

# Векторные представления СЛОВ

## Лекция 5

Майоров Владимир Дмитриевич

8 октября 2021

# Векторное представление слов

- Атомарные единицы текста – слова
- Word embedding – вещественный вектор в пространстве с фиксированной размерностью
  - Пусть есть словарь всех слов языка  $V = \{v_i\}$  размером  $n = |V|$
  - Пусть задана фиксированная размерность  $d$
  - Каждому слову  $v_i \in V$  ставится в соответствие вектор  $w_i \in \mathbb{R}^d$

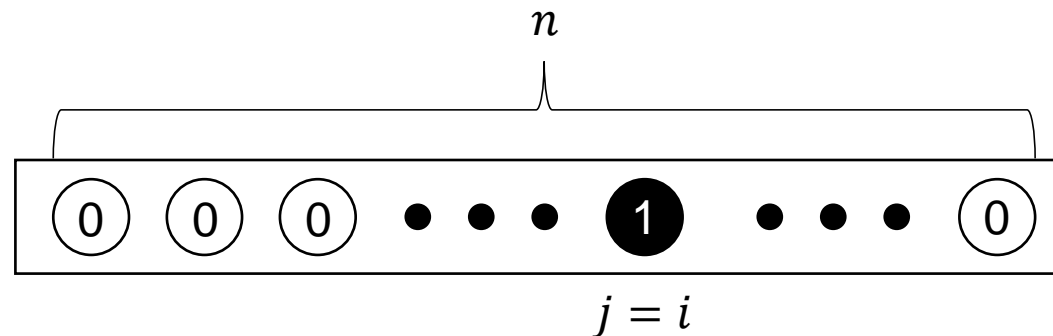
# Векторное представление слов

- Атомарные единицы текста – слова
- Word embedding – вещественный вектор в пространстве с фиксированной размерностью
  - Пусть есть словарь всех слов языка  $V = \{v_i\}$  размером  $n = |V|$
  - Пусть задана фиксированная размерность  $d$
  - Каждому слову  $v_i \in V$  ставится в соответствие вектор  $w_i \in \mathbb{R}^d$
- Пример: one-hot encoding

# One-hot encoding

- Пусть есть словарь всех слов языка  $V = \{v_i\}$  размером  $n = |V|$
- Пусть задана фиксированная размерность  $d = n$ 
  - Каждому слову  $v_i \in V$  ставится в соответствие вектор  $w_i \in \mathbb{R}^d$ ,

$$w_{ij} = \begin{cases} 1, & j = i \\ 0, & j \neq i, j = \overline{1, d} \end{cases}$$



# Многозначность (Синонимия)

- Для большинства задач NLP важен смысл слова (**лексическое значение**), а не само слово:

16 августа 1820 года Пушкин **прибыл** в Феодосию  
**приехал**  
**пожаловал**  
**...**

- Похожесть слов (косинусная мера)

$$\text{similarity}(w_i, w_j) = \frac{(w_i, w_j)}{\|w_i\| \cdot \|w_j\|} = \frac{\sum_{k=1}^n w_{ik} \cdot w_{jk}}{\sqrt{\sum_{k=1}^n w_{ik}^2} \cdot \sqrt{\sum_{k=1}^n w_{jk}^2}}$$

# Многозначность (Синонимия)

- One-hot encoding

- $V = \{v_i\}, |V| = n$
- $d = n$

- Каждому слову  $v_i \in V$  ставится в соответствие вектор из  $w_i \in \mathbb{R}^d$ ,  $w_{ij} = \begin{cases} 1, j = i \\ 0, j \neq i \end{cases}, j = \overline{1, n}$

- все слова одинаково непохожи:

$$(w_i, w_j) = 0, \quad i \neq j$$

$$\text{similarity}(w_i, w_j) = 0, \quad i \neq j$$

# Embedding слой в нейронной сети

- Пусть задан one-hot encoding:

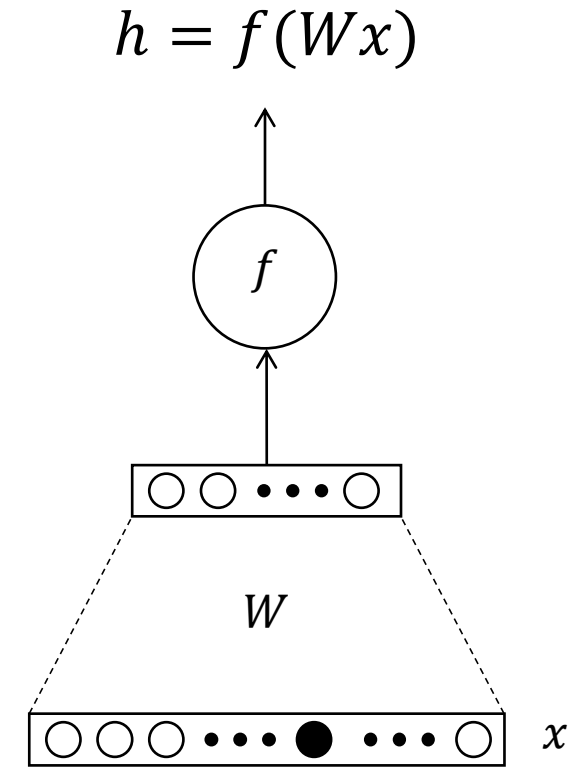
- Словарь  $V = \{v_i\}$ ,  $|V| = n$
- Каждому слову  $v_i \in V$  поставлен в соответствие вектор  $x_i \in \mathbb{R}^n$

- Пусть задана некоторая нейронная сеть

$$h = f(Wx), x \in \mathbb{R}^n, W \in \mathbb{R}^{d \times n}$$

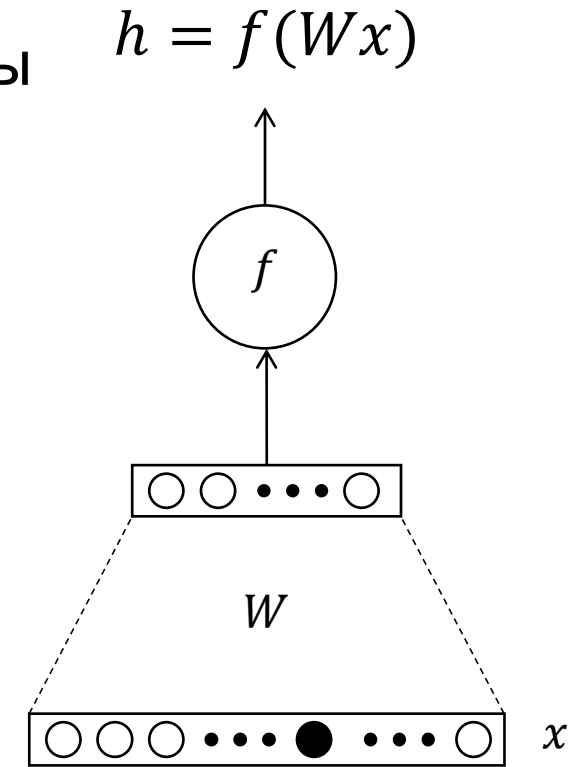
- Преобразование перед первой активацией ( $w = Wx$ ):

- Словарь  $V = \{v_i\}$ ,  $|V| = n$
- Фиксированная размерность  $d$
- Каждому слову  $v_i \in V$  поставлен в соответствие вектор  $w = Wx \in \mathbb{R}^d$



# Embedding слой в нейронной сети

- В процессе обучения с учителем оптимизируются параметры сети (в том числе матрица  $W$ )
- В результате вектор  $w = Wx$  отражает «хорошие» векторы слов с точки зрения целевой задачи
- Основная проблема:
  - Для большинства задач обработки текстов обучающих данных мало  $\Rightarrow$  построить «хорошую» матрицу  $W$  не удастся
- Решение:
  - Инициализировать матрицу  $W$  посчитанными заранее «хорошими» векторами





# Векторное представление слов

Задача обучения без учителя (unsupervised learning):

По коллекции объектов (обучающей выборке) определить внутренние взаимосвязи, зависимости, существующие между объектами

По коллекции неразмеченных текстов построить векторные представления слов из этих текстов

причем хочется, чтобы *similarity* близких по значению слов была выше, чем для различных по значению

# Дистрибутивная гипотеза

- Firth, J. R. (1957):  
“You shall know a word by the company it keeps”

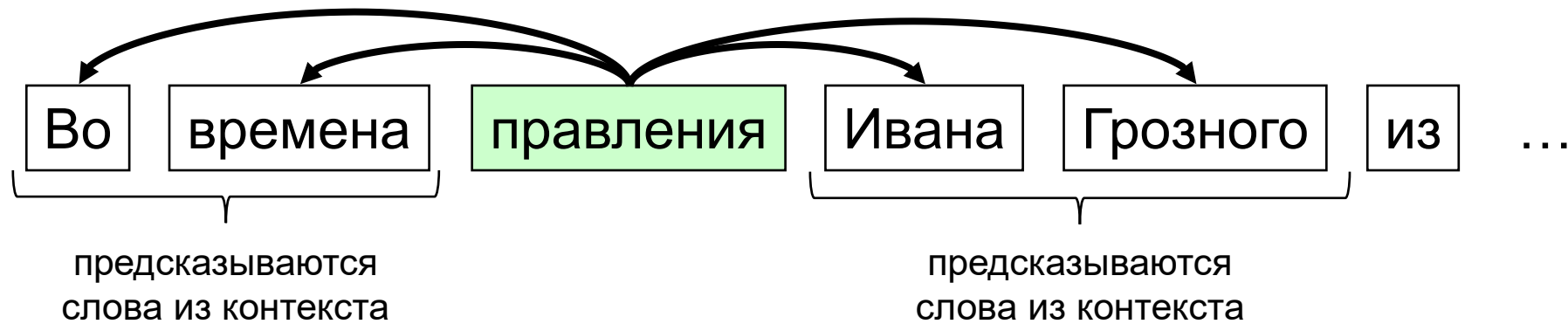
- Бутылка ***tesgüino*** стоит на столе
- ***Tesgüino*** делает тебя пьяным
- Мы делаем ***tesgüino*** из кукурузы
- Вместо хмеля в ***tesgüino*** используется местная трава

# Дистрибутивная гипотеза

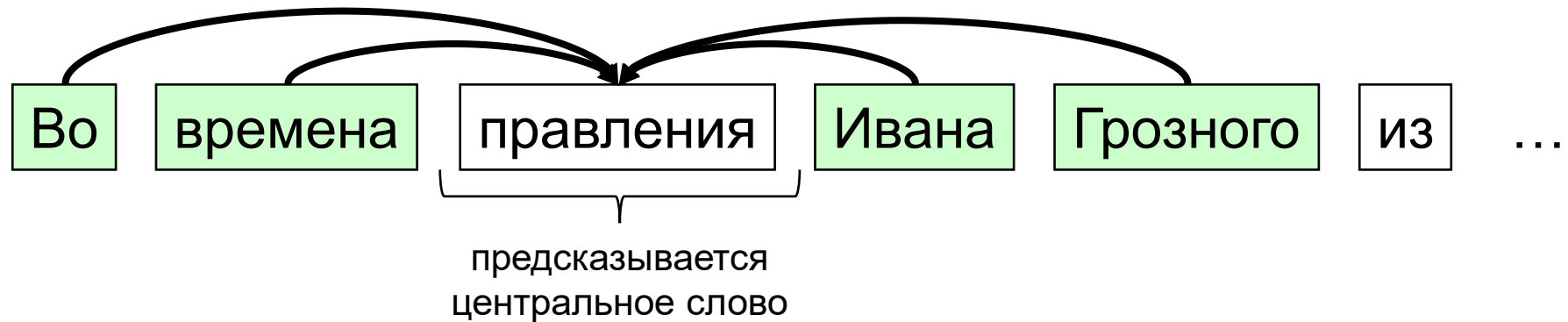
- Слова, которые встречаются в схожих контекстах, имеют схожий СМЫСЛ
- Контекстом слова может являться:
  - Соседние слова
    - Слева
    - Справа
    - Симметрично
  - Весь текст (параграф, предложение)

# word2vec

- Continuous skip-gram

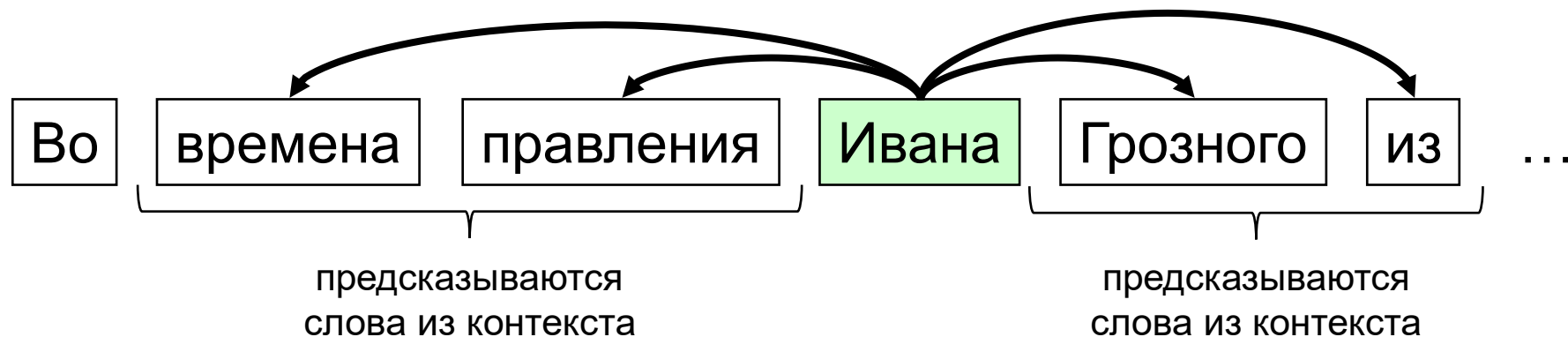


- Continuous bag of words

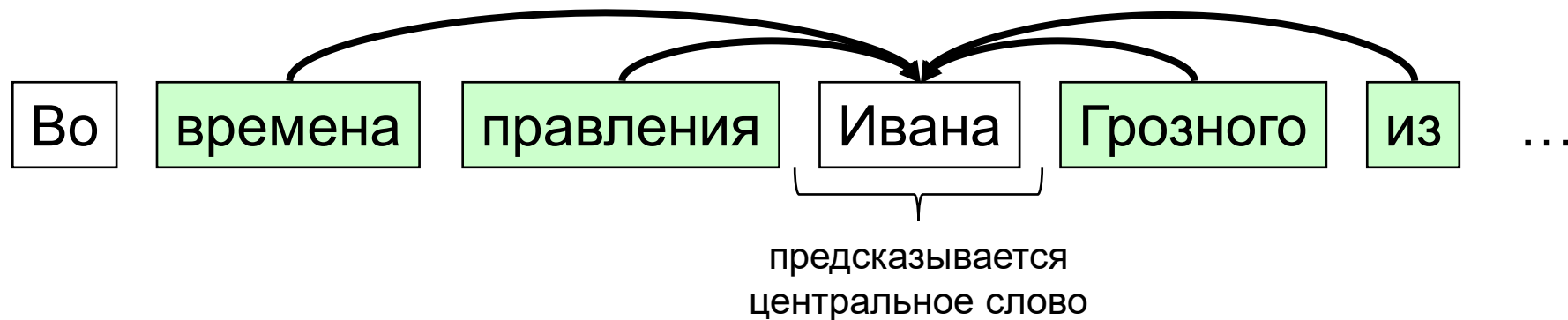


# word2vec

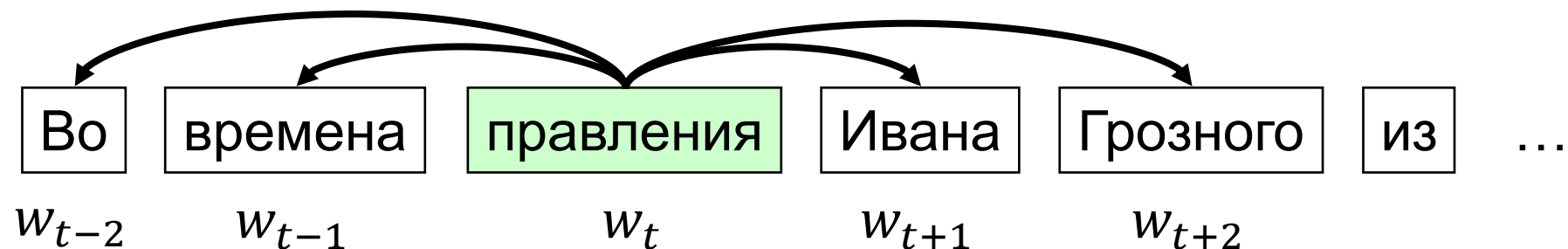
- Continuous skip-gram



- Continuous bag of words



# word2vec skip-gram

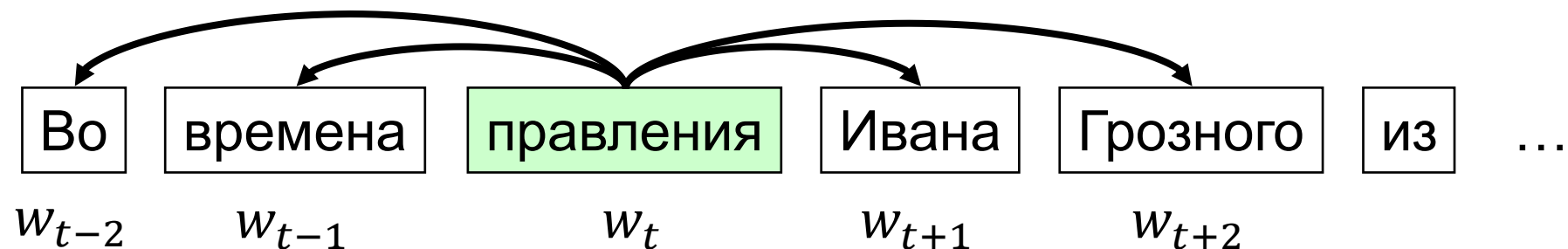


- Цель: максимизировать вероятность всех контекстных слов при данном центральном слове

$$J'(\theta) = \prod_{t=1}^T \prod_{-m \leq j \leq m, j \neq 0} p(w_{t+j} | w_t, \theta)$$

- $\theta$  – оптимизируемые параметры

# word2vec skip-gram



- Цель: максимизировать логарифм вероятности всех контекстных слов при данном центральном слове

$$J(\theta) = -\log J'(\theta) = -\sum_{t=1}^T \sum_{-m \leq j \leq m, j \neq 0} \log p_{\theta}(w_{t+j}|w_t)$$

- $\theta$  – оптимизируемые параметры

# word2vec skip-gram

$$J(\theta) = - \sum_{t=1}^T \sum_{-m \leq j \leq m, j \neq 0} \log p_{\theta}(w_{t+j} | w_t)$$

- $\theta = \{V, U\}$ ;
- $V$  – векторы центрального слова
- $U$  – векторы слова из контекста

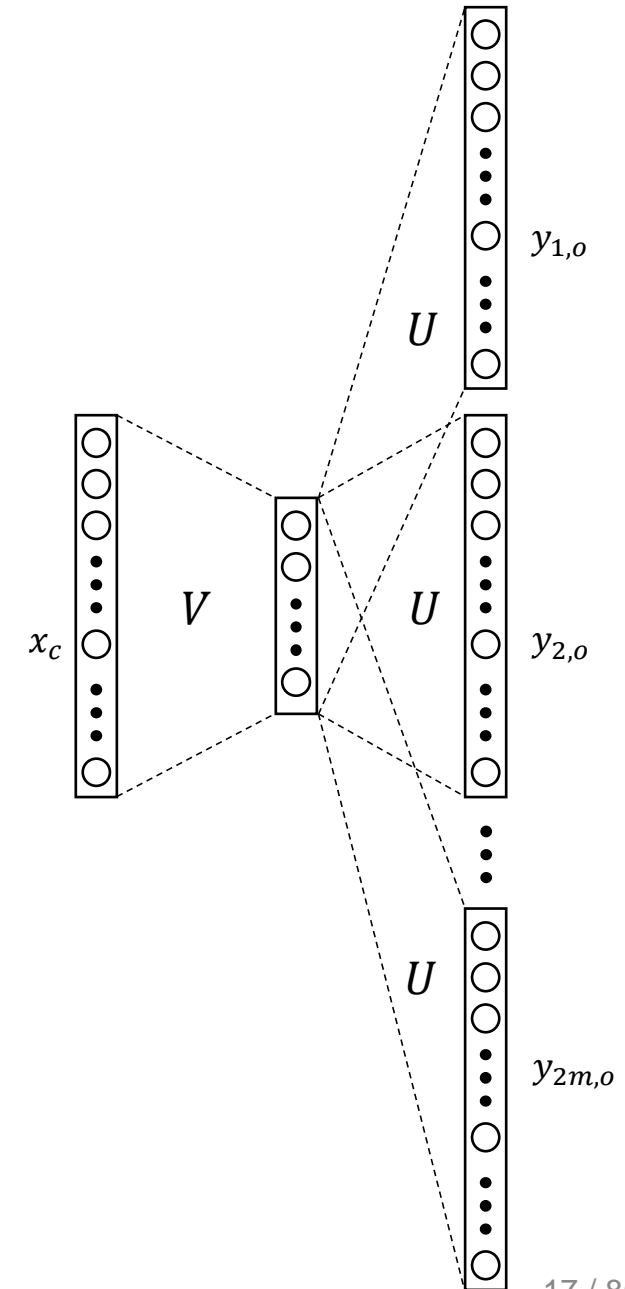
- $p(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w=1}^n \exp(u_w^T v_c)}$ ;



# word2vec skip-gram

Модель может быть представлена в виде нейронной сети:

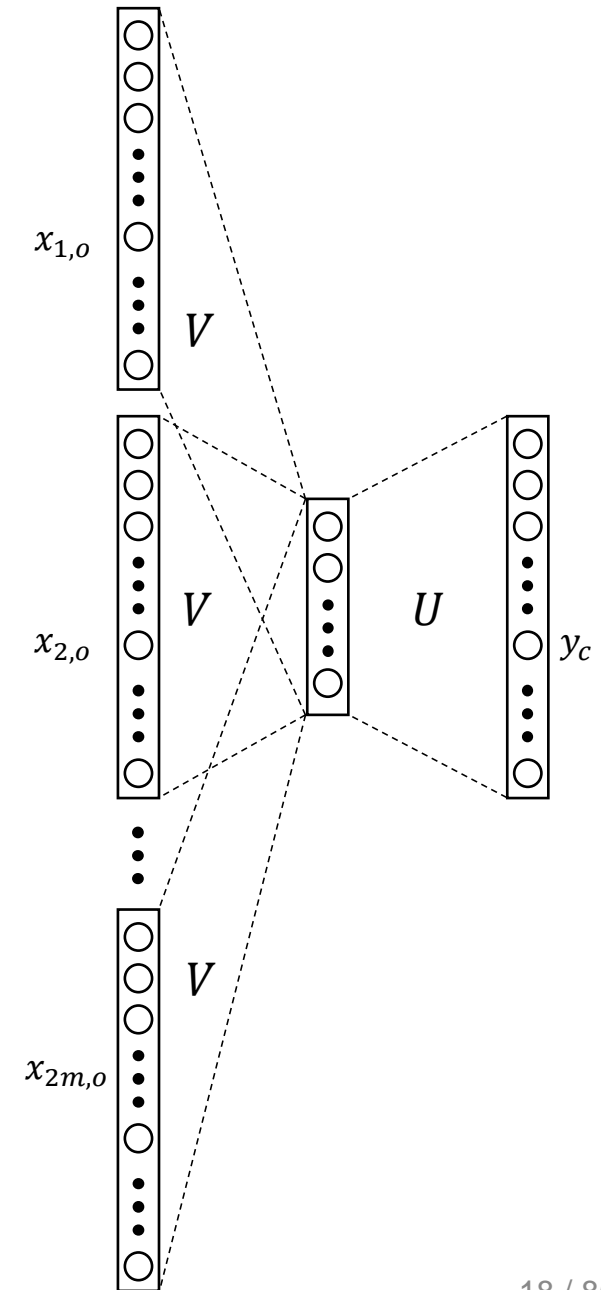
- **Вход:** one-hot центрального слова
- **Скрытый слой:** линейный
- **Выходной слой:** softmax
- **Функция ошибки:** cross-entropy



# word2vec CBOW

Модель может быть представлена в виде нейронной сети:

- **Вход:** one-hot контекстных слов
- **Скрытый слой:** линейный
- **Выходной слой:** softmax
- **Функция ошибки:** cross-entropy



# word2vec skip-gram

$$J(\theta) = - \sum_{t=1}^T \sum_{-m \leq j \leq m, j \neq 0} \log p_{\theta}(w_{t+j} | w_t)$$

- Для каждого окна минимизируем

$$-\log p(o|c) = -\log \frac{\exp(u_o^T v_c)}{\sum_{w=1}^n \exp(u_w^T v_c)};$$

# word2vec skip-gram

$$J(\theta) = - \sum_{t=1}^T \sum_{-m \leq j \leq m, j \neq 0} \log p_{\theta}(w_{t+j} | w_t)$$

- Для каждого окна минимизируем

$$-\log p(o|c) = -\log \frac{\exp(u_o^T v_c)}{\sum_{w=1}^n \exp(u_w^T v_c)};$$

- Метод градиентного спуска

$$\frac{\partial(-\log p(o|c))}{\partial v_c} = -u_o + \sum_{i=1}^n \frac{\exp(u_i^T v_c)}{\sum_{w=1}^n \exp(u_w^T v_c)} u_i == -u_o + \sum_{x \in V} p(x|c) u_x$$

# word2vec skip-gram

$$J(\theta) = - \sum_{t=1}^T \sum_{-m \leq j \leq m, j \neq 0} \log p_{\theta}(w_{t+j} | w_t)$$

- Для каждого окна минимизируем

$$-\log p(o|c) = -\log \frac{\exp(u_o^T v_c)}{\sum_{w=1}^n \exp(u_w^T v_c)};$$

- Метод градиентного спуска

$$\frac{\partial(-\log p(o|c))}{\partial v_c} = -u_o + \sum_{i=1}^n \frac{\exp(u_i^T v_c)}{\sum_{w=1}^n \exp(u_w^T v_c)} u_i == -u_o + \sum_{x \in V} p(x|c) u_x$$

# word2vec skipgram

- Проблема:

- На каждом шаге градиентного спуска вычисляется

$$\sum_{w=1}^n \exp(u_w^T v_c)$$

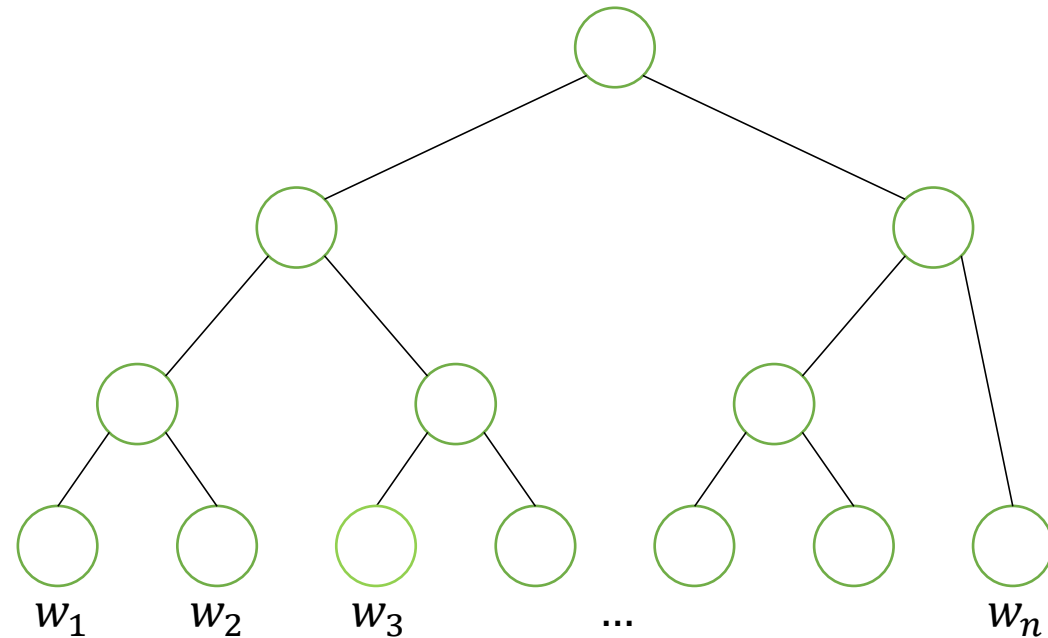
словарь большой  $\Rightarrow$  долго

- Решения:

- Hierarchical softmax
- Negative sampling

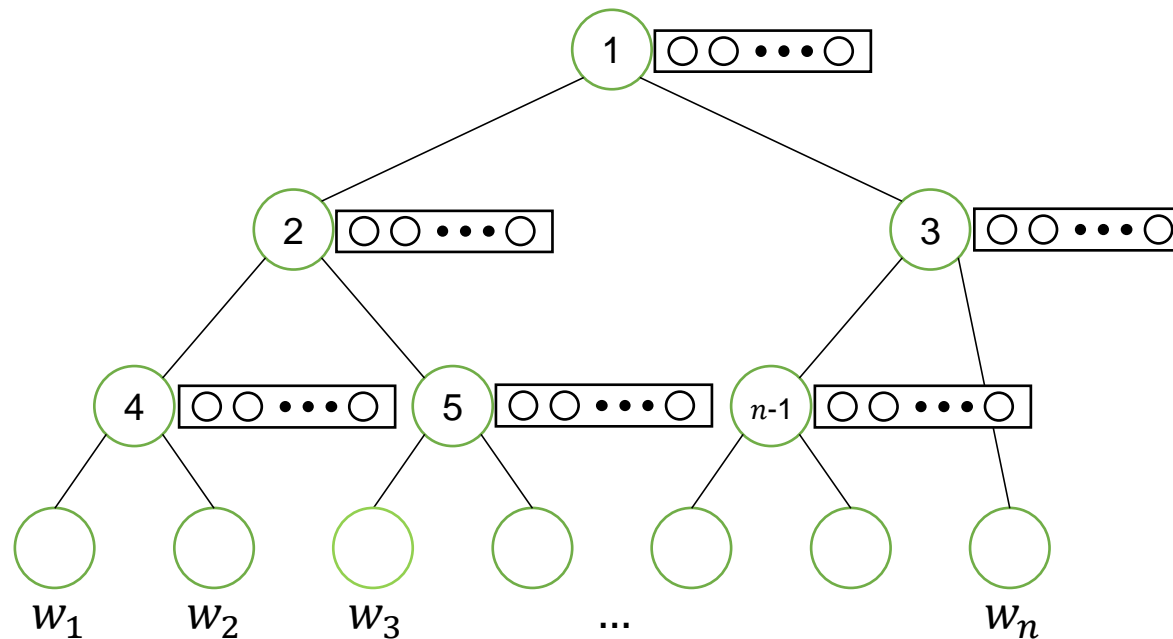
# Hierarchical softmax

- Основная идея:
  - Составить из словаря бинарное дерево



# Hierarchical softmax

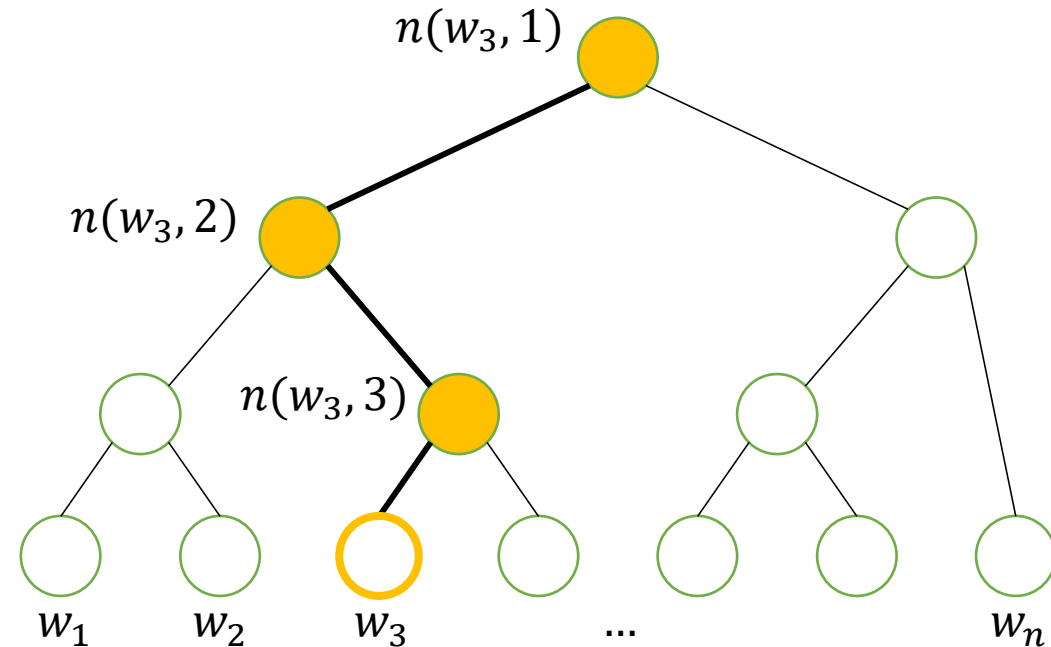
- Основная идея:
  - Составить из словаря бинарное дерево
  - Назначить каждому промежуточному узлу  $i$  вектор  $u_i$





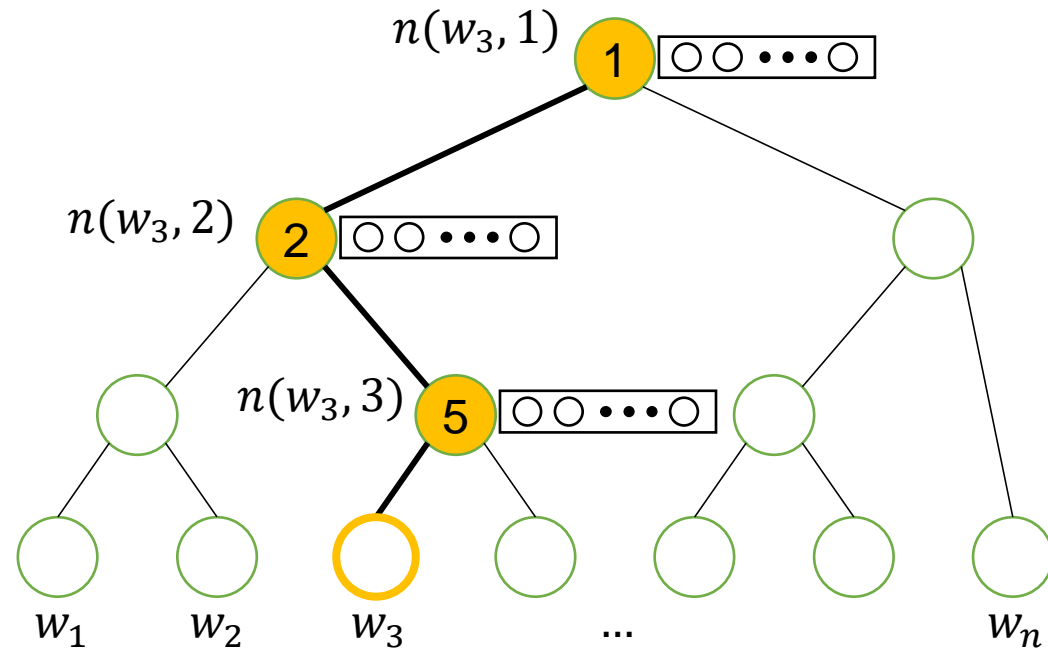
# Hierarchical softmax

- Основная идея:
  - Составить из словаря бинарное дерево
  - Назначить каждому промежуточному узлу  $i$  вектор  $u_i$
  - Предсказывать путь в дереве вместо слова из словаря



$n(w, p)$  – номер  $p$ -того узла в пути от корня к  $w$

# Hierarchical softmax



$$p(o|c) = \prod_{j=1}^{L(o)-1} \sigma(\llbracket n(o, j+1) = ch(n(o, j)) \rrbracket u_{n(o, j)}^T v_c)$$

$ch(n)$  – левый потомок узла  $n$ ,  $\llbracket x \rrbracket = \begin{cases} 1, & \text{если } x \text{ – истина} \\ -1, & \text{если } x \text{ – ложь} \end{cases}$

## Negative sampling

- Сводим задачу к бинарной классификации:

$$z = \begin{cases} 1, & (c, o) \in D \\ 0, & (c, o) \notin D \end{cases};$$

$$p(z = 1|o, c) = \frac{1}{1 + \exp(-v_c^T u_o)} = \sigma(v_c^T u_o)$$

- На каждый положительный пример берем  $K$  отрицательных:
  - Небольшие наборы данных – 5-20 примеров
  - Большие наборы данных – 2-5 примеров

# Negative sampling

- Для каждого окна максимизируем

$$\begin{aligned}
 J_t(\theta) &= \log p(z = 1|o, c) + \sum_{j \sim P(w)} \log p(z = 0|j, c) \\
 &= \log \sigma(v_c^T u_o) + \sum_{j \sim P(w)} \log \sigma(-v_c^T u_j)
 \end{aligned}$$

- $P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=1}^n f(w_j)^{3/4}}$ ;
- $f(w_i)$  – относительная частота слова  $w_i$  в  $D$

# Negative sampling



- Набор данных:

- Положительные примеры: пары слов из окон наших текстов
- Отрицательные примеры: случайные слова из текстов

| o       | c         | z |
|---------|-----------|---|
| времена | правления | 1 |
| из      | правления | 0 |
| слово   | правления | 0 |
| Москвы  | правления | 0 |
|         | ...       |   |

# Проблема частых слов

- Слишком частые слова (предлоги, союзы, пунктуация)
  - часто встречаются в корпусе  $\Rightarrow$  вносят большое влияние на векторы слов
  - встречаются во всевозможных контекстах  $\Rightarrow$  векторы не отражают значение слова

Решение:

- Выкидывать слишком частые слова из корпуса

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}};$$

- $f(w_i)$  – относительная частота слова  $w_i$  в  $D$
- $t$  – порог частоты (обычно около  $10^{-5}$ )

# Оценка качества

- Intrinsic (in vitro)
  - Пытаемся оценить результат решения задачи сравнением с эталонным результатом
  
- Extrinsic (in vivo)
  - Пытаемся оценить результат решения задачи как подзадачи более сложной задачи

# Оценка качества (Intrinsic)

- Задача аналогии слов:

Слово  $a$  относится к слову  $b$  также, как слово  $c$  относится к слову \_\_\_\_.

- $d = \arg \max_i \text{similarity}(x_b - x_a + x_c, x_i)$

- Метрика:

- $\text{Accuracy} = \frac{\text{correct}}{\text{total}};$

## Синтаксические аналогии

| <b>a</b> | <b>b</b> | <b>c</b> | _____           |
|----------|----------|----------|-----------------|
| лекция   | лекции   | семинар  | <b>семинары</b> |
| бежать   | бегущий  | лежать   | <b>лежащий</b>  |



# Оценка качества (Intrinsic)

- Задача аналогии слов:

Слово  $a$  относится к слову  $b$  также, как слово  $c$  относится к слову \_\_\_\_.

- $d = \arg \max_i \text{similarity}(x_b - x_a + x_c, x_i)$

- Метрика:

- $Accuracy = \frac{correct}{total}$ ;

## Семантические аналогии

| <b>a</b> | <b>b</b> | <b>c</b> | _____          |
|----------|----------|----------|----------------|
| лететь   | плыть    | самолет  | <i>корабль</i> |
| Россия   | Москва   | Франция  | <i>Париж</i>   |

## Оценка качества (Intrinsic)

- Задача аналогии слов
  - Найти слово по аналогии с другими
- Задача похожести пар слов
  - Отсортировать пары слов в соответствии со смысловой близостью
- Задача поиска синонимов
  - Для слова найти синонимы среди заданного множества слов
- Задача поиска лишнего слова
  - В множестве слов найти лишнее

# Матрица совместной встречаемости

мама мыла раму.  
раму мыла мама.  
мыла мылом раму.

word-word

|       | мама | мыла | раму | мылом | . |
|-------|------|------|------|-------|---|
| мама  | 0    | 2    | 0    | 0     | 1 |
| мыла  | 2    | 0    | 2    | 1     | 0 |
| раму  | 0    | 2    | 0    | 1     | 2 |
| мылом | 0    | 1    | 1    | 0     | 0 |
| .     | 1    | 0    | 2    | 0     | 0 |

# Матрица совместной встречаемости

- Понижение размерности (SVD)
  - $A = U\Sigma V^T$ ;
    - $\Sigma$  – матрица сингулярных значений ( $m \times m$ )
  - $\hat{A} = \hat{U}\hat{\Sigma}\hat{V}^T$ ; (теорема Эккарта-Янга)
    - $\hat{U}$  – первые  $k$  столбцов матрицы  $U$
    - $\hat{\Sigma}$  –  $k$  первых столбцов и строк матрицы  $\Sigma$
    - $\hat{V}$  – первые  $k$  столбцов матрицы  $V$

# Матрица совместной встречаемости

- Понижение размерности (SVD)

- $A = U\Sigma V^T$ ;

- $\Sigma$  – матрица сингулярных значений(m\*m)

$$A = \begin{bmatrix} 0 & 2 & 0 & 0 & 1 \\ 2 & 0 & 2 & 1 & 0 \\ 0 & 2 & 0 & 1 & 2 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 2 & 0 & 0 \end{bmatrix}$$

$U$

$\Sigma$

$V^T$

$$\begin{bmatrix} -0.385 & -0.435 & -0.359 & -0.557 & -0.472 \\ -0.557 & 0.557 & 0.411 & -0.435 & 0.142 \\ -0.557 & -0.557 & 0.411 & 0.435 & 0.142 \\ -0.286 & 0 & -0.636 & 0 & 0.716 \\ -0.385 & 0.435 & -0.359 & 0.557 & -0.472 \end{bmatrix} * \text{diag} \begin{pmatrix} 3.895 \\ 3.562 \\ 1.292 \\ 0.562 \\ 0.397 \end{pmatrix} * \begin{bmatrix} -0.385 & 0.435 & 0.359 & -0.557 & -0.472 \\ -0.557 & -0.557 & -0.411 & -0.435 & 0.142 \\ -0.557 & 0.557 & -0.411 & 0.435 & 0.142 \\ -0.286 & 0 & 0.636 & 0 & 0.716 \\ -0.385 & -0.435 & 0.359 & 0.557 & -0.472 \end{bmatrix}^T$$

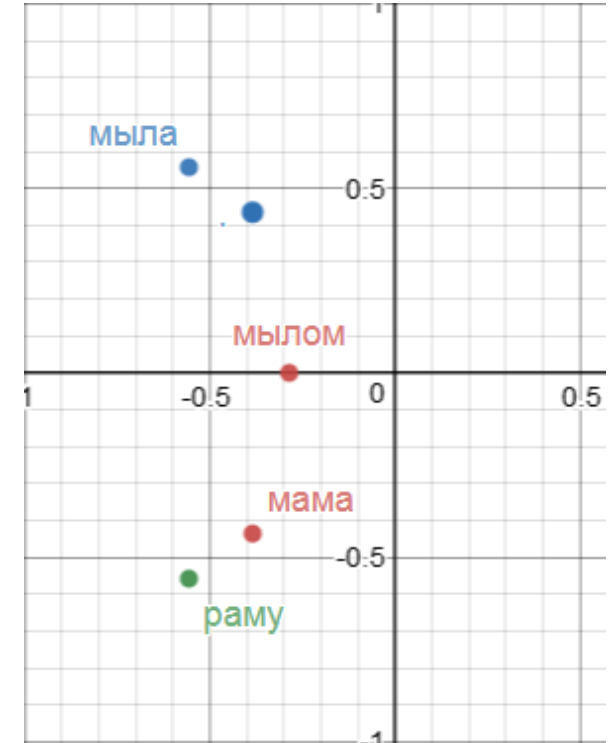
# Матрица совместной встречаемости

- Понижение размерности (SVD)

- $A\hat{A} = \hat{U}\hat{\Sigma}\hat{V}^T$ ;
  - $\hat{U}$  – первые  $k$  столбцов матрицы  $U$
  - $\hat{\Sigma}$  –  $k$  первых столбцов и строк матрицы  $\Sigma$
  - $\hat{V}$  – первые  $k$  столбцов матрицы  $V$

$$\hat{A} = \begin{bmatrix} -0.10 & 1.70 & -0.02 & 0.43 & 1.25 \\ 1.70 & 0.10 & 2.32 & 0.62 & -0.03 \\ -0.03 & 2.32 & 0.10 & 0.62 & 1.70 \\ 0.43 & 0.62 & 0.62 & 0.32 & 0.43 \\ 1.25 & -0.03 & 1.70 & 0.43 & -0.10 \end{bmatrix}$$

$$\hat{U} \hat{\Sigma} \hat{V}^T = \begin{bmatrix} -0.385 & -0.435 & -0.359 & -0.557 & -0.472 \\ -0.557 & 0.557 & 0.411 & -0.435 & 0.142 \\ -0.557 & -0.557 & 0.411 & 0.435 & 0.142 \\ -0.286 & 0 & -0.636 & 0 & 0.716 \\ -0.385 & 0.435 & -0.359 & 0.557 & -0.472 \end{bmatrix} * \text{diag} \begin{pmatrix} 3.895 \\ 3.562 \\ 1.292 \\ 0.562 \\ 0.397 \end{pmatrix} * \begin{bmatrix} -0.385 & 0.435 & 0.359 & -0.557 & -0.472 \\ -0.557 & -0.557 & -0.411 & -0.435 & 0.142 \\ -0.557 & 0.557 & -0.411 & 0.435 & 0.142 \\ -0.286 & 0 & 0.636 & 0 & 0.716 \\ -0.385 & -0.435 & 0.359 & 0.557 & -0.472 \end{bmatrix}^T$$



# GloVe

Обозначения:

- $X$  – матрица совместной встречаемости
- $X_{ij}$  – сколько раз слово  $j$  встретилось в контексте слова  $i$
- $X_i = \sum_k X_{ik}$  – сколько раз любое слово встретилось в контексте  $i$
- $P_{ij} = P(j|i) = X_{ij}/X_i$  – вероятность слова  $j$  встретиться в контексте  $i$

# GloVe

## Основная идея:

Для понимания, насколько близки слова, не достаточно частоты совместной встречаемости – нужно сравнивать частоты встречаемости слова в разных контекстах

| Probability and Ratio | $k = solid$          | $k = gas$            | $k = water$          | $k = fashion$        |
|-----------------------|----------------------|----------------------|----------------------|----------------------|
| $P(k ice)$            | $1.9 \times 10^{-4}$ | $6.6 \times 10^{-5}$ | $3.0 \times 10^{-3}$ | $1.7 \times 10^{-5}$ |
| $P(k steam)$          | $2.2 \times 10^{-5}$ | $7.8 \times 10^{-4}$ | $2.2 \times 10^{-3}$ | $1.8 \times 10^{-5}$ |
| $P(k ice)/P(k steam)$ | 8.9                  | $8.5 \times 10^{-2}$ | 1.36                 | 0.96                 |

- *solid* больше относится к *ice*, чем к *steam*
- *gas* больше относится к *steam*, чем к *ice*
- *water* и *fashion* одинаково хорошо (плохо) относятся к *ice* и *steam*



# GloVe

## Основная идея:

Для понимания, насколько близки слова, не достаточно частоты совместной встречаемости – нужно сравнивать частоты встречаемости слова в разных контекстах

| Probability and Ratio | $k = solid$          | $k = gas$            | $k = water$          | $k = fashion$        |
|-----------------------|----------------------|----------------------|----------------------|----------------------|
| $P(k ice)$            | $1.9 \times 10^{-4}$ | $6.6 \times 10^{-5}$ | $3.0 \times 10^{-3}$ | $1.7 \times 10^{-5}$ |
| $P(k steam)$          | $2.2 \times 10^{-5}$ | $7.8 \times 10^{-4}$ | $2.2 \times 10^{-3}$ | $1.8 \times 10^{-5}$ |
| $P(k ice)/P(k steam)$ | 8.9                  | $8.5 \times 10^{-2}$ | 1.36                 | 0.96                 |

$$F(i, j, k) = P(k|i)/P(k|j) = P_{ik}/P_{jk}$$

# GloVe

$$F(i, j, k) = P(k|i)/P(k|j) = P_{ik}/P_{jk}$$

- $V$  – векторы слов
- $U$  – векторы контекстных слов

$$F(v_i, v_j, u_k) = P_{ik}/P_{jk}$$

# GloVe

$$F(i, j, k) = P(k|i)/P(k|j) = P_{ik}/P_{jk}$$

- $V$  – векторы слов
- $U$  – векторы КОНТЕКСТНЫХ СЛОВ

$$F(v_i, v_j, u_k) = P_{ik}/P_{jk}$$

- $(v_{king} - v_{male} + v_{female} = v_{queen})$

$$F(v_i - v_j, u_k) = P_{ik}/P_{jk}$$

$$F\left((v_i - v_j)^T u_k\right) = P_{ik}/P_{jk}$$

# GloVe

$$F(i, j, k) = P(k|i)/P(k|j) = P_{ik}/P_{jk}$$

- $V$  – векторы слов

- $U$  – векторы КОНТЕКСТНЫХ СЛОВ

$$F(v_i, v_j, u_k) = P_{ik}/P_{jk}$$

- $(v_{king} - v_{male} + v_{female} = v_{queen})$

$$F(v_i - v_j, u_k) = P_{ik}/P_{jk}$$

$$F\left((v_i - v_j)^T u_k\right) = P_{ik}/P_{jk}$$

- $F$  – гомоморфизм групп  $(\mathbb{R}, +)$  и  $(\mathbb{R}_{>0}, \times)$

$$F\left((v_i - v_j)^T u_k\right) = F(v_i^T u_k) / F(v_j^T u_k) = P_{ik}/P_{jk};$$

# GloVe

- $F\left((v_i - v_j)^T u_k\right) = F(v_i^T u_k) / F(v_j^T u_k) = P_{ik} / P_{jk}$  ;
- $F(v_i^T u_k) = P_{ik} = \frac{X_{ik}}{X_i}$

# GloVe

- $F\left((v_i - v_j)^T u_k\right) = F(v_i^T u_k) / F(v_j^T u_k) = P_{ik} / P_{jk}$  ;
- $F(v_i^T u_k) = P_{ik} = \frac{X_{ik}}{X_i}$
- $F = \exp$
- $v_i^T u_k = \log P_{ik} = \log X_{ik} - \log X_i$

# GloVe

- $F\left((v_i - v_j)^T u_k\right) = F(v_i^T u_k) / F(v_j^T u_k) = P_{ik} / P_{jk}$  ;
- $F(v_i^T u_k) = P_{ik} = \frac{X_{ik}}{X_i}$
- $F = \exp$
- $v_i^T u_k = \log P_{ik} = \log X_{ik} - \underline{\log X_i}$
- $v_i^T u_k + \underline{b_i} + \underline{\tilde{b}_k} - \log X_{ik} = 0$

# GloVe

- $F\left((v_i - v_j)^T u_k\right) = F(v_i^T u_k) / F(v_j^T u_k) = P_{ik} / P_{jk}$  ;
- $F(v_i^T u_k) = P_{ik} = \frac{X_{ik}}{X_i}$
- $F = \exp$
- $v_i^T u_k = \log P_{ik} = \log X_{ik} - \log X_i$
- $v_i^T u_k + b_i + \tilde{b}_k - \log X_{ik} = 0$
- $J(v_i, u_k) = (v_i^T u_k + b_i + \tilde{b}_k - \log X_{ik})^2$



# GloVe

$$J(v_i, u_k) = (v_i^T u_k + b_i + \tilde{b}_k - \log X_{ik})^2$$

- $\log X_{ik}$  не определен при  $X_{ik} = 0$

# GloVe

$$J(v_i, u_k) = (v_i^T u_k + b_i + \tilde{b}_k - \log X_{ik})^2$$

- $\log X_{ik}$  не определен при  $X_{ik} = 0$

$$J(v_i, u_k) = \underline{f(X_{ik})} (v_i^T u_k + b_i + \tilde{b}_k - \log X_{ik})^2$$

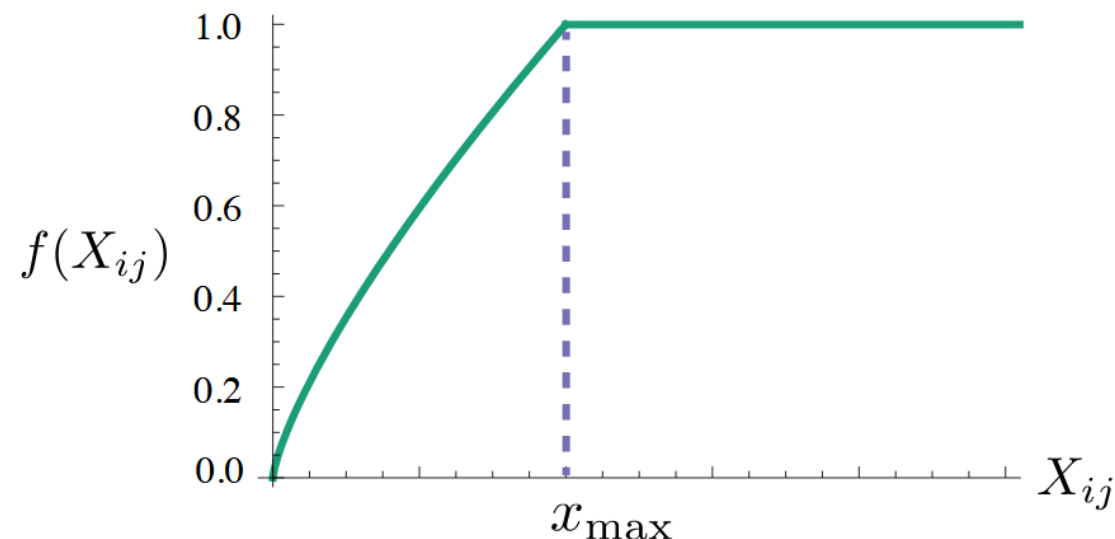
- функция  $f(x)$ :
  - $f(0) = 0$  (более того  $\lim_{x \rightarrow 0} f(x) \log^2 x$  конечен)
  - $f(x)$  – неубывающая
  - $f(x)$  – «не слишком высока» для больших  $x$

# GloVe

Функция потерь

$$J(\theta) = \sum_{i=1}^n \sum_{j=1}^n f(X_{ij}) (v_i^T u_j + b_i + \tilde{b}_j - \log X_{ij})^2,$$

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



## Проблема редких слов

- Для слов, которые встречаются слишком редко, невозможно построить хорошие векторы
- В обучающем корпусе могут отсутствовать редкие слова  $\Rightarrow$  такие слова не попадут в словарь  $V$

## Проблема редких слов

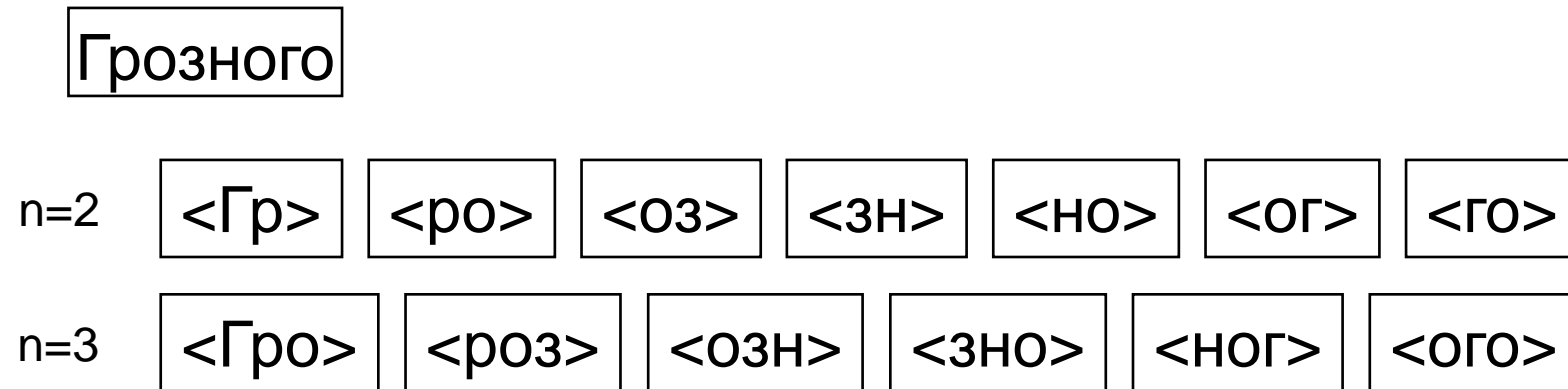
- Редкие слова заменяются на специальную константу OOV (out of vocabulary)
  - В процессе обучения вычисляется вектор для OOV
  - Этот вектор используется для слов, не вошедших в  $V$

## Проблема редких слов

- Редкие слова заменяются на специальную константу OOV (out of vocabulary)
  - В процессе обучения вычисляется вектор для OOV
  - Этот вектор используется для слов, не вошедших в  $V$
- Слова рассматриваются не как атомарные единицы, а как последовательности символов

# fasttext

- Каждое слово  $w$  представим как мультимножество символьных  $n$ -gram



- Составим словарь слов  $V$  и словарь символьных  $n$ -gram  $G$

# fasttext

- В word2vec:

$$p(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{i=1}^n \exp(u_i^T v_c)} = \frac{\exp(s(w_o, w_c))}{\sum_{i=1}^n \exp(s(w_i, w_c))},$$

$s(w_o, w_c) = u_o^T v_c$  - функция оценки (score)



# fasttext

- В word2vec:

$$p(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{i=1}^n \exp(u_i^T v_c)} = \frac{\exp(s(w_o, w_c))}{\sum_{i=1}^n \exp(s(w_i, w_c))},$$

$s(w_o, w_c) = u_o^T v_c$  - функция оценки (score)

- В fasttext:

$$s(w_o, w_c) = u_o^T v_c + \sum_{g \in \Gamma(w_c)} u_o^T z_g,$$

$\Gamma(w_c)$  – n-gram'ы слова  $w_c$

# Сверточные сети (CNN)

- Архитектура нейронной сети, нацеленная на эффективное распознавание образов
- Состоит из:
  - Слой свертки
  - Слой активации
  - Pooling слой

# Сверточные сети (CNN)

- Слой свертки

- На входе матрица (изображение)  $X \in \mathbb{R}^{m \times n}$
- Задана матрица весов (фильтр, ядро свертки)  $K \in \mathbb{R}^{h \times w}$
- Строим выходное изображение, «двигая» фильтр по матрице

$$Y \in \mathbb{R}^{(m-h+1) \times (n-w+1)}$$

$$y_{i,j} = \sum_{q=1}^h \sum_{r=1}^w X_{i+q-1, j+r-1} * K_{q,r}, i = \overline{1, m-h+1}, j = \overline{1, n-w+1}$$

# Сверточные сети (CNN)

- Слой свертки

$$K \in \mathbb{R}^{3 \times 3}$$

|    |    |    |
|----|----|----|
| 2  | -1 | -1 |
| -1 | 0  | 0  |
| -1 | 0  | 2  |

$$Y \in \mathbb{R}^{? \times ?}$$

$$X \in \mathbb{R}^{4 \times 5}$$

|   |   |   |   |   |
|---|---|---|---|---|
| 2 | 1 | 3 | 1 | 2 |
| 3 | 4 | 2 | 0 | 1 |
| 2 | 0 | 1 | 2 | 3 |
| 1 | 3 | 2 | 4 | 0 |

# Сверточные сети (CNN)

- Слой свертки

$$K \in \mathbb{R}^{3 \times 3}$$

|    |    |    |
|----|----|----|
| 2  | -1 | -1 |
| -1 | 0  | 0  |
| -1 | 0  | 2  |

$$Y \in \mathbb{R}^{2 \times 3}$$

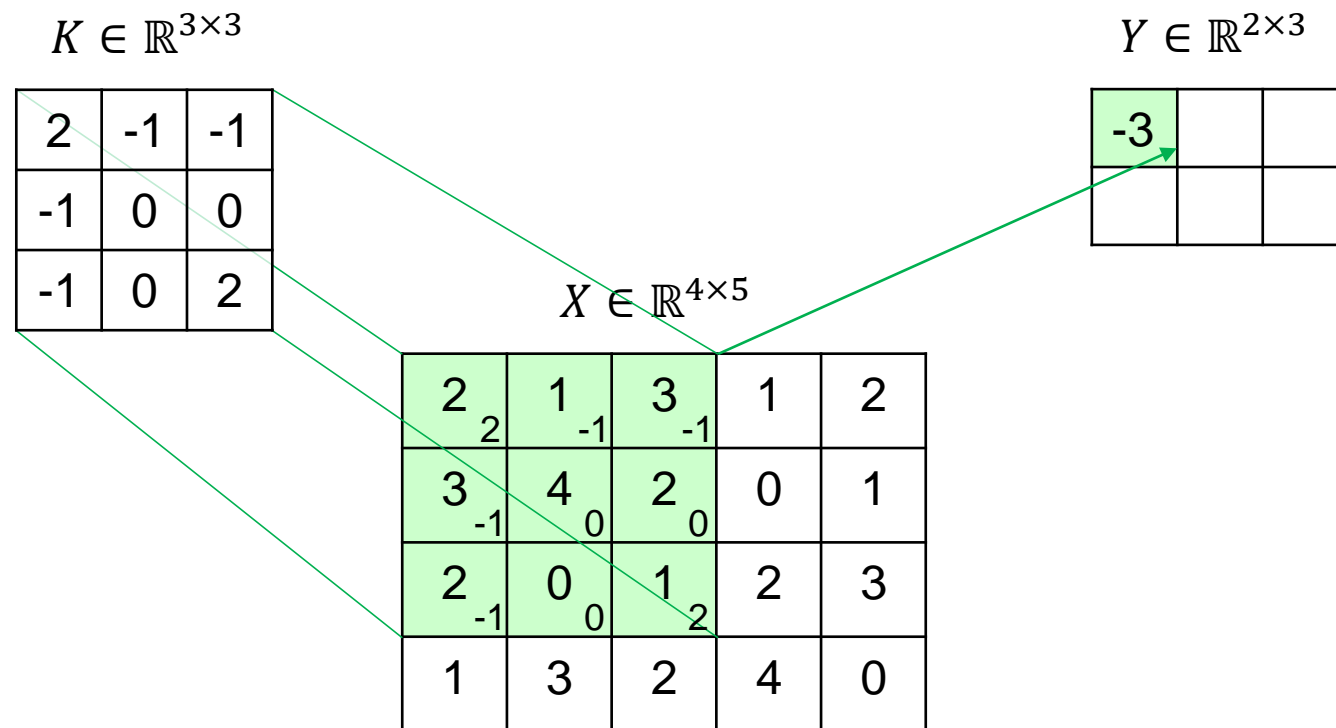
|  |  |  |
|--|--|--|
|  |  |  |
|  |  |  |

$$X \in \mathbb{R}^{4 \times 5}$$

|   |   |   |   |   |
|---|---|---|---|---|
| 2 | 1 | 3 | 1 | 2 |
| 3 | 4 | 2 | 0 | 1 |
| 2 | 0 | 1 | 2 | 3 |
| 1 | 3 | 2 | 4 | 0 |

# Сверточные сети (CNN)

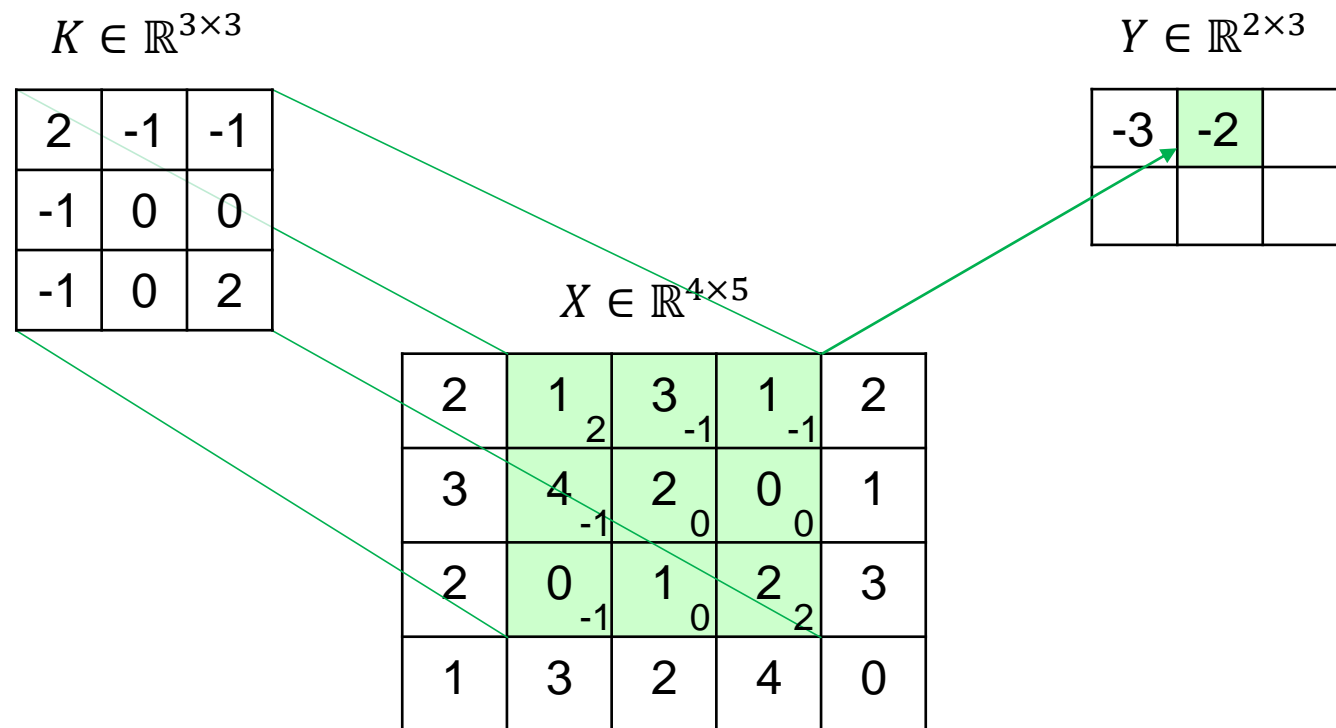
- Слой свертки



$$y_{1,1} = 2 * 2 + (-1) * 1 + (-1) * 3 + (-1) * 3 + 0 * 4 + 0 * 2 + (-1) * 2 + 0 * 0 + 2 * 1 = -3$$

# Сверточные сети (CNN)

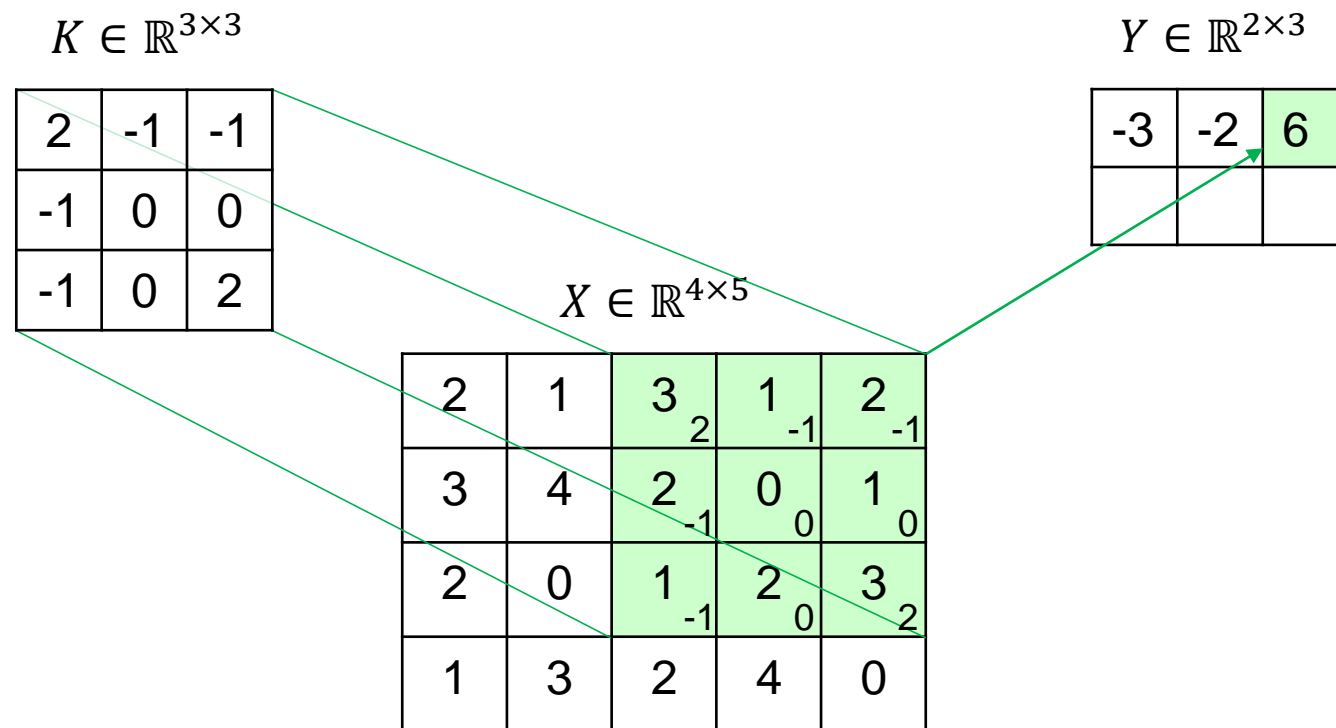
- Слой свертки



$$y_{1,2} = 2 * 1 + (-1) * 3 + (-1) * 1 + (-1) * 4 + 0 * 2 + 0 * 0 + (-1) * 0 + 0 * 1 + 2 * 2 = -2$$

# Сверточные сети (CNN)

- Слой свертки

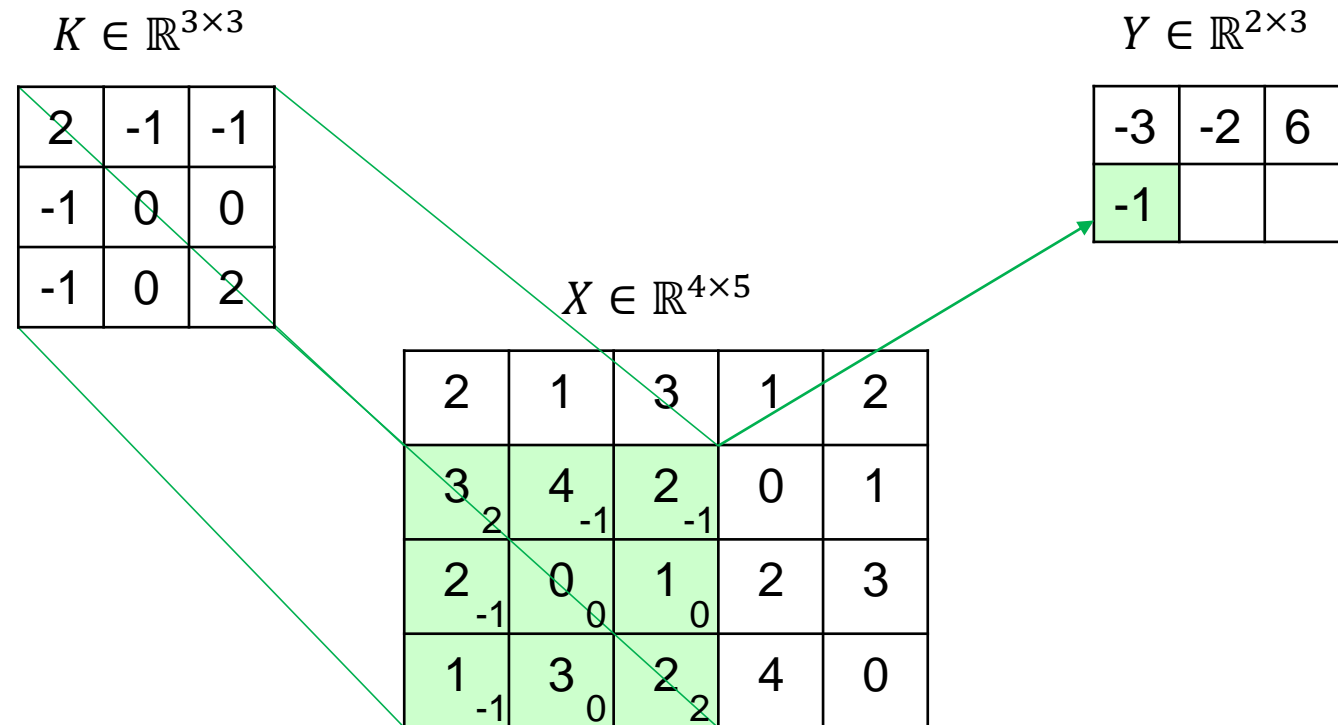


$$y_{1,3} = 2 * 3 + (-1) * 1 + (-1) * 2 + (-1) * 2 + 0 * 0 + 0 * 1 + (-1) * 1 + 0 * 2 + 2 * 3 = 6$$



# Сверточные сети (CNN)

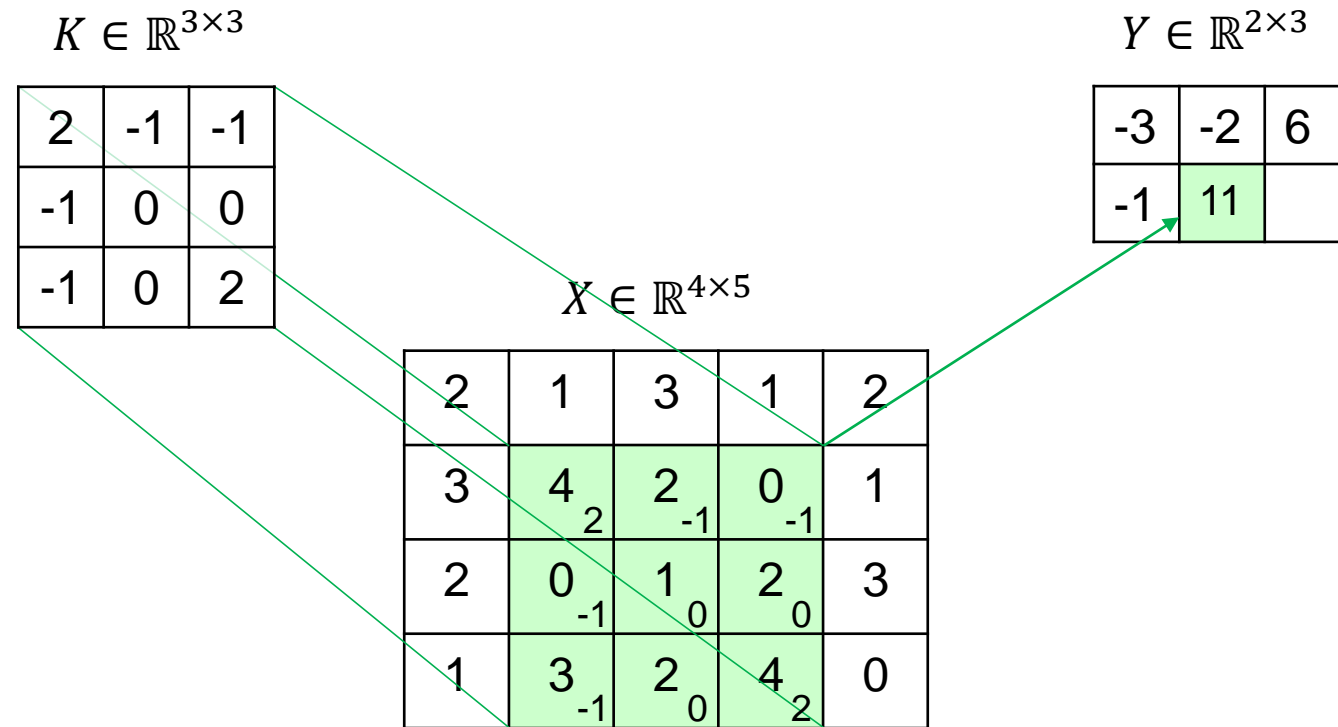
- Слой свертки



$$y_{2,1} = 2 * 3 + (-1) * 4 + (-1) * 2 + (-1) * 2 + 0 * 0 + 0 * 1 + (-1) * 1 + 0 * 3 + 2 * 2 = -1$$

# Сверточные сети (CNN)

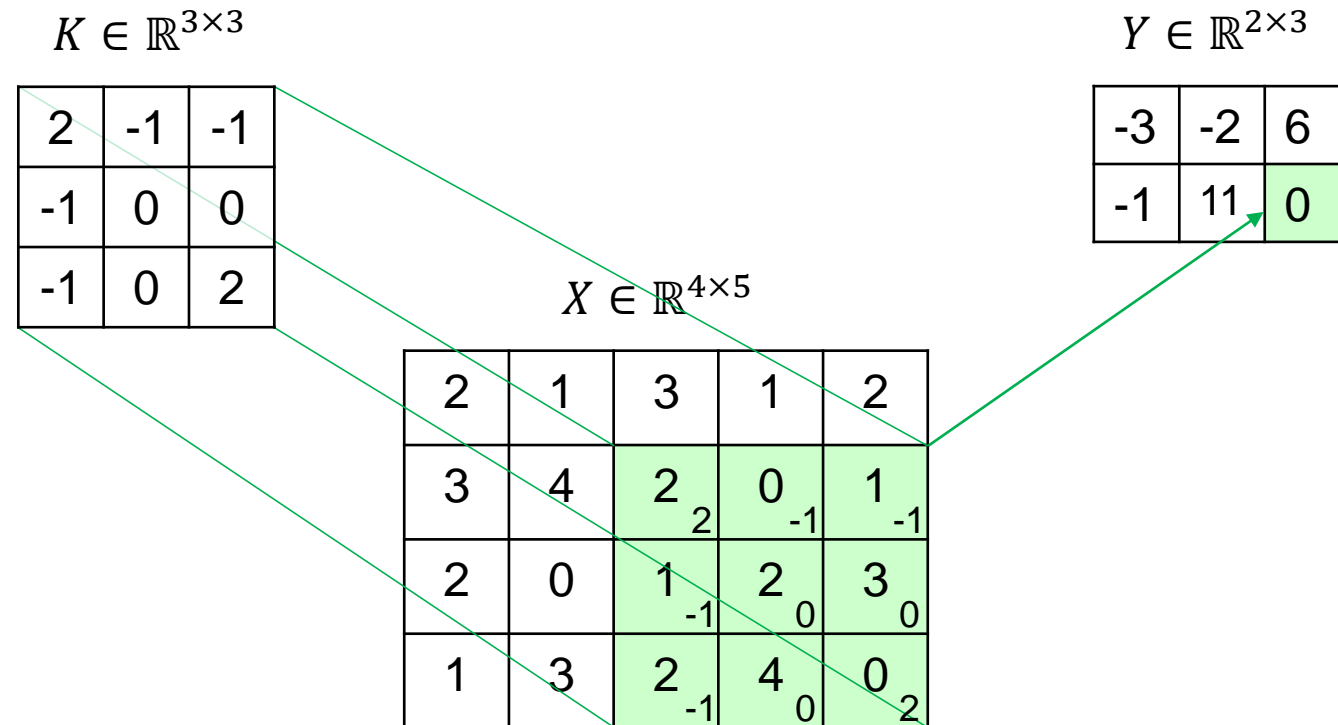
- Слой свертки



$$y_{2,2} = 2 * 4 + (-1) * 2 + (-1) * 0 + (-1) * 0 + 0 * 1 + 0 * 2 + (-1) * 3 + 0 * 2 + 2 * 4 = 11$$

# Сверточные сети (CNN)

- Слой свертки



$$y_{2,3} = 2 * 2 + (-1) * 0 + (-1) * 1 + (-1) * 1 + 0 * 2 + 0 * 3 + (-1) * 2 + 0 * 4 + 2 * 0 = 0$$

# Сверточные сети (CNN)

- Pooling слой

- Агрегирует выходы группы нейронов в один выход
- На входе матрица (признаки)  $X \in \mathbb{R}^{m \times n}$
- Задан размер окна ( $h \times w$ )
- Строим выходные признаки, «двигая» окно по матрице и выполняя операцию агрегации

$$Y \in \mathbb{R}^{\frac{m}{h} \times \frac{n}{w}};$$

$$y_{i,j} = f\left(X_{(i-1)*h:i*h, (j-1)*w:j*w}\right)$$

- В качестве  $f$  обычно используется функция  $\max$

# Сверточные сети (CNN)

- Pooling слой

Окно ( $h \times w$ ):  $h = 2; w = 1$

$$X \in \mathbb{R}^{2 \times 3}$$

|    |    |   |
|----|----|---|
| -3 | -2 | 6 |
| -1 | 11 | 0 |

$$Y \in \mathbb{R}^{? \times ?}$$

# Сверточные сети (CNN)

- Pooling слой

Окно ( $h \times w$ ):  $h = 2; w = 1$

$$X \in \mathbb{R}^{2 \times 3}$$

|    |    |   |
|----|----|---|
| -3 | -2 | 6 |
| -1 | 11 | 0 |

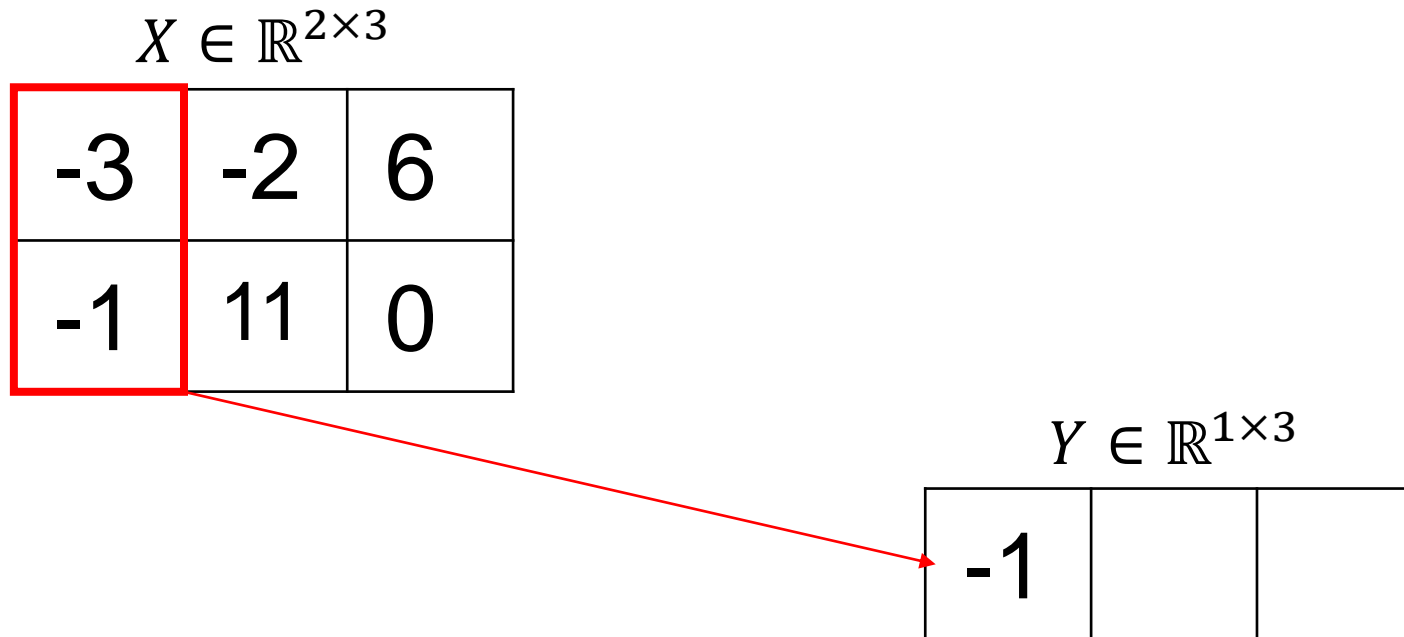
$$Y \in \mathbb{R}^{1 \times 3}$$

|  |  |  |
|--|--|--|
|  |  |  |
|--|--|--|

# Сверточные сети (CNN)

- Pooling слой

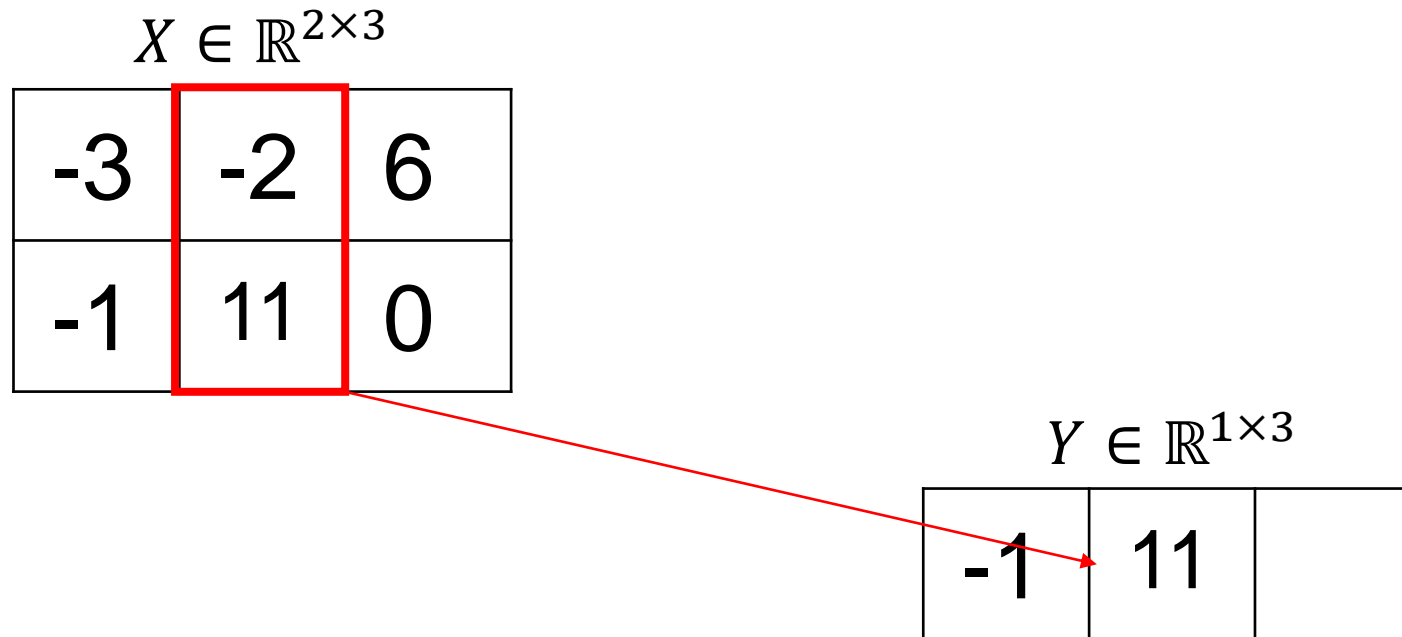
Окно ( $h \times w$ ):  $h = 2; w = 1$



# Сверточные сети (CNN)

- Pooling слой

Окно ( $h \times w$ ):  $h = 2; w = 1$

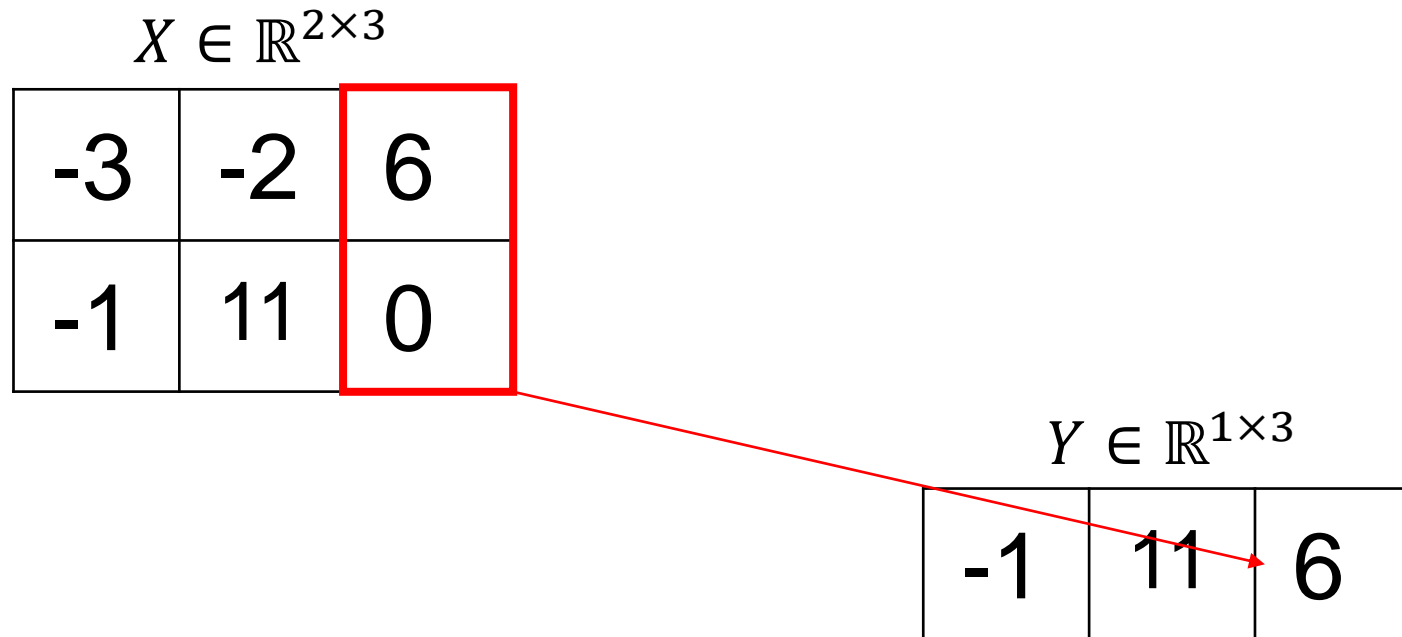




# Сверточные сети (CNN)

- Pooling слой

Окно ( $h \times w$ ):  $h = 2; w = 1$

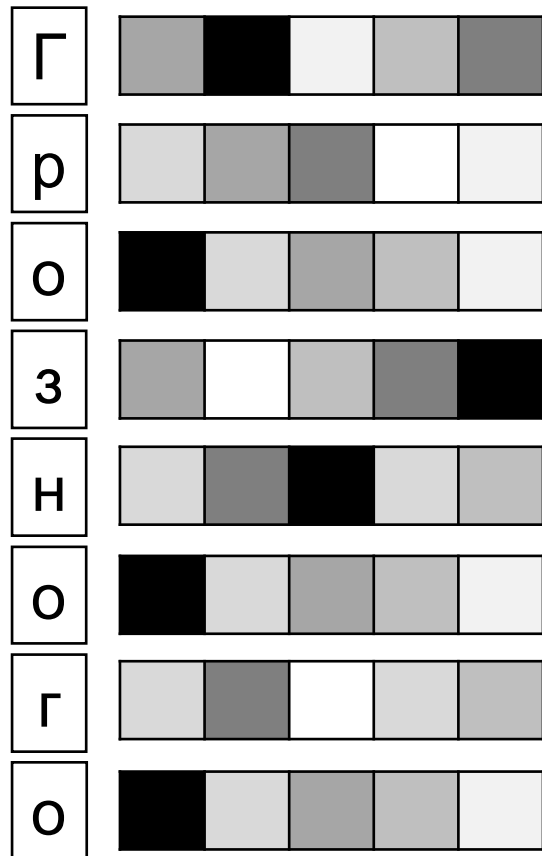


# CharCNN

- Собираем словарь символов  $C$  по обучающему корпусу
- Каждому символу  $c \in C$  ставим в соответствие вектор  $v_c \in \mathbb{R}^d$
- Слово  $w$  рассматриваем как последовательность символов  $[c_1, c_2, \dots, c_n]$
- Подставляя для каждого символа  $c_i$  вектор  $v_{c_i}$  получаем матрицу  $X \in \mathbb{R}^{n \times d}$
- Задаем  $m$  фильтров  $K_i \in \mathbb{R}^{k_i \times d}$
- Применяем к  $X$  свертки с фильтрами  $K_i$  с max pooling и получаем вектор  $y \in \mathbb{R}^m$  для слова  $w$

# CharCNN

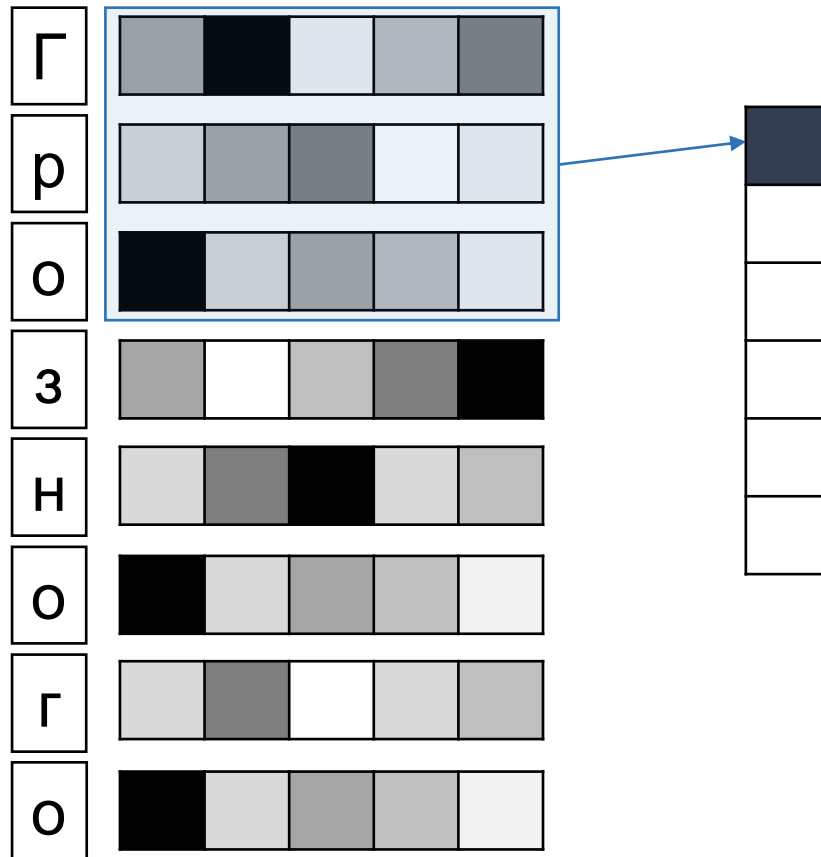
Грозного



# CharCNN

Грозного

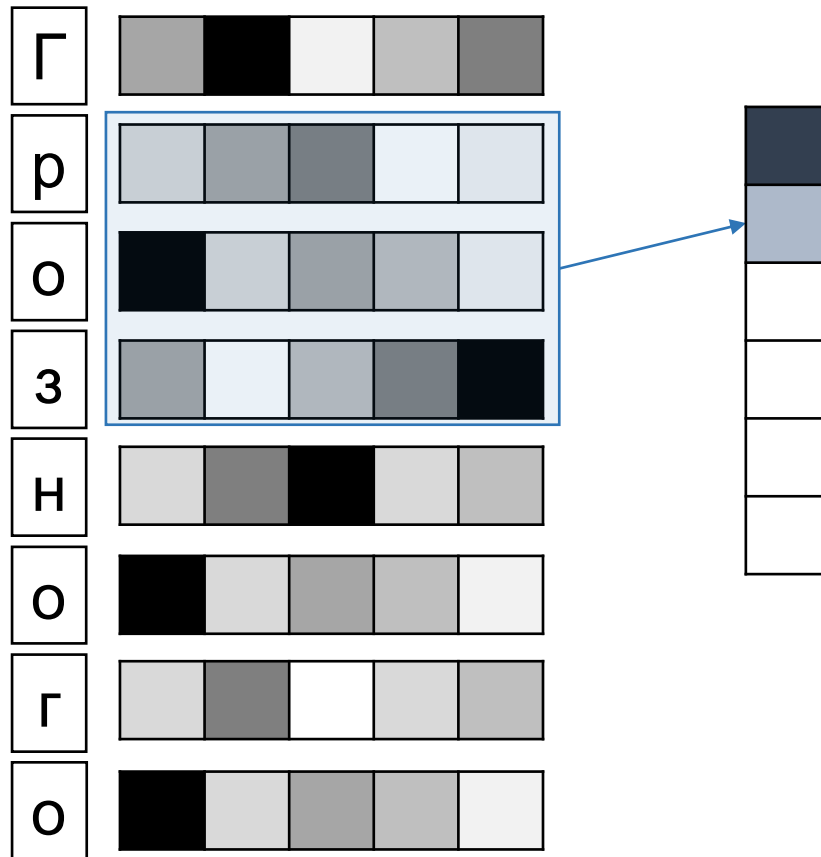
$$K_1 \in \mathbb{R}^{3 \times 5}$$



# CharCNN

Грозного

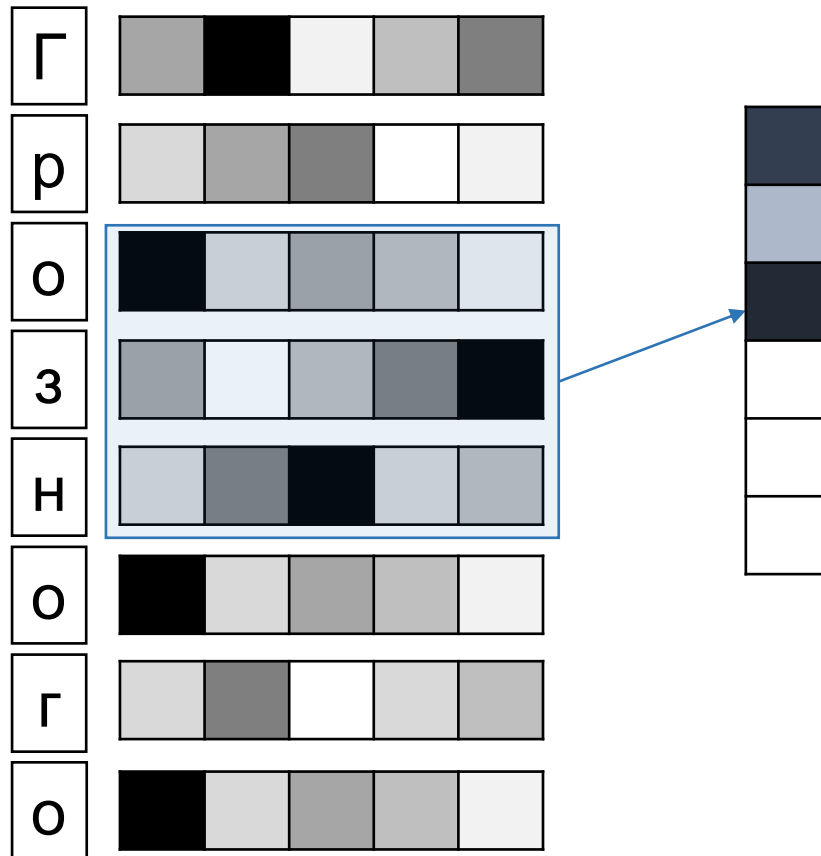
$$K_1 \in \mathbb{R}^{3 \times 5}$$



# CharCNN

Грозного

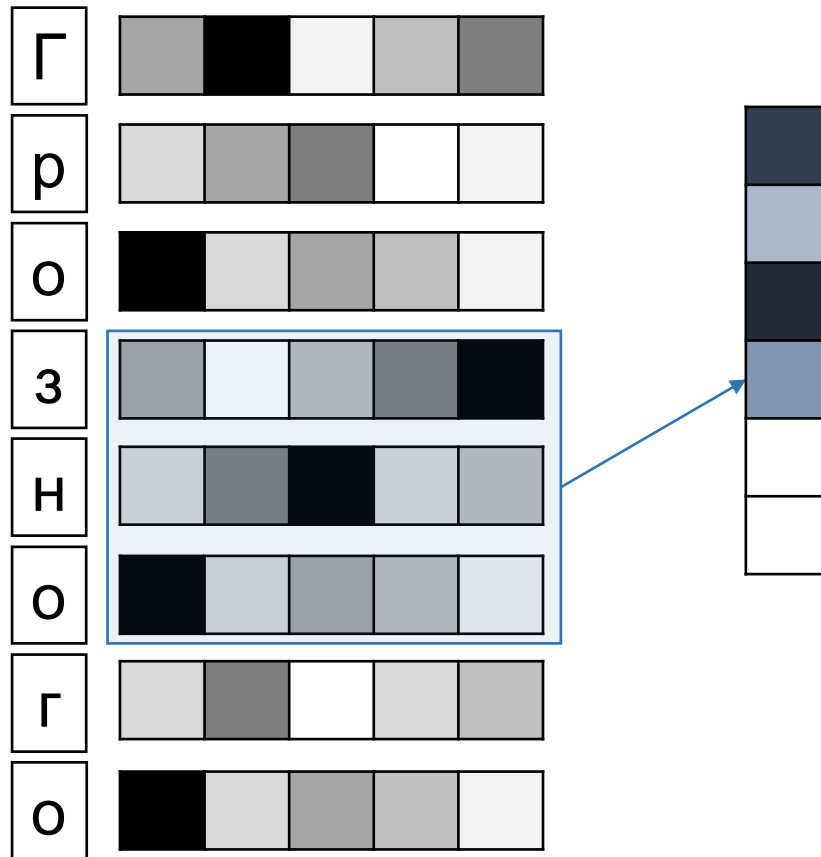
$$K_1 \in \mathbb{R}^{3 \times 5}$$



# CharCNN

Грозного

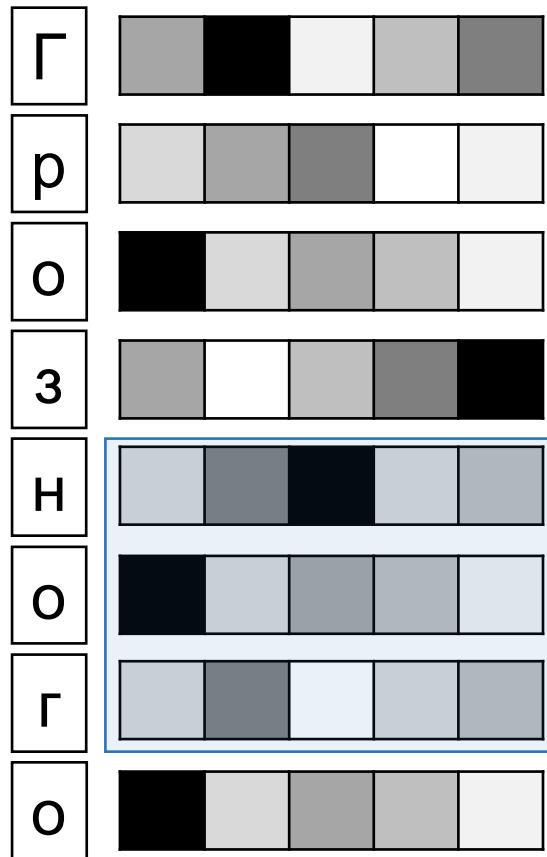
$$K_1 \in \mathbb{R}^{3 \times 5}$$



# CharCNN

Грозного

$$K_1 \in \mathbb{R}^{3 \times 5}$$

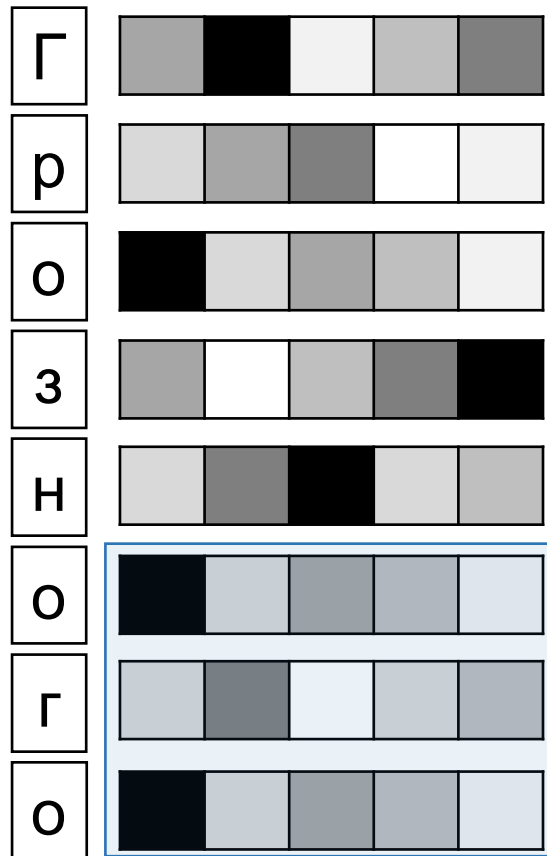




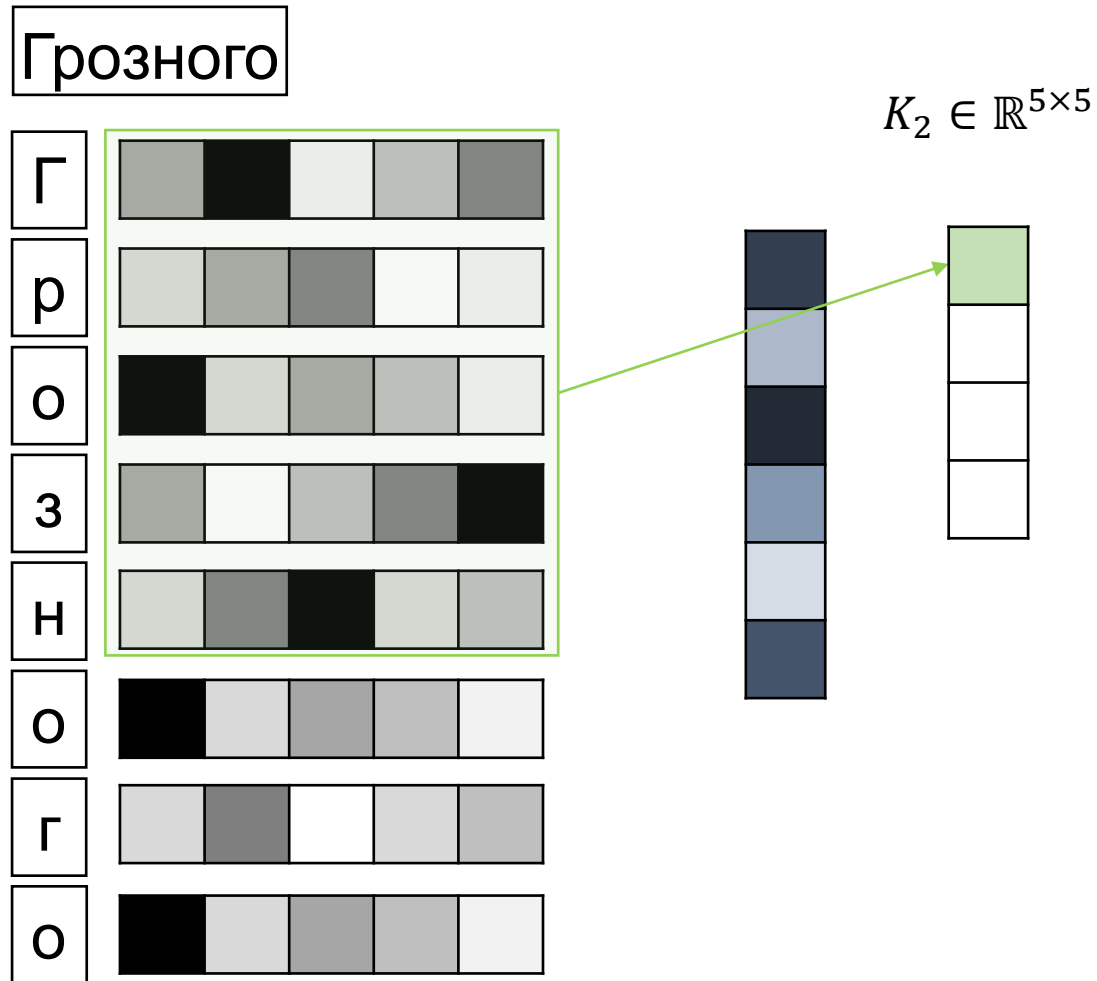
# CharCNN

Грозного

$$K_1 \in \mathbb{R}^{3 \times 5}$$

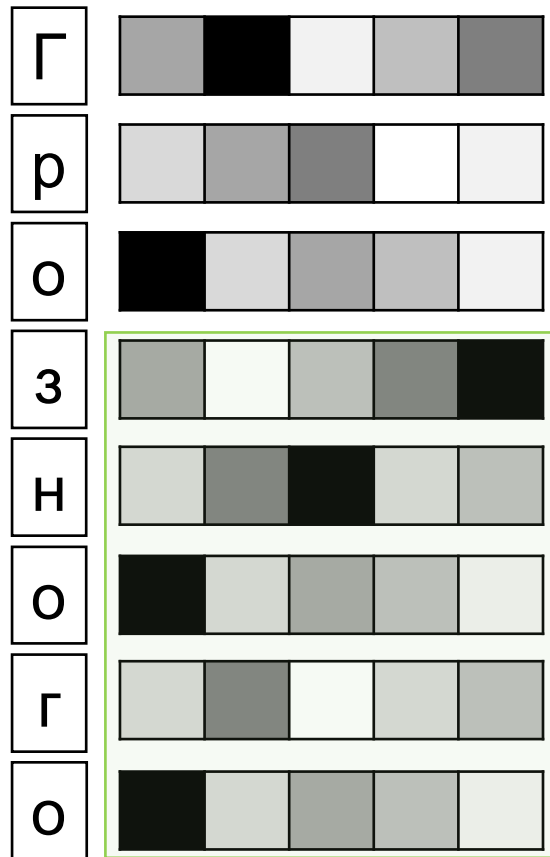


# CharCNN

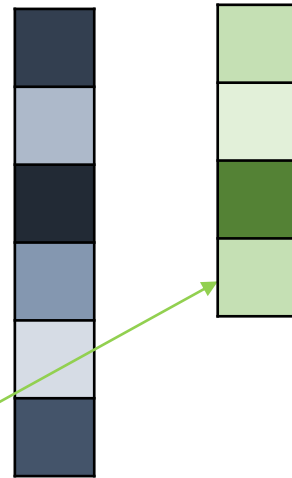


# CharCNN

Грозного

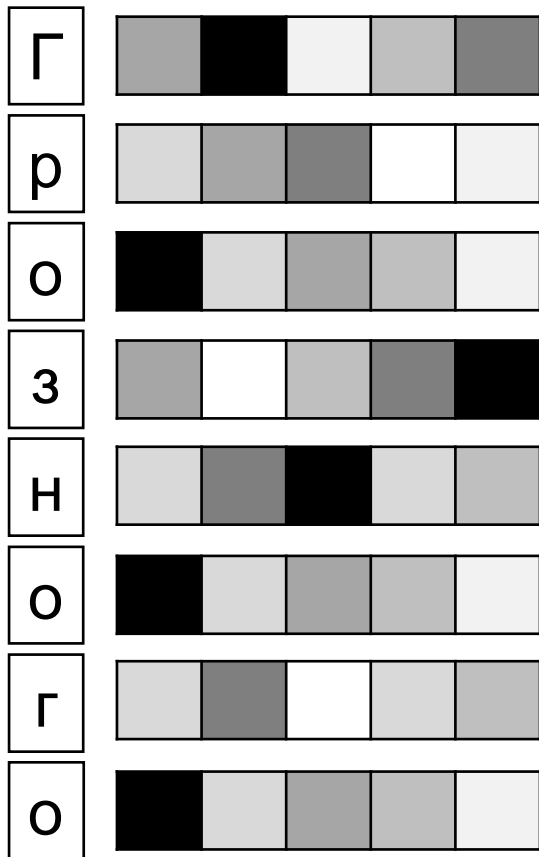


$$K_2 \in \mathbb{R}^{5 \times 5}$$



# CharCNN

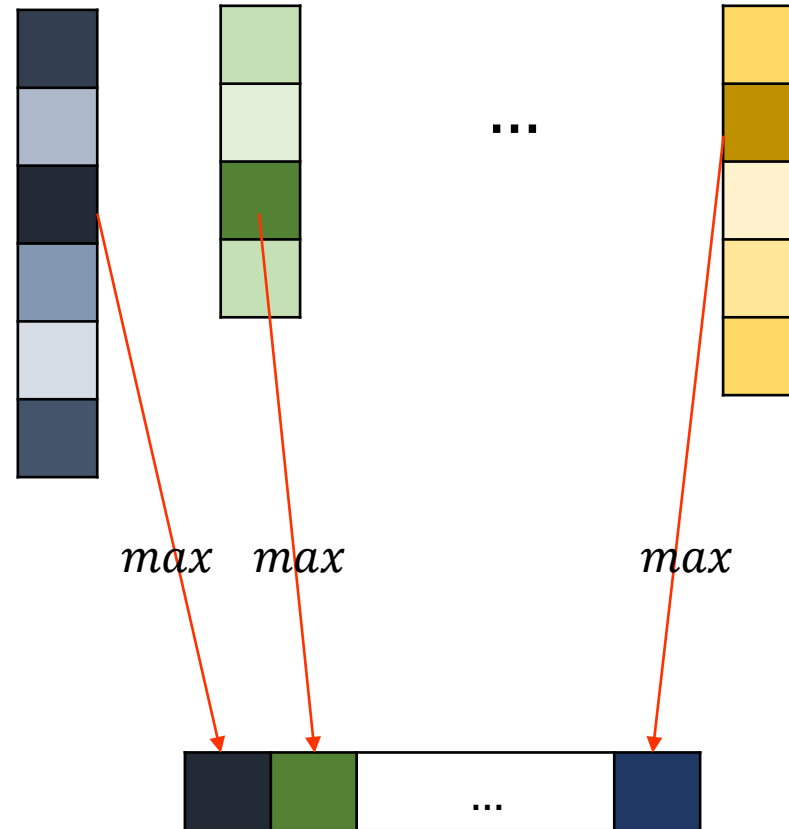
Грозного



$$K_1 \in \mathbb{R}^{3 \times 5}$$

$$K_2 \in \mathbb{R}^{5 \times 5}$$

$$K_m \in \mathbb{R}^{4 \times 5}$$



# Следующая лекция

Базовые задачи обработки текстов