ОСНОВЫ ОБРАБОТКИ ТЕКСТОВ

Лекция #3:
Синонимия: дистрибутивные векторные представления слов

Лектор: м.н.с. ИСП РАН Майоров Владимир Дмитриевич

Синонимия в задачах NLP

- 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)

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

One-hot вектора

- На предыдущих лекциях был предложен one-hot способ кодирования слова.
  - Пусть есть словарь всех слов языка $V = \{w_i\}$, размером $n = |V|$
  - Каждому слову из $w_i \in V$ ставится в соответствие вектор $\mathbb{R}^n$, в котором $w_{ij} = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases}, j = \overline{1, n}$

- все слова одинаково непохожи:

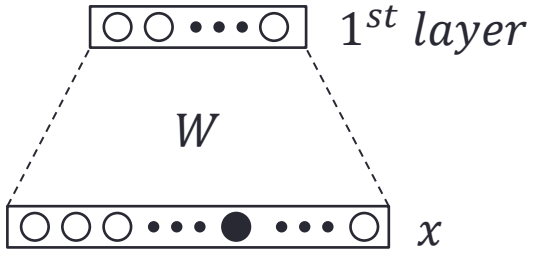
$$(w_i, w_j) = 0, \qquad i \neq j$$
$$similarity(w_i, w_j) = 0, \qquad i \neq j$$

Векторное представление слова

- Word embedding – вещественный вектор в пространстве с фиксированной невысокой размерностью
  - Пусть есть словарь всех слов языка $V = \{w_i\}$, размером $n = |V|$
  - Пусть задана фиксированная размерность пространства d (100, 200, 300)
  - Каждому слову из $w_i \in V$ ставится в соответствие вектор из $\mathbb{R}^d$

Embedding слой в нейронной сети

- На вход сети подается one-hot вектор
- Перед активацией первого слоя линейное преобразование Wx



- Проблема:
  - Для большинства задач NLP обучающих данных мало ⇒ построить «хорошую» матрицу W не удастся
- Решение:
  - Инициализировать матрицу W посчитанными заранее «хорошими» векторами

Векторное представление слова

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

Дистрибутивная гипотеза

- Слова, которые встречаются в схожих контекстах, имеют схожий смысл

... пил крепкий **кофе** у себя …
… подаётся чёрный **кофе** без всякого …
ведь не **кофе** пить зашёл
……

… умеренно крепкий **чай** с лимоном …
… завариваем чёрный **чай** кипящим …
… чтоб мне **чай** всегда пить …
……

Дистрибутивная гипотеза

- Слова, которые встречаются в схожих контекстах, имеют схожий смысл

- Контекстом слова может являться:
  - Соседние слова
    - Слева
    - Справа
    - Симметрично
  - Документ (параграф, предложение)

Матрица совместной встречаемости

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

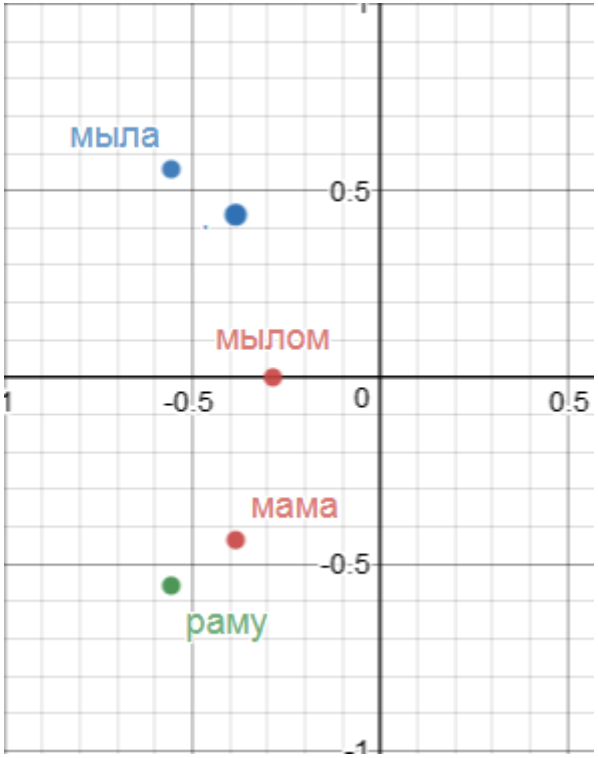
word-word

	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

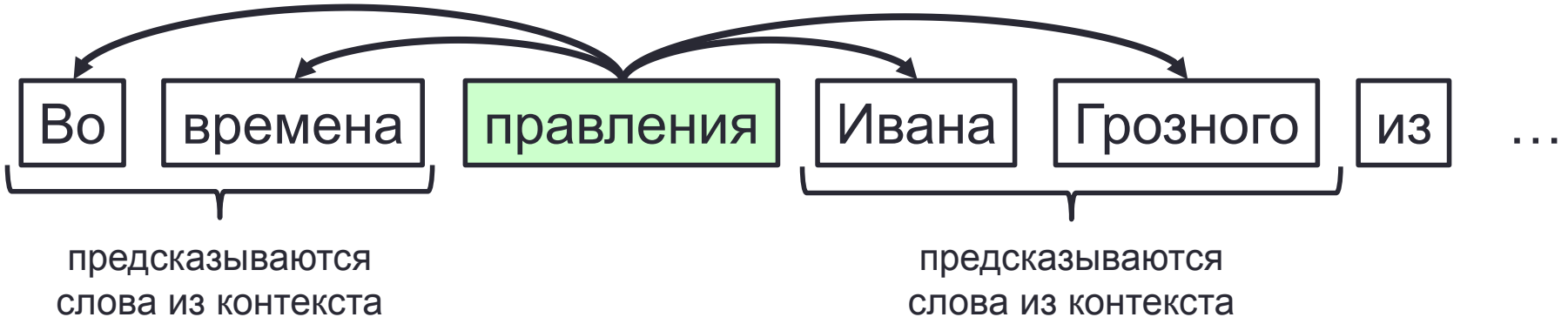
- Понижение размерности (SVD)
  - $A = U\Sigma V^T$;
    - Σ – матрица сингулярных значений (m*m)
  - $\hat{A} = \hat{U}\hat{\Sigma}\hat{V}^T$;
    - $\hat{U}$ – первые k столбцов матрицы U
    - $\hat{\Sigma}$ – k первых столбцов и строк матрицы Σ
    - $\hat{V}$ – первые k столбцов матрицы V

мама	-0.3851	-0.4352
мыла	-0.5574	0.5573
раму	-0.5574	-0.5573
мылом	-0.2862	0.0000
.	-0.3851	0.4352

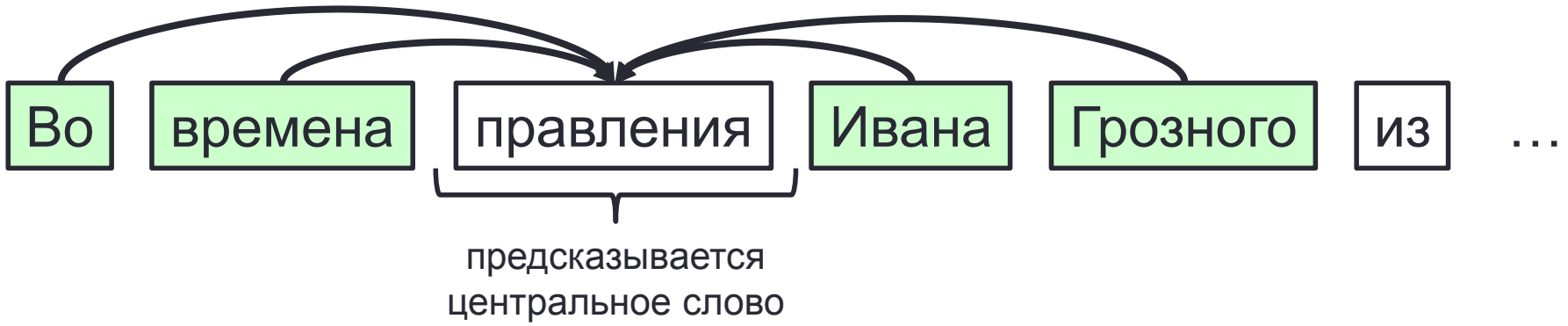


word2vec

- Continuous skip-gram

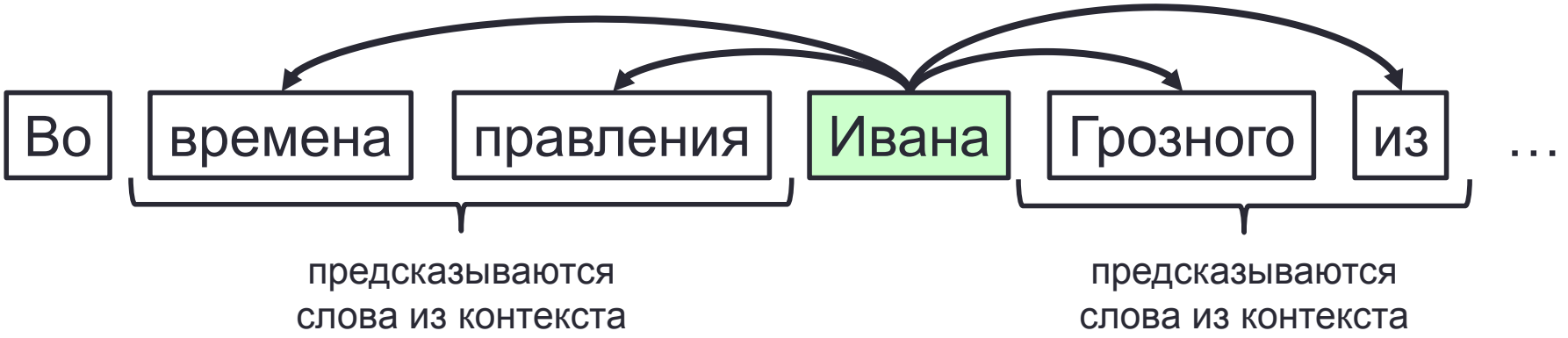


- Continuous bag of words

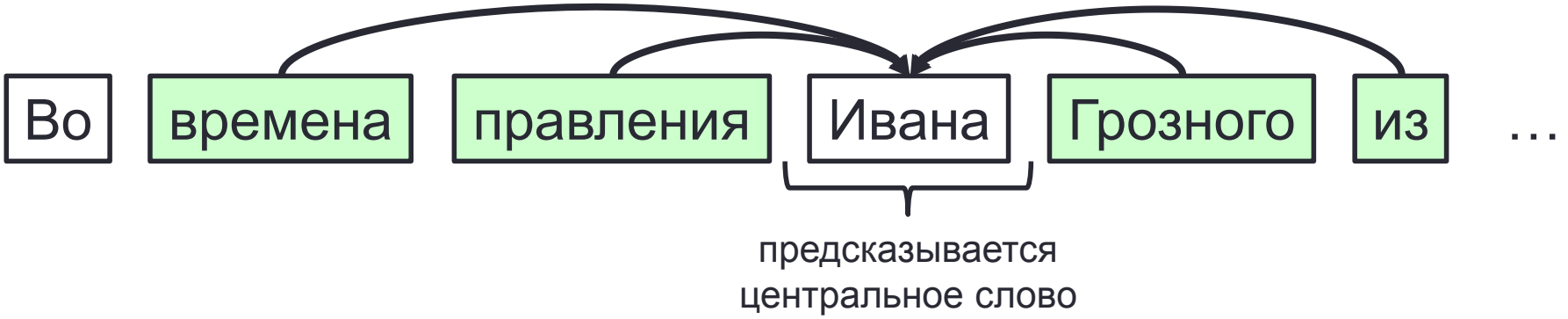


word2vec

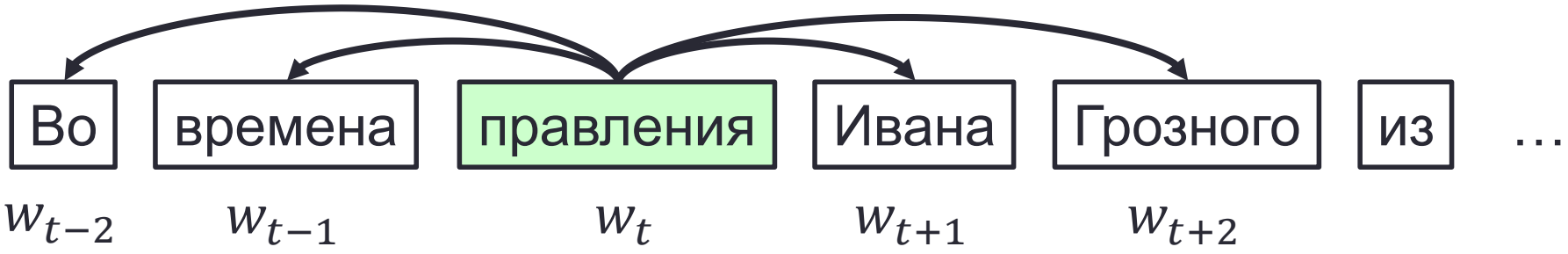
- Continuous skip-gram



- Continuous bag of words



word2vec skip-gram



- Цель: максимизировать логарифм вероятности каждого слова из контекста при данном центральном слове

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

- θ – оптимизируемые параметры

word2vec skip-gram

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

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

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

word2vec skip-gram

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

- 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

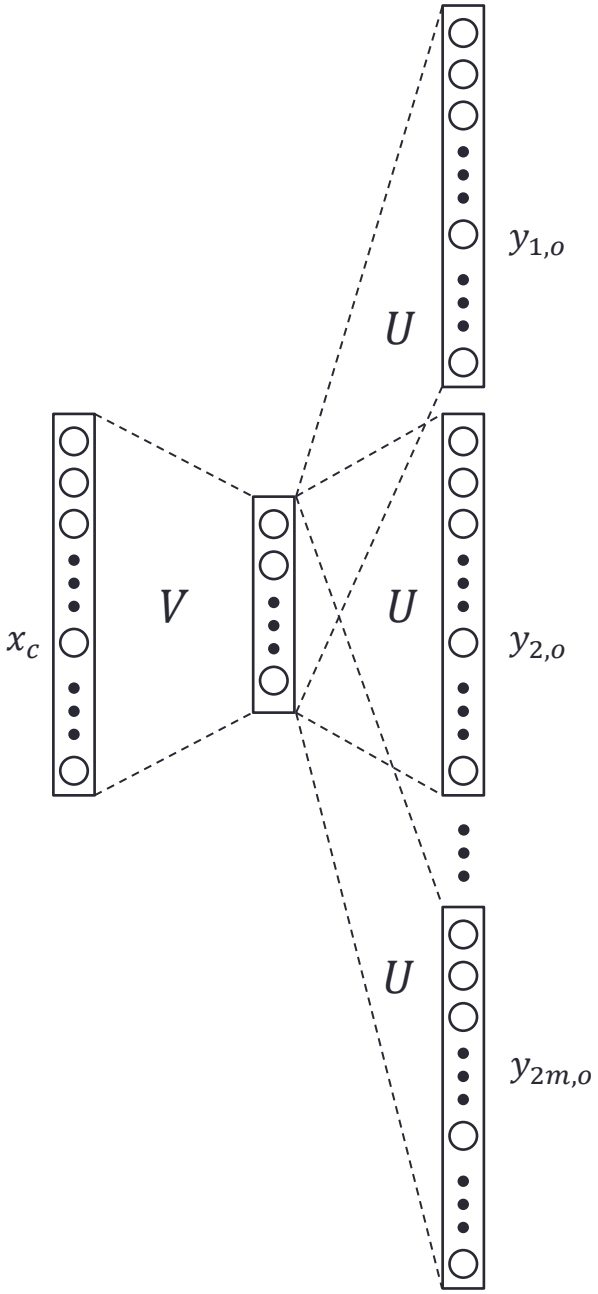
$$\frac{\partial(-\log p(o|c))}{\partial v_c} = -u_o + \sum_{i=1}^{n}\frac{\exp\left(u_i^T v_c\right)}{\sum_{w=1}^{n}\exp(u_w^T v_c)}u_i =$$

$$= -u_o + \sum_{x\in V} p(x|c)u_x$$

word2vec skip-gram

Модель может быть представлена в виде нейронной сети:

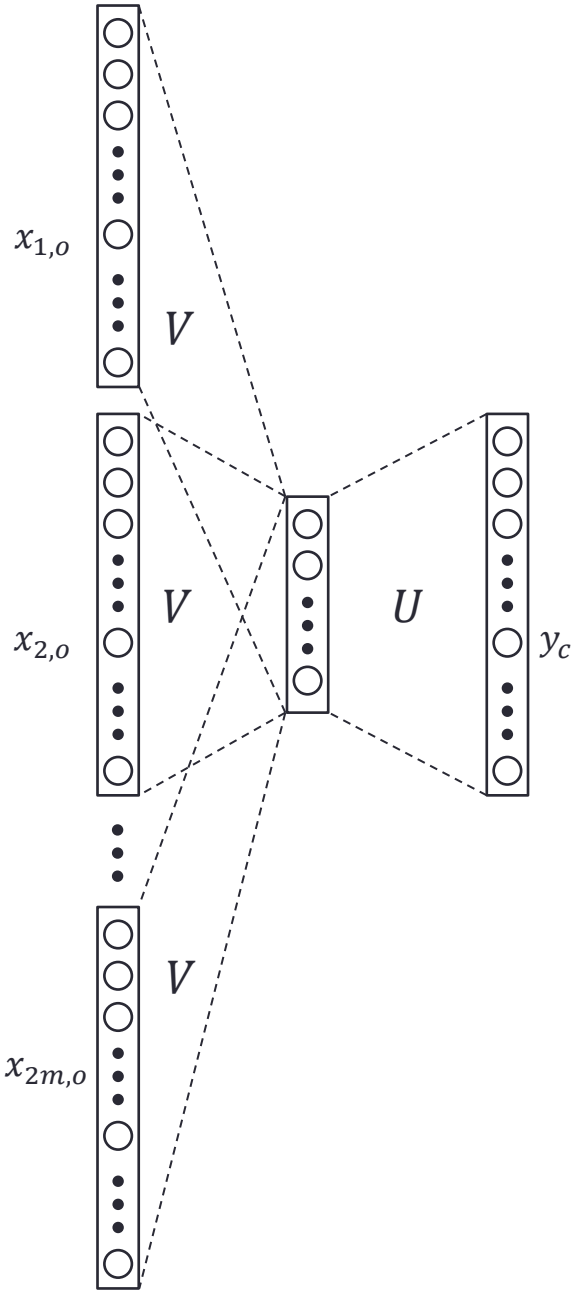
- **Вход**: one-hot центрального слова
- **Скрытый слой**: линейный
- **Выходной слой**: softmax
- **Функция ошибки**: cross-entropy



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 skipgram

- Проблемы:
  - На каждом шаге градиентного спуска вычисляется

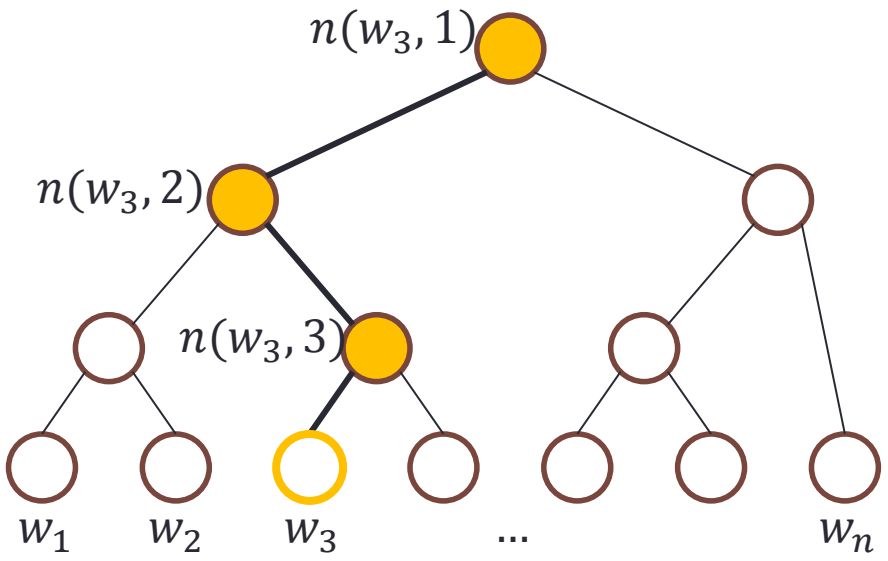
$$\sum_{w=1}^{n} \exp(u_w^T v_c)$$

    словарь большой ⇒ долго

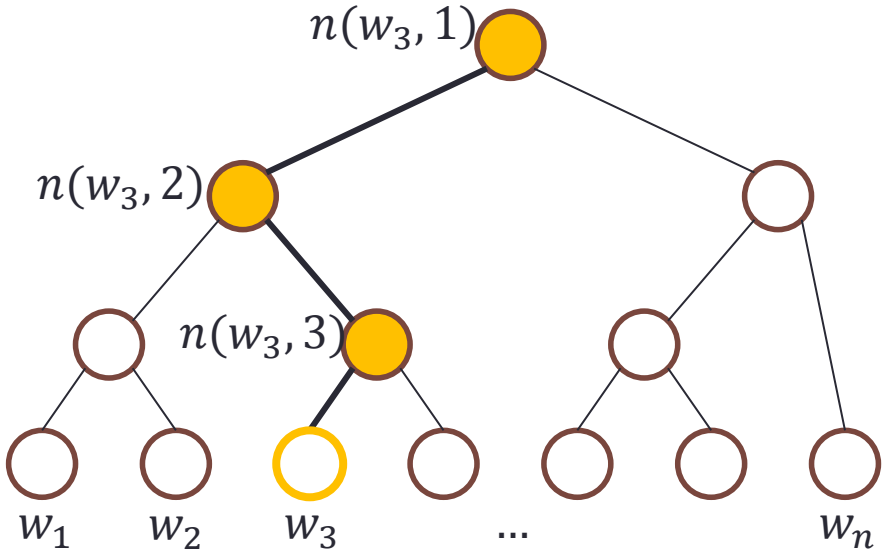
- Решения:
  - Hierarchical softmax
  - Negative sampling

Hierarchical softmax

- Идея:
  - Составить из словаря бинарное дерево
  - Предсказывать путь в дереве вместо слова из словаря



Hierarchical softmax



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

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

Negative sampling

- Решаем более простую задачу бинарной классификации:

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

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

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

Negative sampling

- 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
Ивана	правления	0
	…	

Negative sampling

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

$$P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=1}^{n} f(w_i)^{3/4}}$$

Проблема редких слов

- Для слов, которые встречаются слишком редко невозможно построить хорошие вектора
- Такие слова заменяются на специальную константу OOV (out of vocabulary)
- Вычисляется вектор для OOV, этот вектор в будущем используется для слов, не вошедших в V

Проблема частых слов

- Слишком частые слова (предлоги, союзы, пунктуация)
  - часто встречаются в корпусе ⇒ вносят большое влияние на вектора слов
  - встречаются во всевозможных контекстах ⇒ вектора не отражают семантику слова

- Решение:
  - Выкидывать слишком частые слова из корпуса

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

  - t – порог частоты (обычно около 10^{-5})

GloVe

- Строится на основе word-word матрицы совместной встречаемости

- X_{ij} – количество употреблений слова w_j встретилось в контексте слова w_i
- $X_i = \sum_k X_{ik}$ – количество слов в контексте слова w_i

- Функция потерь:

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

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

Оценка качества

- Задача аналогии слов:
  - Слово a относится к слову b также, как слово c относится к слову ___.
- Метрика:
  - $Accuracy$
- Синтаксические аналогии:

a	b	c	___
лекция	лекции	семинар	***семинары***
бежать	бегущий	лежать	***лежащий***

Оценка качества

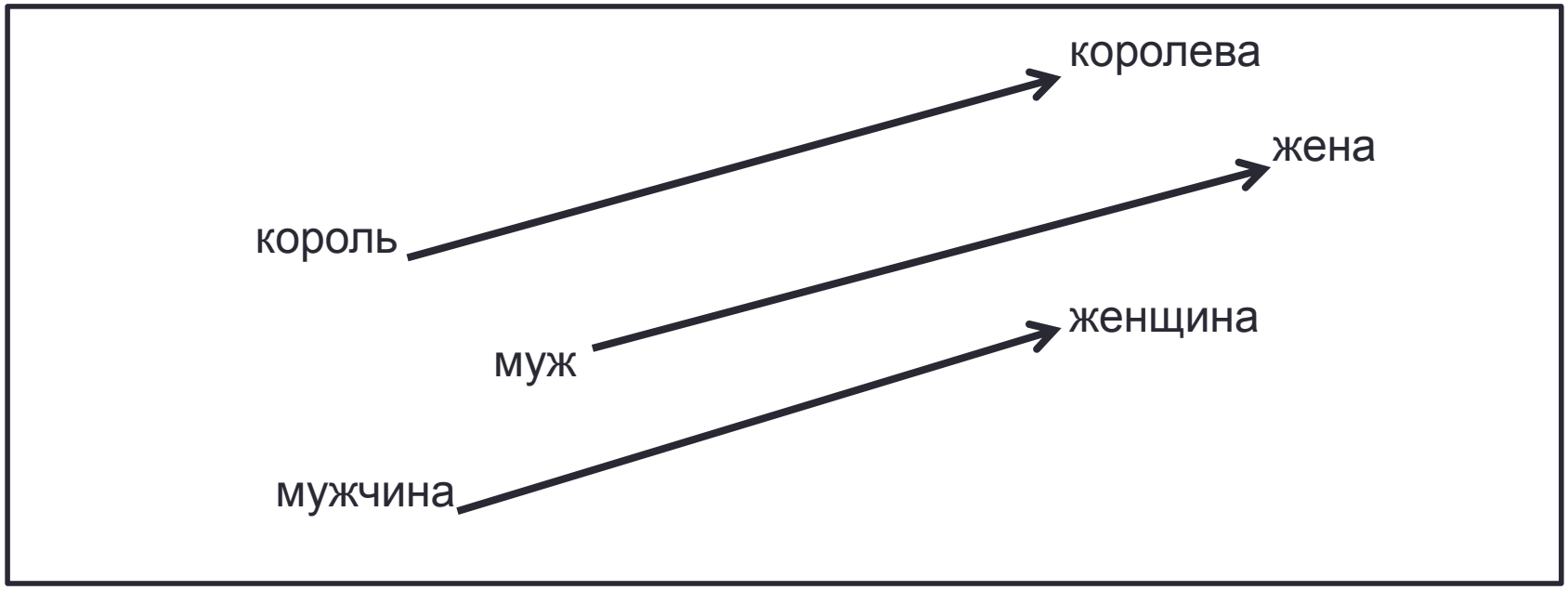
- Задача аналогии слов:
  - Слово a относится к слову b также, как слово c относится к слову ___.
- Метрика:
  - $Accuracy$
- Семантические аналогии:

a	b	c	___
лететь	плыть	самолет	***корабль***
Россия	Москва	Франция	***Париж***

Задача аналогии слов

Слово a относится к слову b также, как слово c относится к слову d.

$$v_d \approx v_b - v_a + v_c$$



Следующая лекция

- Символьные представления слов

- Лектор: Андрианов Иван Алексеевич