

# Лекция №1

Эмбеддинг — векторное представление слова, которое хранит в себе информацию о слове.

Способы составления эмбеддингов:

## One - Hot - Encoding

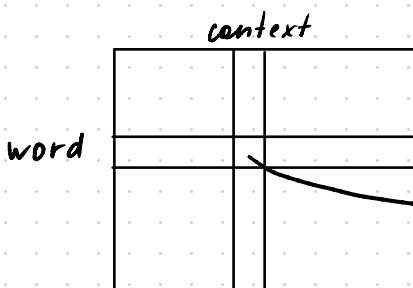
Составление катдему токену вектор из 0 и 1.

Минусы:

- очень большая размерность эмбеддингов
- такие векторы не отображают семантическую информацию (все векторы ортогональны)

	1	2	3	4	5	6	7	8
I	1	0	0	0	0	0	0	0
ate	0	1	0	0	0	0	0	0
an	0	0	1	0	0	0	0	0
apple	0	0	0	1	0	0	0	0
and	0	0	0	0	1	0	0	0
played	0	0	0	0	0	1	0	0
the	0	0	0	0	0	0	1	0
piano	0	0	0	0	0	0	0	1

## Count - based Methods



Идея: Задать в эмбеддинге информацию о контексте на основе статистики корпуса.

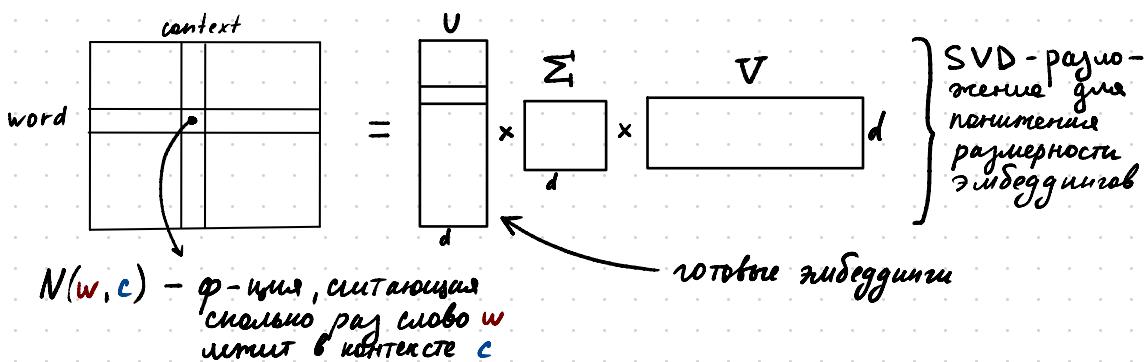
нападать то, насколько слово соответствует контексту

Как посчитать и-чью words-context?

### 1) Co-occurrence count

КОНТЕКСТ - слова, стоящие в окне конкретного слова

"Каждый охотник тешает **знат**, где спит **фазан**."  
контекст для  
слова с окном  
размера 2



### 2) TF - IDF (Term Frequency - Inverse Document Frequency)

КОНТЕКСТ - документ

$$\text{tf-idf}(w, d) = \frac{n_w}{\sum_k n_k} \cdot \log \frac{\text{idf}}{|\{d_i \in D | w \in d_i\}|}$$

$w$  - слово

$d$  - документ

$n_w$  - как-то раз слово  $w$  встречалось в  $d$

$\sum_k n_k$  - как-то слово  $w$  в  $d$

$D$  - все документы

$\{d_i \in D | w \in d_i\}$  - документы, в которых встречалось слово  $w$

### 3) PPMI (Positive Pointwise Mutual Information)

КОНТЕКСТ - слова, стоящие в окне конкретного слова

$$PPMI = \max(0, PMI(w, c))$$

$$PMI = \log \frac{P(w, c)}{P(w) \cdot P(c)}$$

### Prediction-based Methods

#### 1) Word2Vec

"Каждый охотник жаждет знать, где сидит фазан."

Для каждого слова поставим пару векторов

$$w_i \rightarrow (v_i, u_i)$$

$$\begin{matrix} v_i \\ u_i \end{matrix} \in \mathbb{R}^d$$

$v_i$  - вектор, когда  $w_i$  является центральным  
 $u_i$  - вектор, когда  $w_i$  стоит в контексте

Будем оптимизировать ф-цию, чтобы вер-ть слов, стоящих в одном окне ( $w_1 w_4, w_3 w_4, w_4 w_5, w_4 w_6$ )보다 подобные, вер-ть слов, не стоящих в одном контексте,보다 поменять.

$$L(v, u) = -\log \left( \prod_{d \in D} \prod_{i=1}^{l(d)} \prod_{\substack{j=-m \\ j \neq i}}^m p(w_{i+j} | w_i) \right) =$$

$$= - \sum_{d \in D} \sum_{i=1}^{l(d)} \sum_{\substack{j=-m \\ j \neq i}}^m \log(p(w_{i+j} | w_i))$$

$$p(w_i | w_j) = \text{softmax} = \frac{\exp(u_i^T v_j)}{\sum_{k=1}^n \exp(u_k^T v_j)}$$

График:

$$v_i = v_i - \eta \nabla_{v_i} L(v, u)$$

$$u_i = u_i - \eta \nabla_{u_i} L(v, u)$$

Word2Vec variants:

- Skip-Gram → предсказывает контекст по слову
- Continuous Bag-of-Words → предсказывает слово по контексту

## Negative Sampling

В W2V выражение  $P(w_i | w_j)$  очень медленное:

$$P(w_i | w_j) = \frac{\exp(u_i^T v_j)}{\sum_{k=1}^m \exp(u_k^T v_j)}$$

проходится по всему словарю + вспомога.  $\exp$

Идея Neg. S.:

сэмплируем  $k$  ( $\approx 10$ ) слов из словаря и считаем однобук. только по этим

Функционал:

$$\prod_{i,j \in \text{Positive}} P(D=1 | w_i, w_j) \prod_{i,j \in \text{Neg.}} P(D=0 | w_i, w_j) \Rightarrow$$

$$L(v, u) = - \sum_{i,j \in \text{Posit.}} \log \sigma(u_i^T v_j) - \sum_{i,j \in \text{Neg.}} \log(1 - \sigma(u_i^T v_j))$$

\*  $P(D=1 | w_i, w_j)$  — вер-тб., что пара имеет родину

## 2) GloVe (Global Vectors)

$\phi$ -функция потерь:

$$\sum_{w,c \in V} f(N(w,c)) \cdot (u_c^T v_w + b_c + b_w - \log N(w,c))^2$$

↓  
 φ-функция, которая  
 измеряет реальную  
 пару слов

← Векторы  
 как в W2V

сумма ген  
 слова и контекста

$N(w,c)$  — кол-во раз  $w$   
 встречается в  
 контексте  $c$

$$f(x) = \begin{cases} \left(\frac{x}{x_{\max}}\right)^{\alpha}, & x < x_{\max} \\ 1, & x \geq x_{\max} \end{cases}$$

