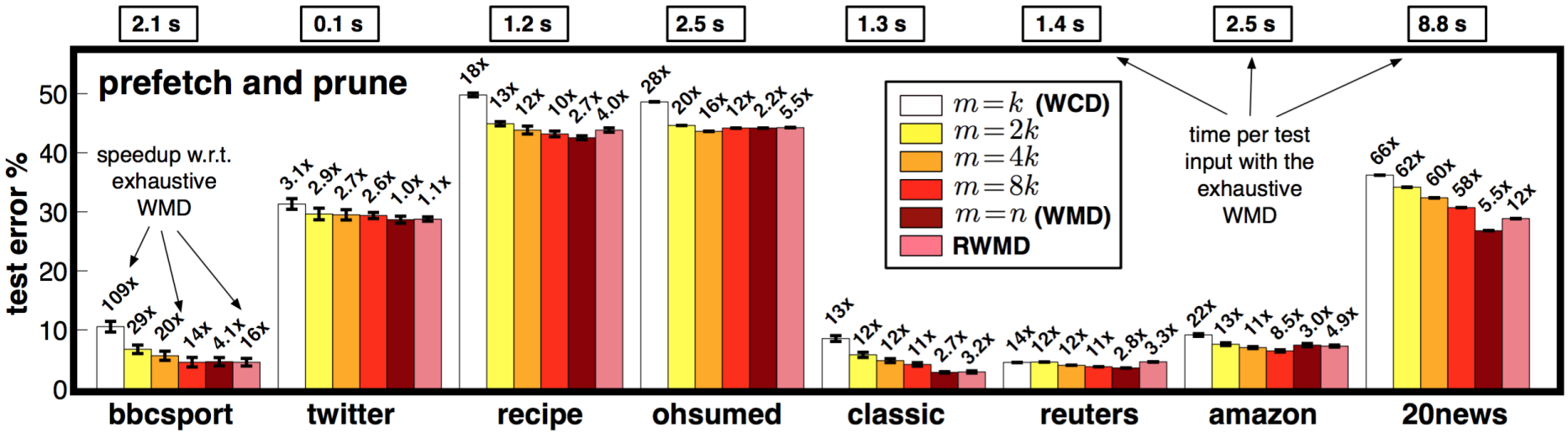




Оптимизация kNN



Оптимизация kNN



Спасибо за внимание =)