Word Embeddings

Sergey Aksenov

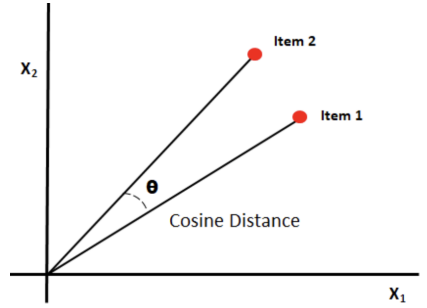
Higher School of Economics

19 сентября 2022 г.

Метрики

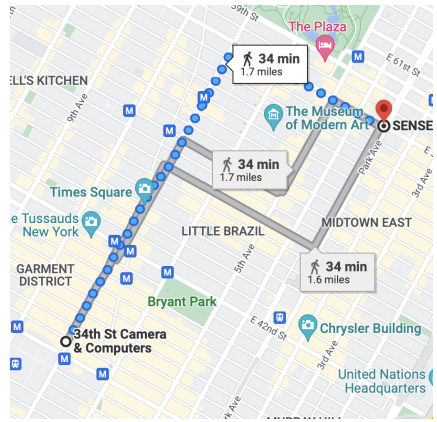
Cosine similarity

$$Sim(p, q) = \frac{\sum_i p_i q_i}{\sqrt{\sum_i p_i^2}\sqrt{\sum_i q_i^2}}$$



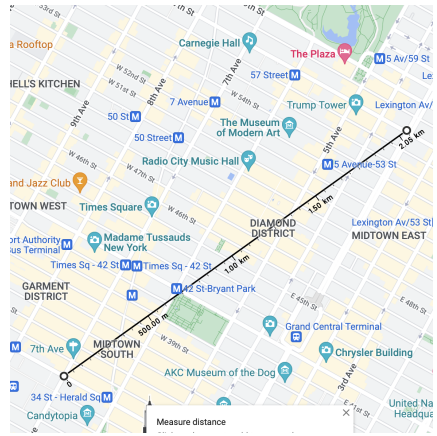
Manhattan distance

$$d(p, q) = \sum_i |p_i - q_i|$$



Euclidean distance

$$d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$$



One-hot кодирование

Преимущества

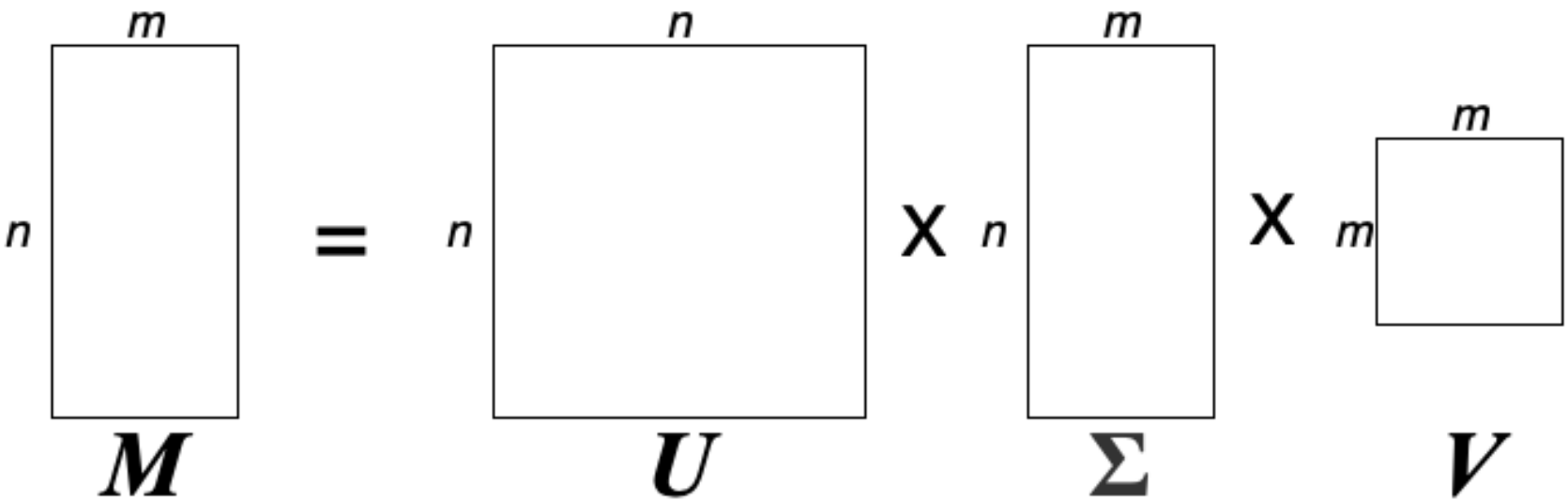
- Простой способ получить какие-то вектора
-
-

Недостатки

- Длина векторов не фиксирована
- Векторы ортогональны
- Не сохраняет смысл слов

Сингулярное разложение

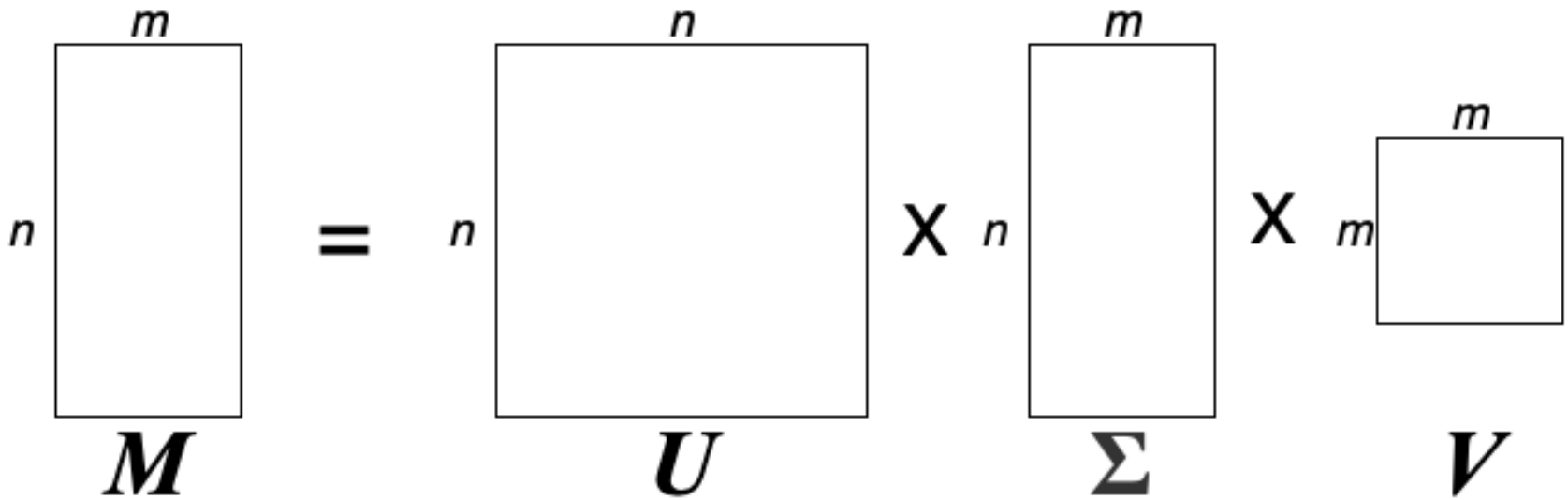
- $d \in D, |D| = m$ – документы
- $w \in V, |V| = n$ – слова
- $M_{ij} = \#w_i \in d_j$



Сингулярное разложение

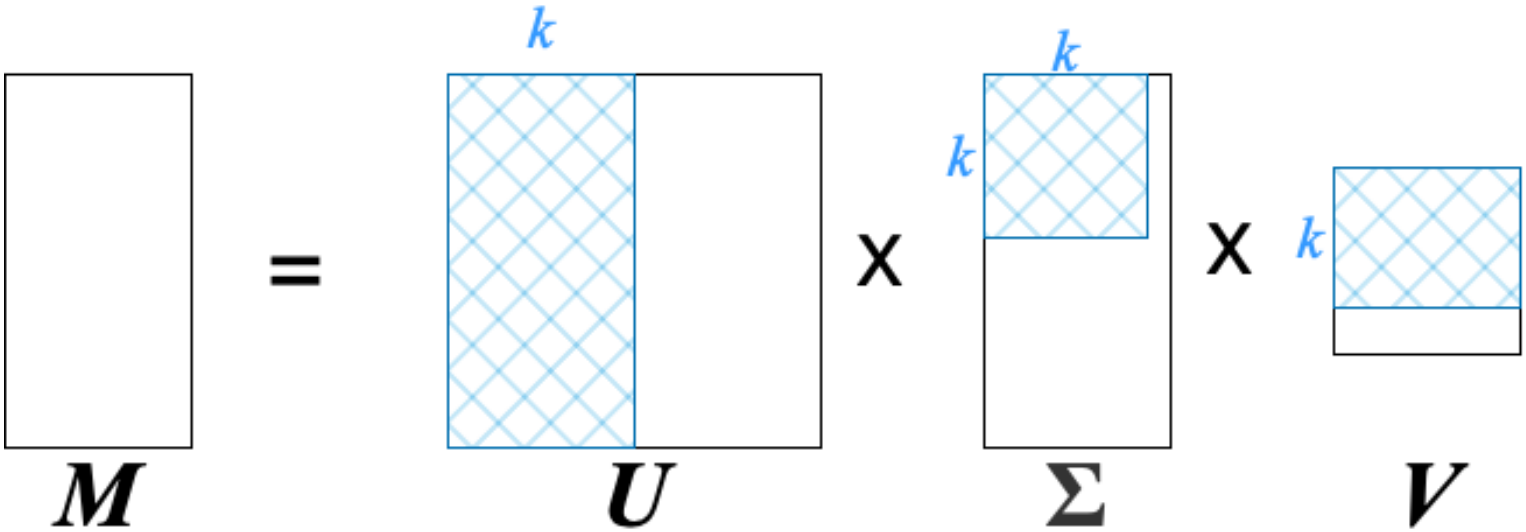
Сингулярное разложение матрицы M

$$M = U\Sigma V^T$$



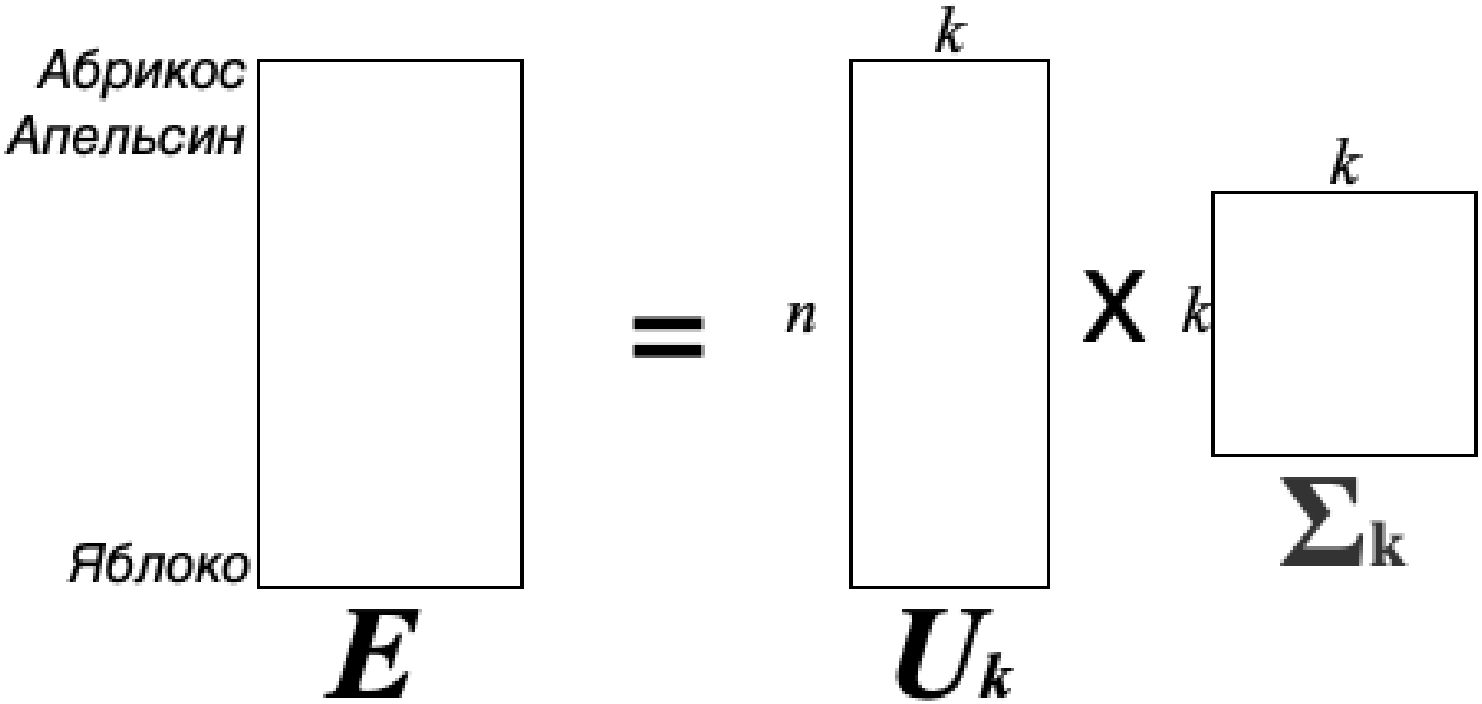
Сингулярное разложение

- Снижение размерности: выберем k наибольших сингулярных чисел в Σ
- Выберем k первых столбцов и строк в U и V^T – U_k и V_k^T



Сингулярное разложение

Каждому слову поставим в соответствие вектор $U_k\sqrt{\Sigma_k}$



Сингулярное разложение

Что стало лучше

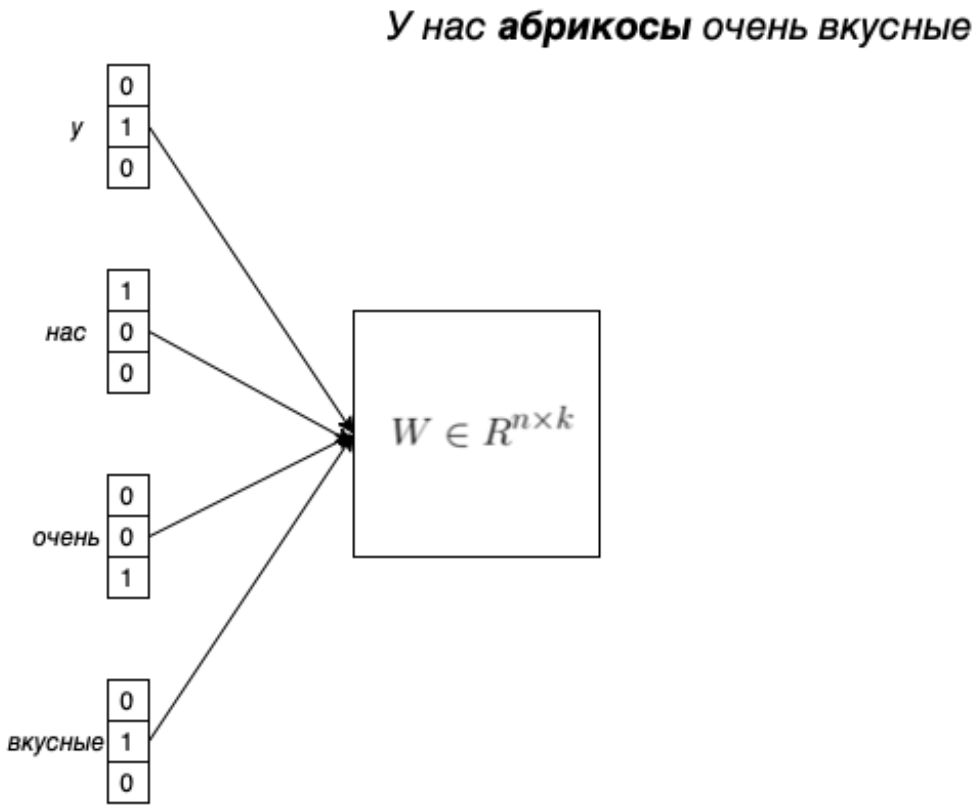
- Векторы имеют фиксированную длину
- Векторы не ортогональны друг другу
- Семантическая близость как-то учитывается

Недостатки

- Нужно в явном виде работать с большой разреженной матрицей
- Добавление новых слов или документов невозможно без построения нового разложения

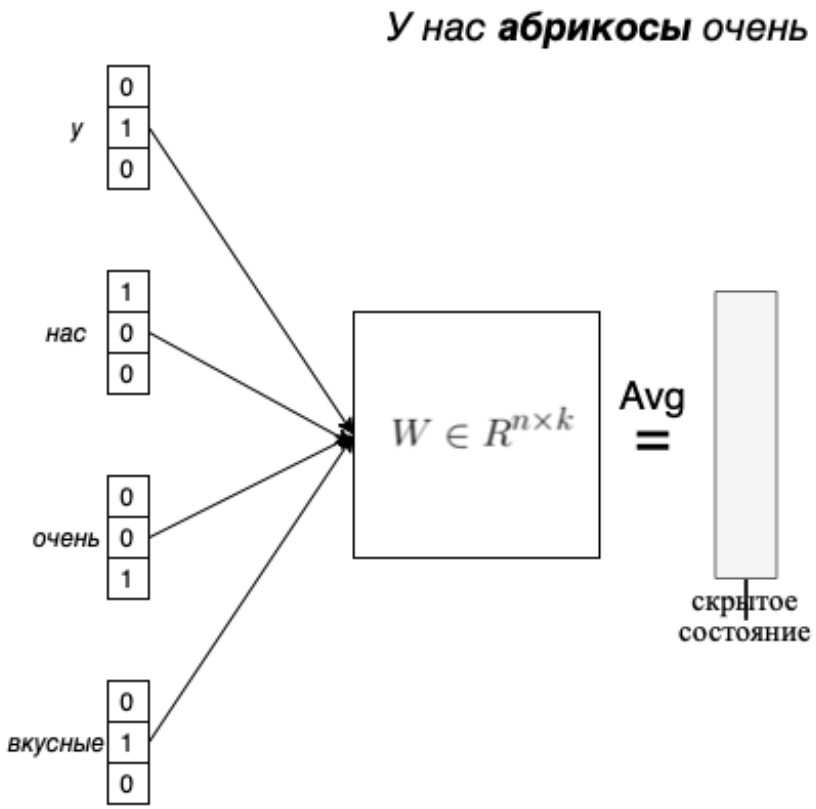
Модель CBOW

- Разобьем корпус текста на контексты размера $2C + 1$
- Вход нейронной сети: 2 векторов контекста, размерность каждого вектра - n



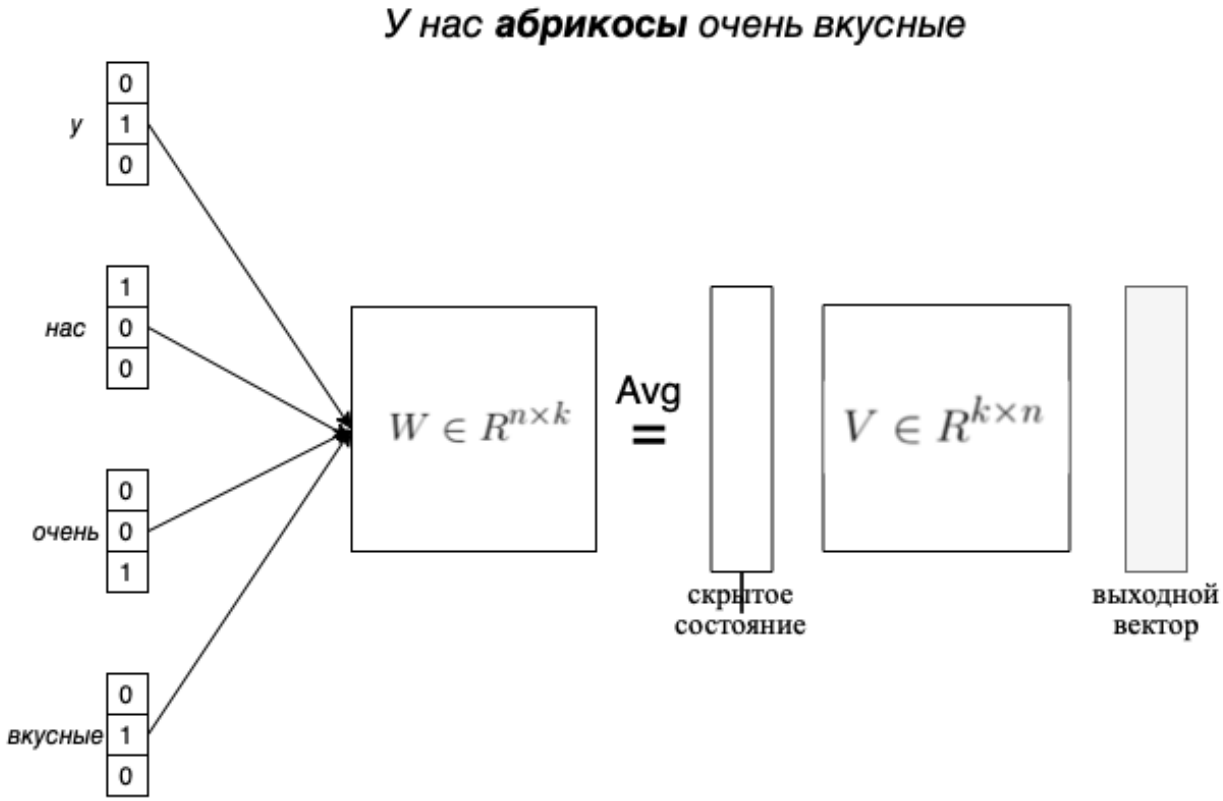
Модель CBOW

- Применим ко входу преобразование W и усредним результаты
- Получим $k-$мерный вектор скрытого состояния $h = \frac{1}{4} W^T (a_{i-2} + a_{i-1} + a_{i+1} + a_{i+2})$



Модель CBOW

- Умножим скрытое состояние на вторую матрицу и получим $n-$мерный выходной вектор $b = V^T h$



Модель CBOW

Оптимизационная задача

- Для $i-$го окна по контексту c_i предсказать центральное слово w_i:

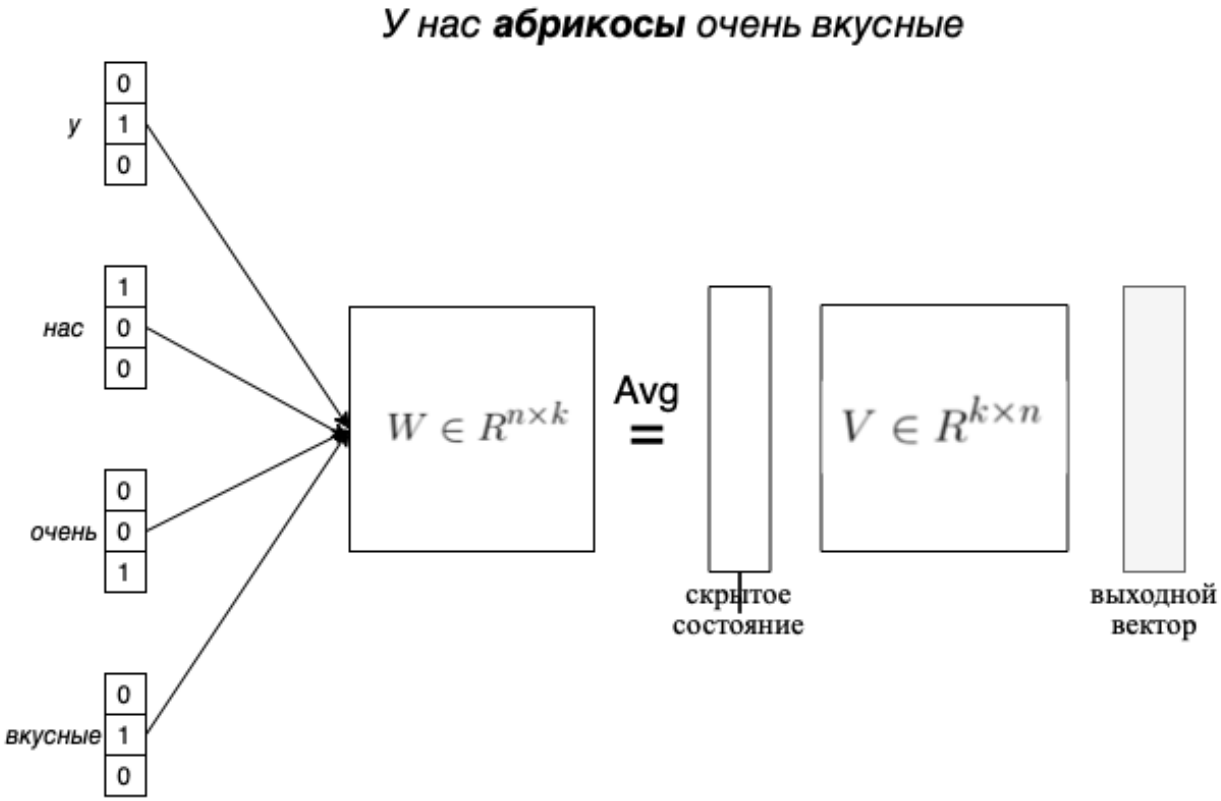
$$\sum_i logp(w_i|c_i) \to max_{W,V}$$

- Выход последнего слоя сети: $softmax(b) = (z_i, ..., z_j)$, где

$$z_j = p(w_j|c_i) = \frac{e^{b_j}}{\sum_k e^{b_k}}$$

Модель CBOW

- Получаем две матрицы весов: W и V
- Обычно матрицу W считают матрицей искомых векторов



Модель Skip-gram

- Разобьем корпус на контексты размера $2C + 1$
- Вход нейронной сети: $n-$мерный one-hot вектор центрального слова
- Вектор скрытого состояния $h = W^T x$

Модель Skip-gram

- Выход нейронной сети: вероятностное распределение слова контекста при условии центрального слова
- Выходной вектор $b = V^T h$

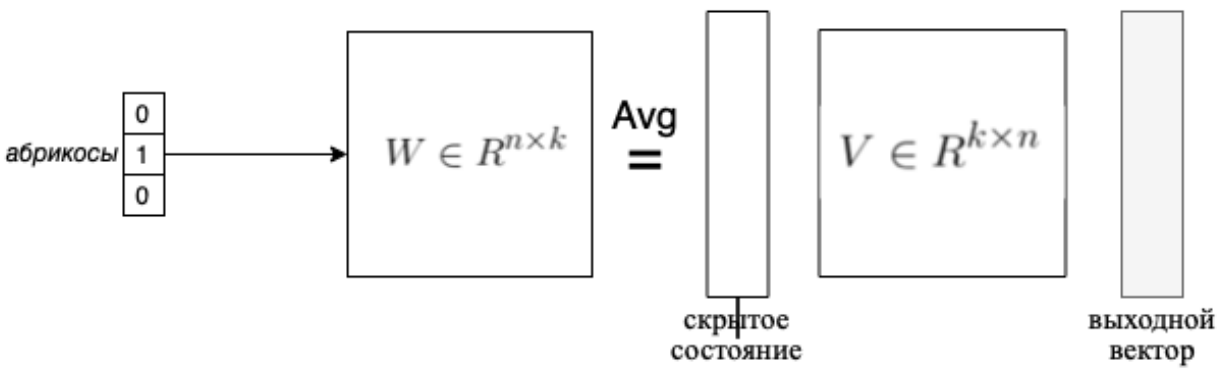
Модель Skip-gram

Оптимизационная задача

- Для $i-$го центрального слова предсказать слово контекста c_j

$$\sum_i \sum_j log(c_j|w_i) \to max_{W,V}$$

*У нас **абрикосы** очень вкусные*



Обучение моделей Word2Vec

Проблема: вычисление *softmax* на выходном слое требует прохода по всему словарю - $O(n)$ операций, вычислительно трудная задача

Методы сокращения перебра

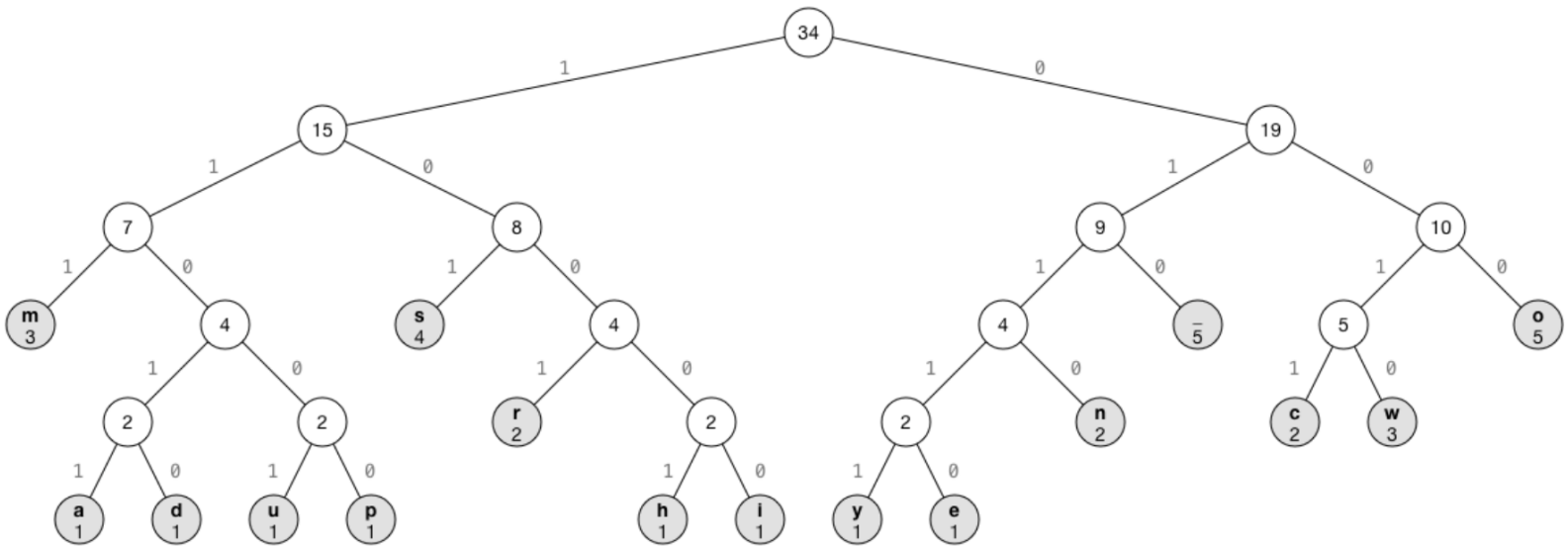
- Иерархический *softmax*
- Noise Constructive Estimation (NCE)
- Negative sampling (NS)

Иерархический softmax

- Будем оценивать только значения, стоящие в позициях предсказываемых слов контекста

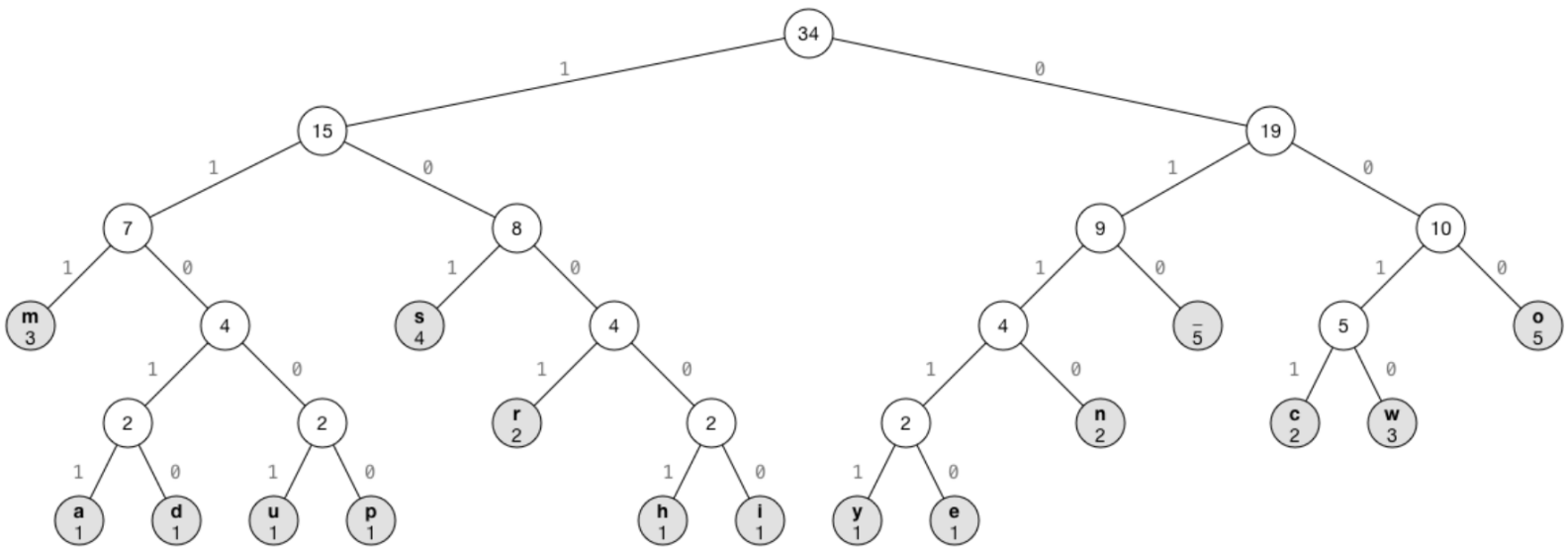
$$\sum_i \sum_j log(c_j|w_i) \rightarrow max_{W,V}$$

- Заменим второй слой сети на дерево Хаффмана



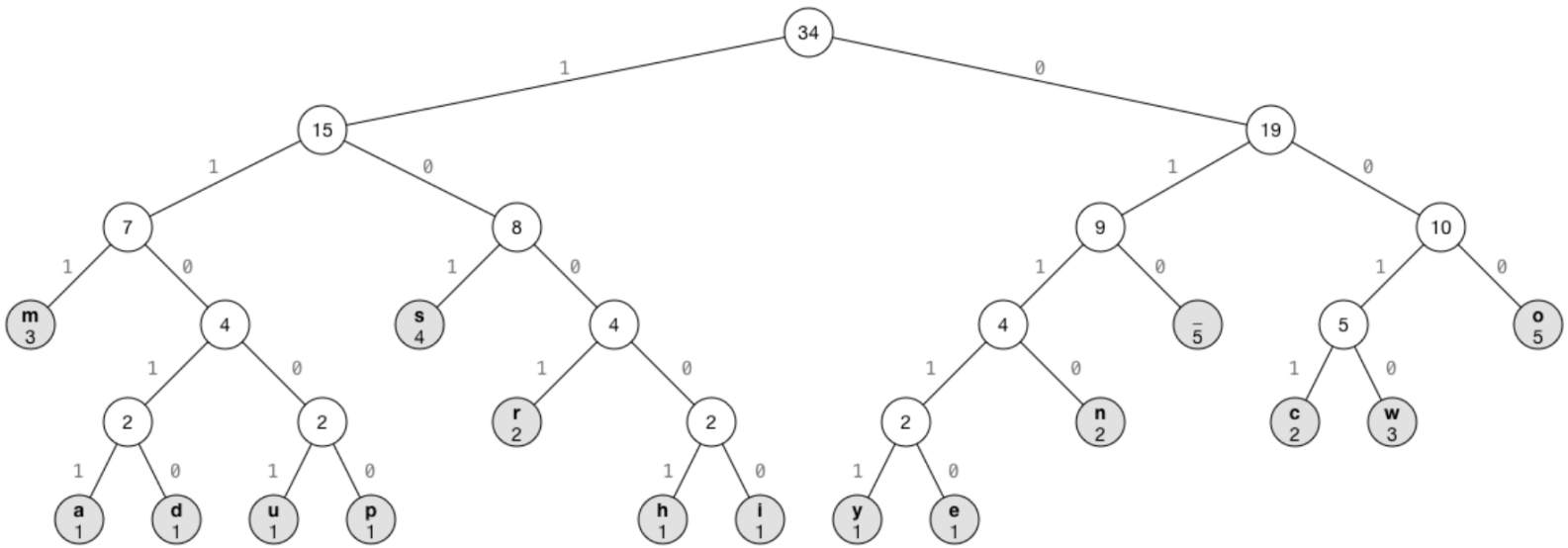
Иерархический softmax

- Пусть мы хотим оценить вероятность слова *"лось"*
- Вероятность перехода из текущей вершины в поддеревья:
  - $p_0 = \sigma(u^T_{n(w,0)}, h)$, если спускаемся по левой ветке
  - $p_0 = \sigma(-u^T_{n(w,0)}, h)$, если спускаемся по правой ветке



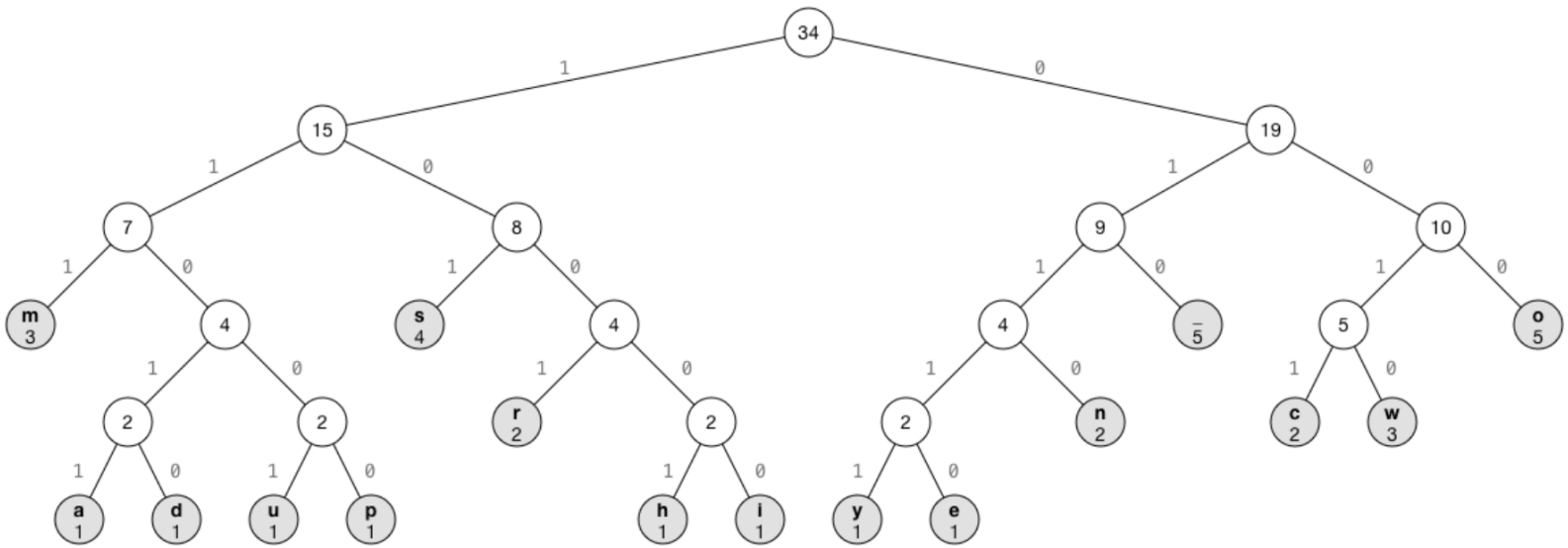
Иерархический softmax

- В результате мы окажемся в целевом листе
- Каждый $i-$й спуск происходил с вероятностью p_i
- Итоговая вероятность равна $\Pi_i p_i$



Иерархический softmax

- Пройдем по дереву $2C$ раз, получим все вероятности, необходимые для подсчета ошибки и настройки весов
- Сложность спуска по дереву $O(nlogn)$



Negative sampling

- Сформулируем задачу skip-gram как задачу бинарной классификации
- Рассмотрим пару слов (w, s) :
  - Класс 0: s не входит в контекст w
  - Класс 1: s входит в контекст w
- Вероятность класса $p(1|(w, s)) = \frac{1}{1+e^{-\nu_w^t \nu_s}} = \sigma(\nu_w^t, \nu_s)$

Negative sampling

- Пусть D_1 – множество наблюдаемых пар (w, s)
- D_0 – множество ненаблюдаемых пар (w, s), на каждой итерации обучения выбираем случайные пары
- Функционал праводоподобия

$$L = \sum_{w,s \in D_1} log(\sigma(\nu_w^t \nu_s)) + \sum_{w,s \in D_0} log(\sigma(-\nu_w^t \nu_s))$$

Модели word2vec

Обучение

- Ускорение обучения за счет иерархического softmax и negative sampling
- Вариант по умолчанию: SGNS (skip-gram with negative sampling)

Модели word2vec

Обучение

- Ускорение обучения за счет иерархического softmax и negative sampling
- Вариант по умолчанию: SGNS (skip-gram with negative sampling)

Недостатки

- Не учитывается морфология
- Нет возможности работать с OOV-словами

Модель FastText

- Разобьем слова на последовательности символов – символьные N-граммы
- Обучим вектора для каждой N-грамы с помощью CBOW или Skip-gram
- Вектор слова получим усреднением N-грамм