

# **Глубинное обучение**

**Обработка естественного языка: эмбеддинги и языковые  
модели**

**Ирина Сапарина**

# NLP: приложения

≡ Google Translate ⋮

Text Documents

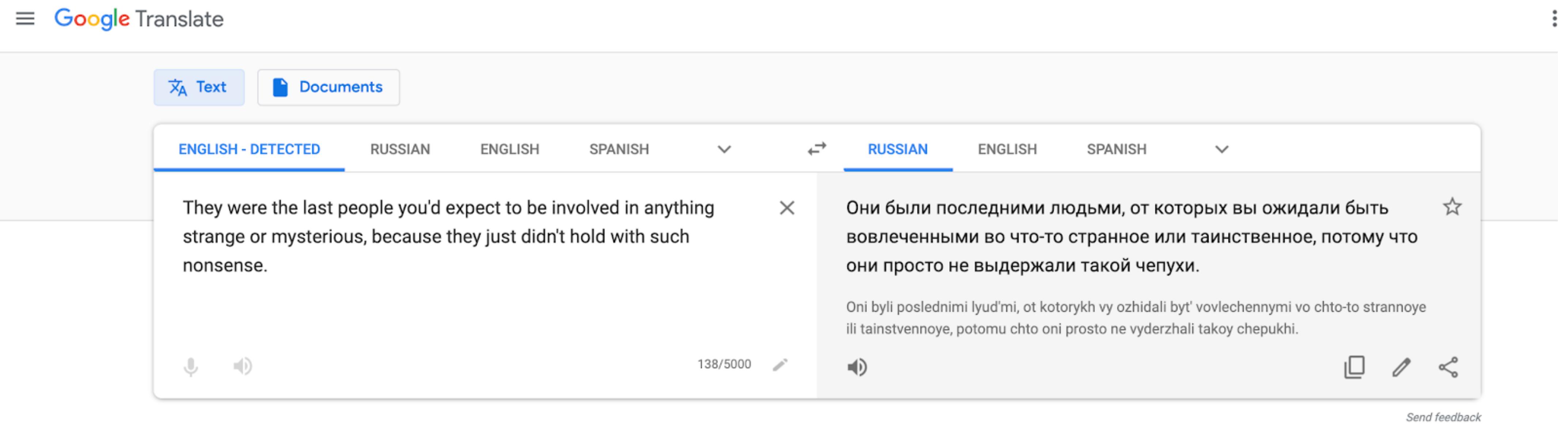
ENGLISH - DETECTED RUSSIAN ENGLISH SPANISH RUSSIAN ENGLISH SPANISH

They were the last people you'd expect to be involved in anything strange or mysterious, because they just didn't hold with such nonsense.

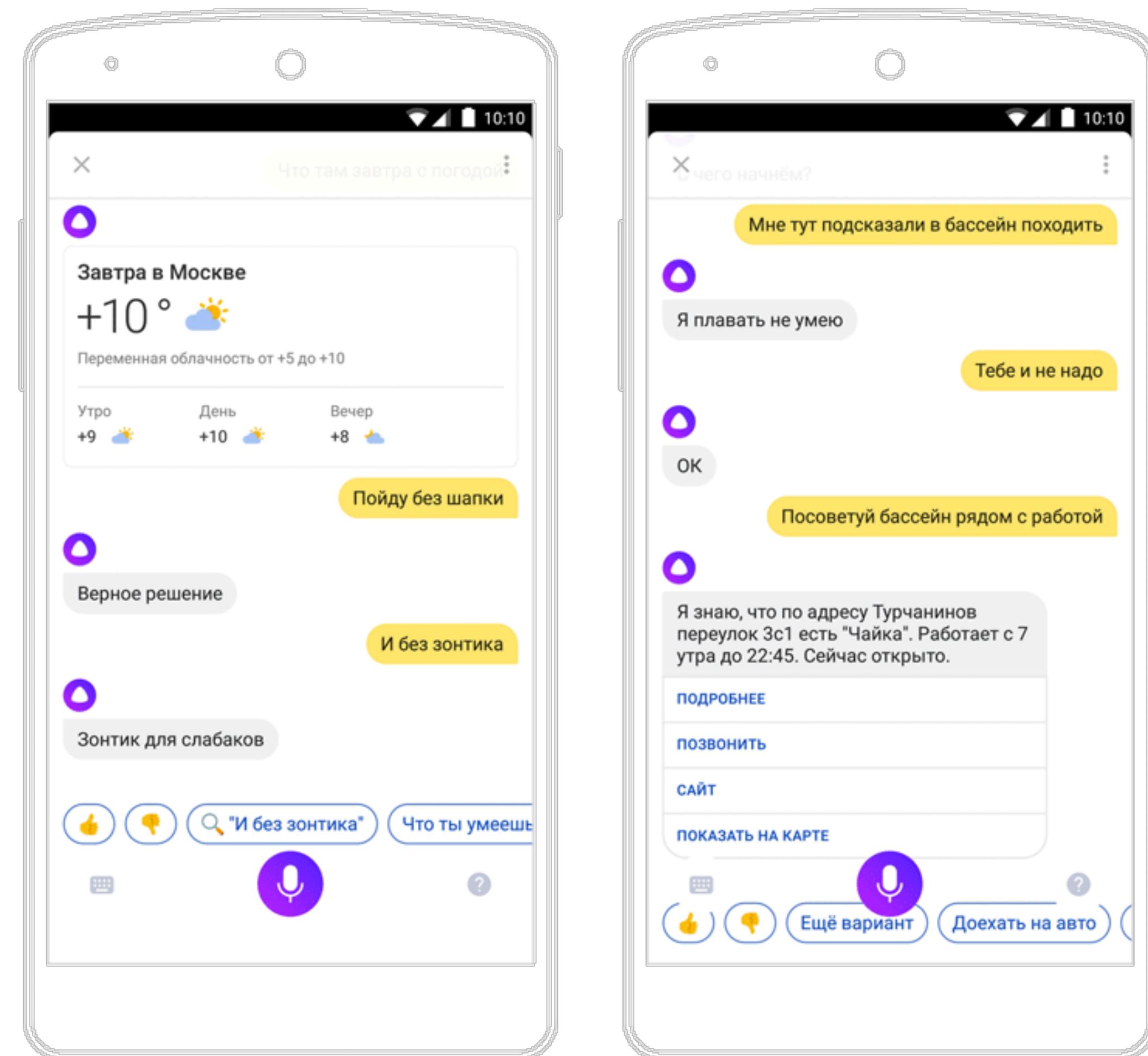
Они были последними людьми, от которых вы ожидали быть вовлеченными во что-то странное или таинственное, потому что они просто не выдержали такой чепухи.

Oni byli poslednimi lyud'mi, ot kotorikh vy ozhidali byt' vovlechennymi vo chto-to strannoye ili tainstvennoye, potomu chto oni prosto ne vyderzhali takoy chepukhi.

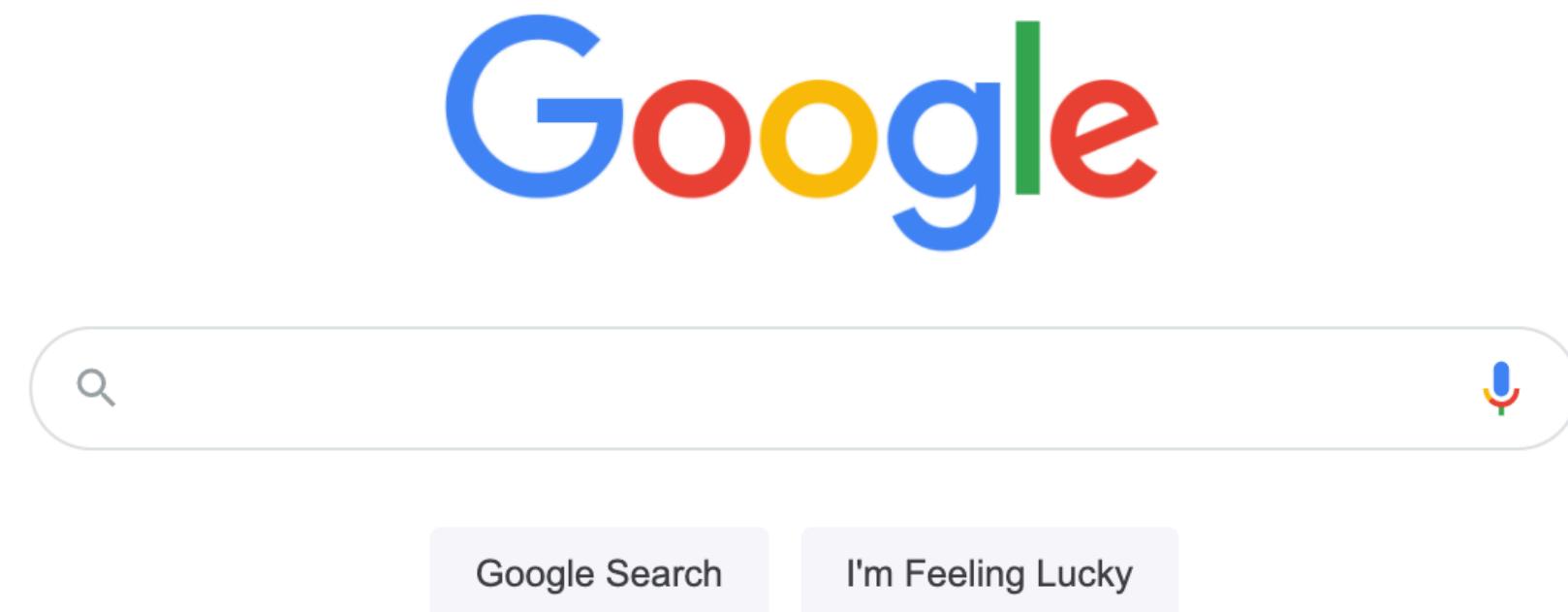
Send feedback



# NLP: приложения



# NLP: приложения



# NLP: задачи

## Классификация:

- Part-of-Speech Tagging
  - Named-Entity Recognition
  - Sentiment Analysis

....

## Ousted WeWork founder Adam Neumann lists his Manhattan penthouse for \$37.5 million

Finished fixing my twitter...I had to unfollow and follow everyone again	Negative
@DinahLady I too, liked the movie! I want to buy the DVD when it comes out	Positive

# NLP: задачи

## Классификация:

- Part-of-Speech Tagging
  - Named-Entity Recognition
  - Sentiment Analysis

....

Why	not	tell	someone	?
adverb	adverb	verb	noun	punctuation mark, sentence closer
<b>Ousted WeWork founder Adam Neumann lists his Manhattan penthouse for \$37.5 million</b>				
[organization]	[person]	[location]	[monetary value]	
Finished fixing my twitter...I had to unfollow and follow everyone again				Negative
@DinahLady I too, liked the movie! I want to buy the DVD when it comes out				Positive

# Генерация (следующая лекция):

- Machine Translation
  - Question Answering

Language Modeling

# **Представления текста**

# Представления текста

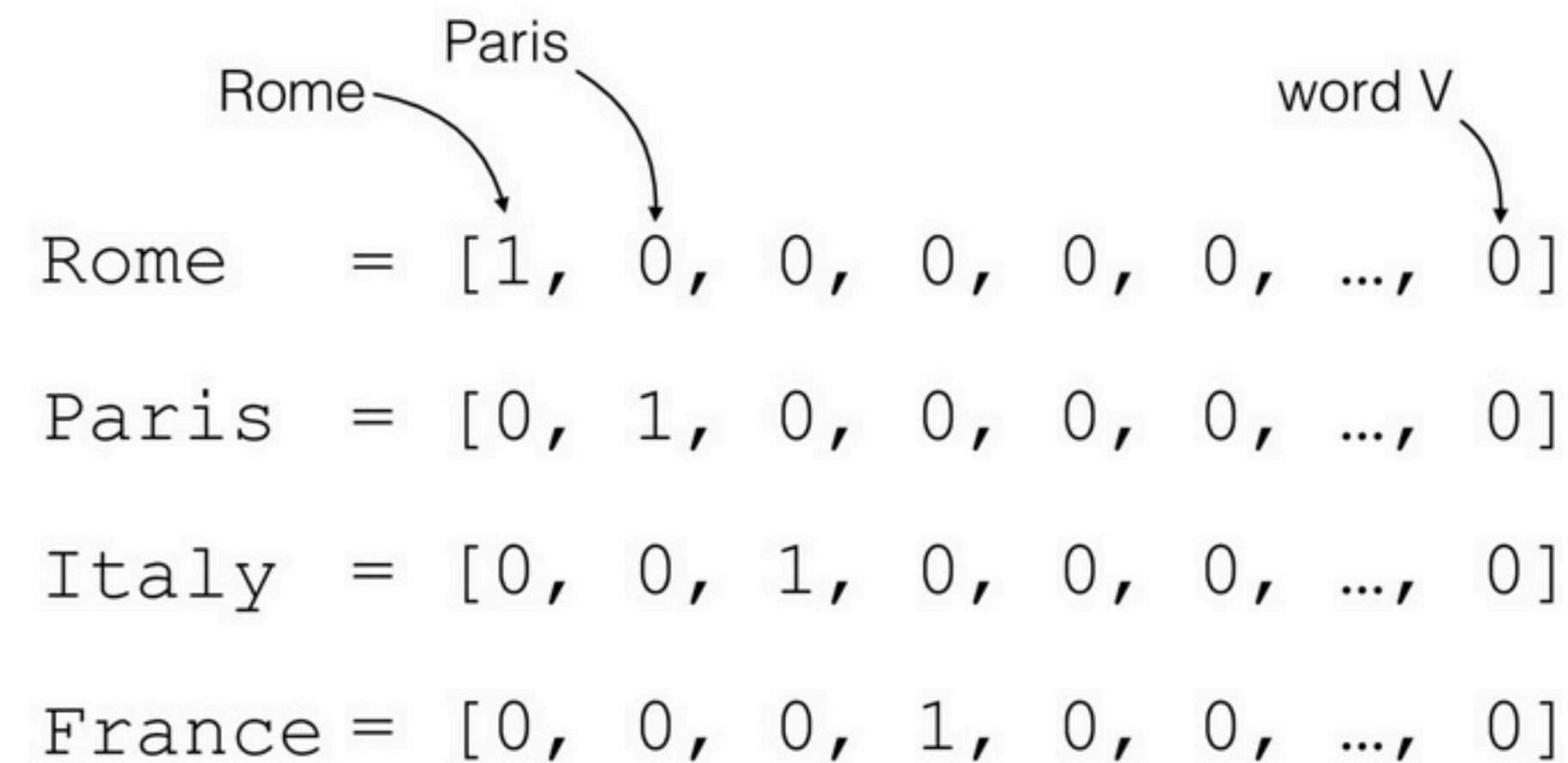
Хотим: перевести слова в числовой вид

# Представления текста

Хотим: перевести слова в числовой вид

## One-hot encoding

Создаем словарь всех слов (размера  $V$ ) и нумеруем их



# Представления текста

Хотим: перевести слова в числовой вид

## One-hot encoding

Создаем словарь всех слов (размера  $V$ ) и нумеруем их

The diagram illustrates the one-hot encoding of four words: Rome, Paris, Italy, and France. Above each word, an arrow points to its corresponding index in a vector. The vector for Rome has a 1 at index 1 and 0s elsewhere. The vector for Paris has a 1 at index 2 and 0s elsewhere. The vector for Italy has a 1 at index 3 and 0s elsewhere. The vector for France has a 1 at index 4 and 0s elsewhere. An arrow labeled "word V" points to the final index of the vectors.

$$\begin{aligned} \text{Rome} &= [1, 0, 0, 0, 0, 0, \dots, 0] \\ \text{Paris} &= [0, 1, 0, 0, 0, 0, \dots, 0] \\ \text{Italy} &= [0, 0, 1, 0, 0, 0, \dots, 0] \\ \text{France} &= [0, 0, 0, 1, 0, 0, \dots, 0] \end{aligned}$$

Все вектора ортогональны друг другу  $\cos_{\text{sim}}(w_1, w_2) = \frac{w_1^T w_2}{\|w_1\|^2 \|w_2\|^2} = 0$

# Представления текста

Хотим: перевести слова в числовой вид

Идея: вектора слов, которые встречаются в одном контексте, должны быть близки  $\text{cos\_sim}(w_1, w_2) \approx w_1^T w_2$

\_\_\_\_\_ *is the most beautiful capital city in Europe*

*Rome? Paris?*

# Представления текста

Хотим: перевести слова в числовой вид

Идея: вектора слов, которые встречаются в одном контексте, должны быть близки  $\cos_{\text{sim}}(w_1, w_2) \approx w_1^T w_2$

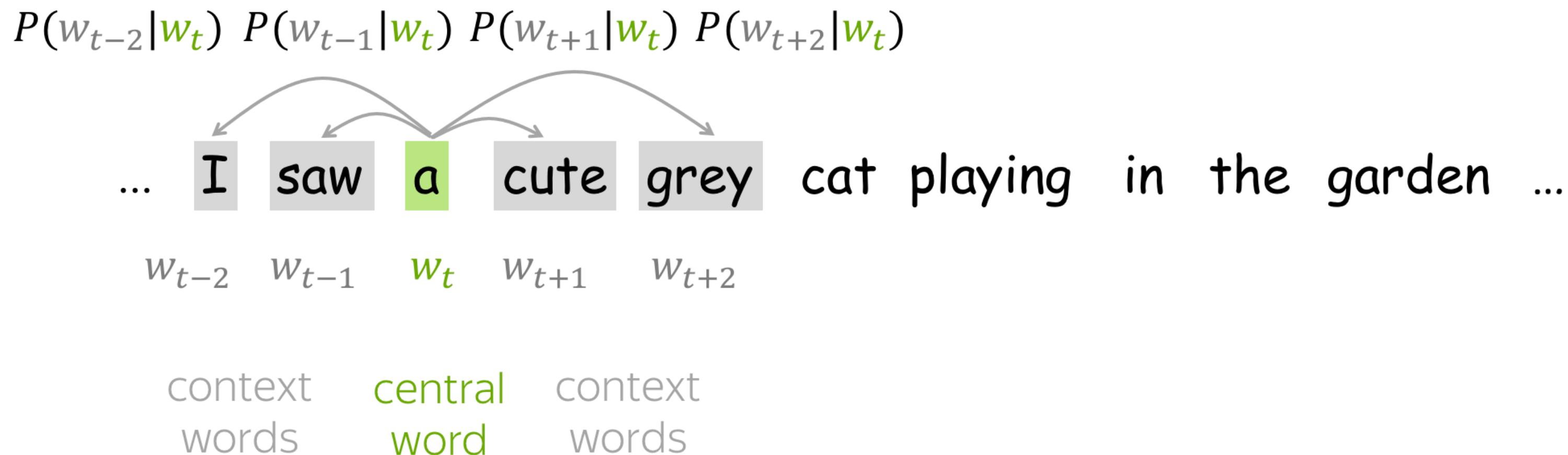
$$\mathbf{expect} = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \\ 0.487 \end{pmatrix}$$



# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**



[Image credit](#)

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**



[Image credit](#)

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

Loss: negative log-likelihood      
$$J(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_{t+j} | w_t; \theta)$$

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

Loss: negative log-likelihood

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

Как задать вероятность?

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

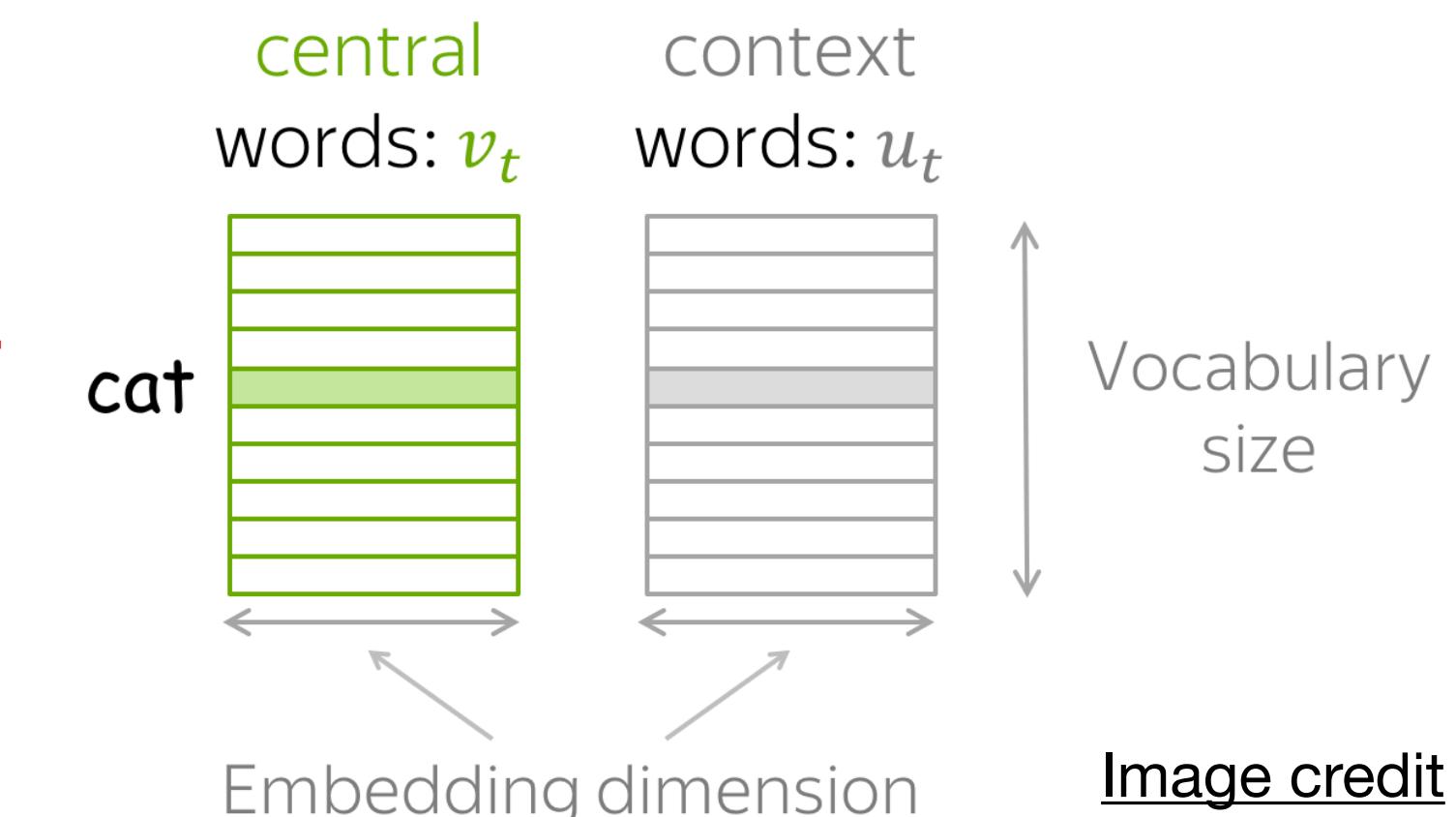
Loss: negative log-likelihood

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

This is our  $\theta!$   
All  $v_t$  and  $u_t$  together

Для каждого слова в словаре будут два вектора: **v** и **u**

**v** - если слово в центре, **u** - есть это слово из контекста



# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

Loss: negative log-likelihood      
$$J(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_{t+j} | w_t; \theta)$$

Вероятность слова из **контекста о** при условии **центра с**:

$$P(o | c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

Loss: negative log-likelihood      
$$J(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_{t+j} | w_t; \theta)$$

Вероятность слова из **контекста о** при условии **центра с**:

$$P(o | c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

$\text{cos\_sim}(w_1, w_2) \approx w_1^T w_2$

Больше значение - больше вероятность

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

Loss: negative log-likelihood      
$$J(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_{t+j} | w_t; \theta)$$

Вероятность слова из **контекста о** при условии **центра с**:

$$P(o | c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

Нормализация по словарю (softmax)

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

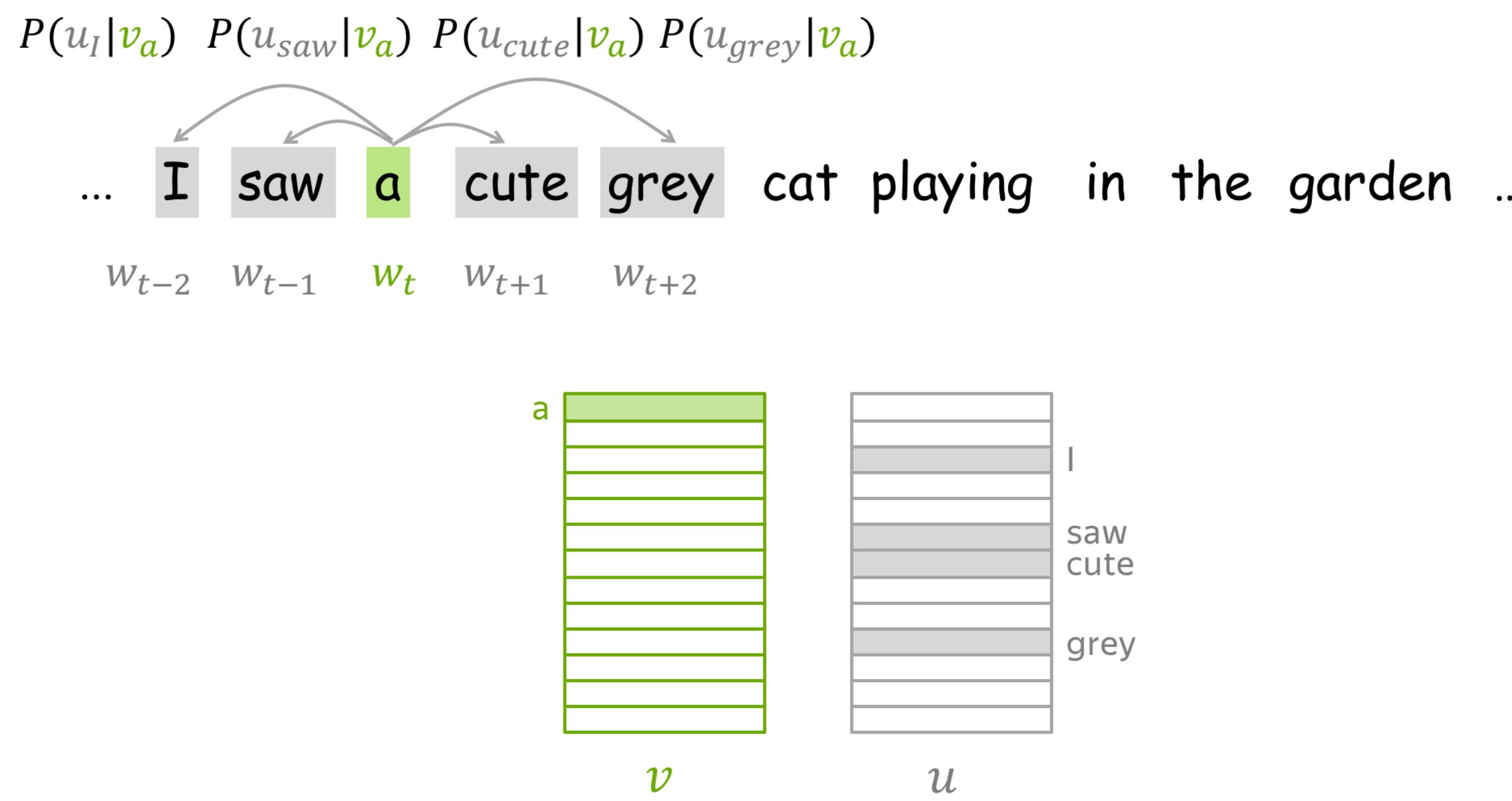
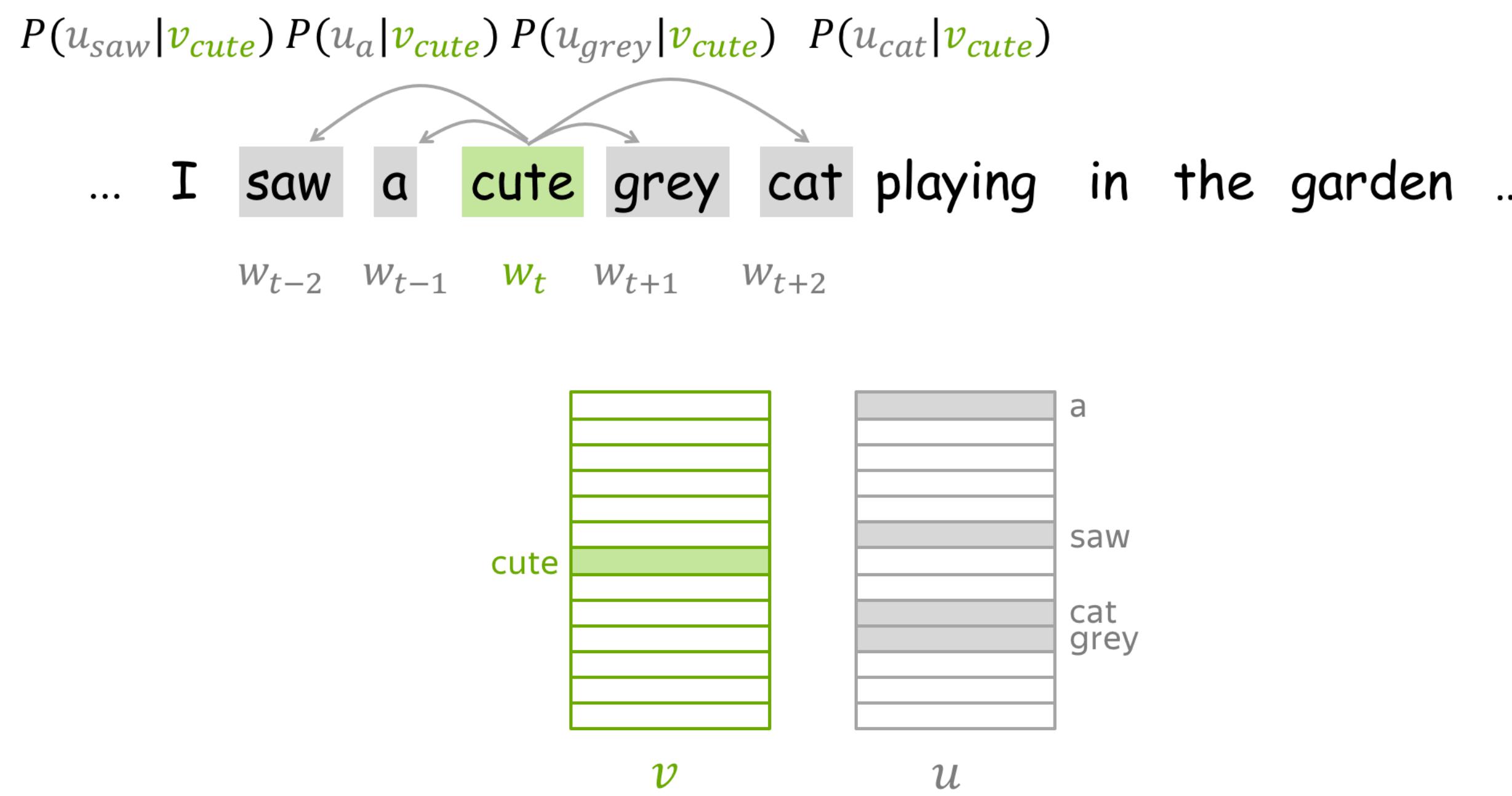


Image credit

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**



[Image credit](#)

# Word2Vec

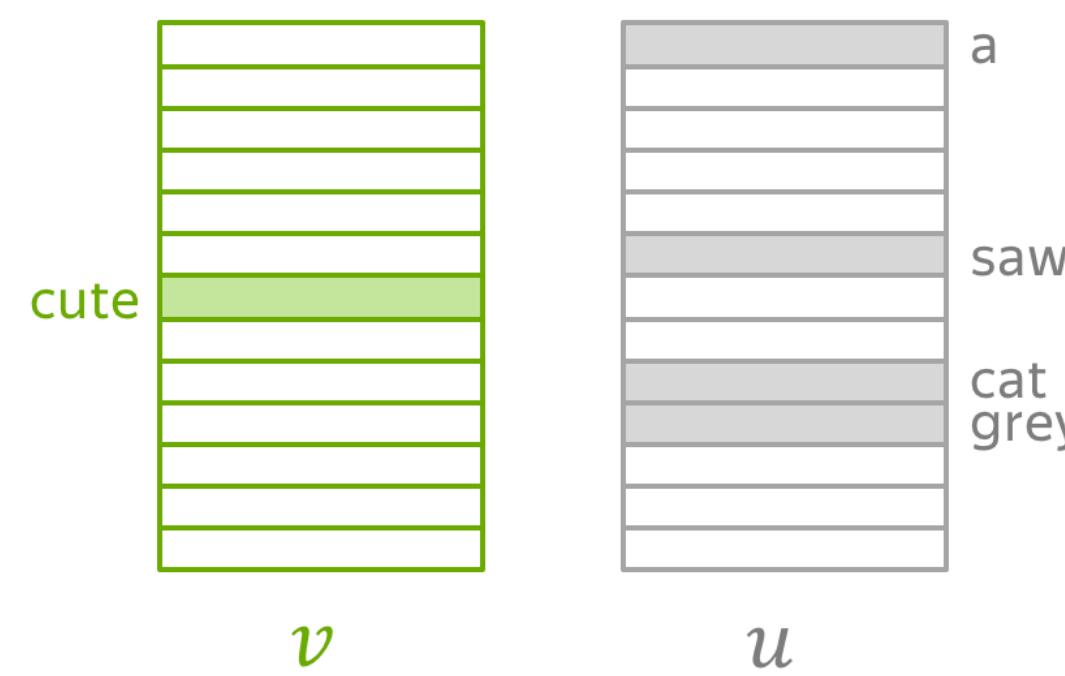
Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

Loss: negative log-likelihood

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

Обучаем градиентным спуском  $u, v(\theta)$



[Image credit](#)

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

Loss: negative log-likelihood       $J(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_{t+j} | w_t; \theta)$

$$P(o | c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

Нормализация по словарю - долго

# Word2Vec

Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому корпусу текстов** и вычислять вероятность **центрального** слова при условии **контекста**

Loss: negative log-likelihood      
$$J(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log P(w_{t+j} | w_t; \theta)$$

**Улучшения:**

- Hierarchical Softmax

$$P(o | c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

Нормализация по словарю - долго

- Negative Sampling

$$\sum_{w \in V} \exp(u_w^T v_c) \xrightarrow{\text{ }} \sum_{w \in \{o\} \cup S_k} \exp(u_w^T v_c)$$

Нормализация по выбранным “negative” словам

# Word2Vec

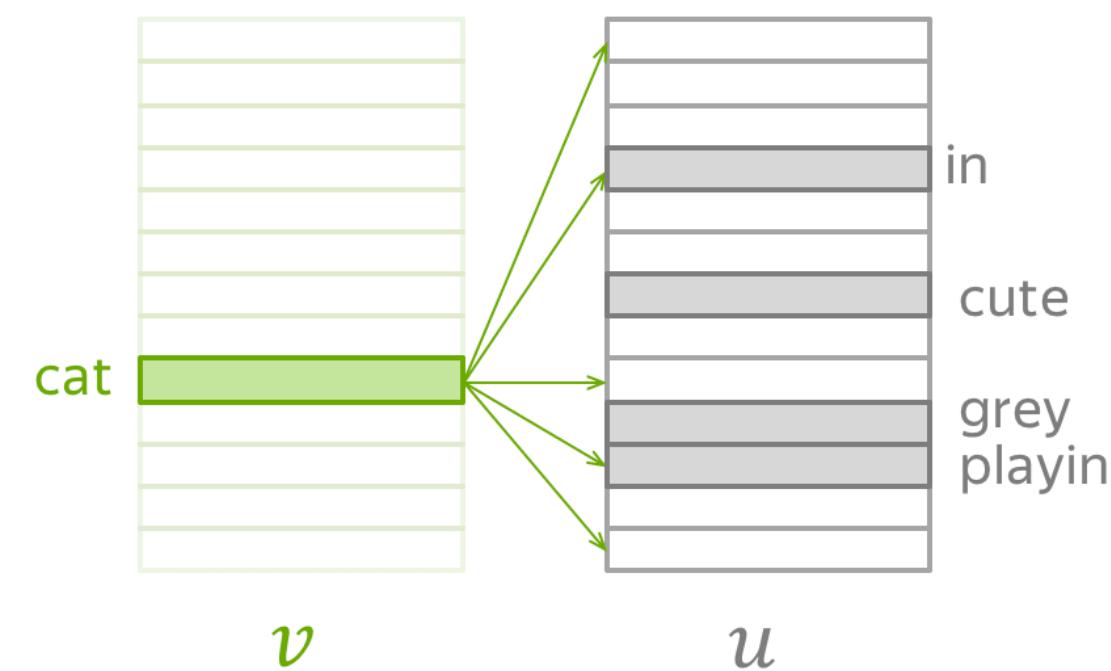
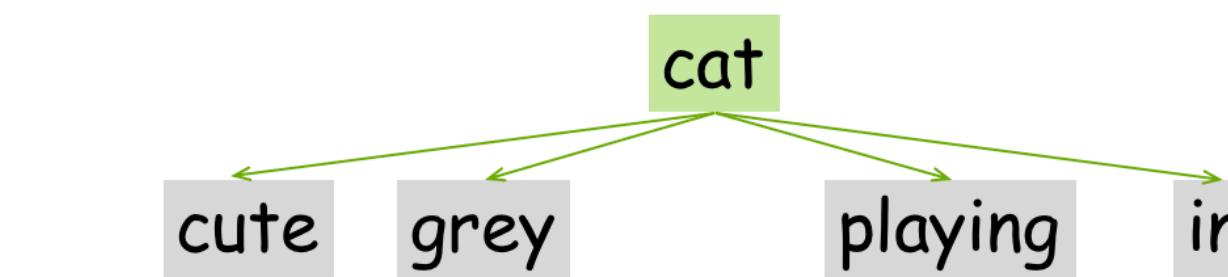
Хотим: перевести слова в числовой вид

Будем проходить **окном** по **большому** корпусу **текстов** и вычислять вероятность **центрального** слова при условии **контекста** - **Skip-Gram**

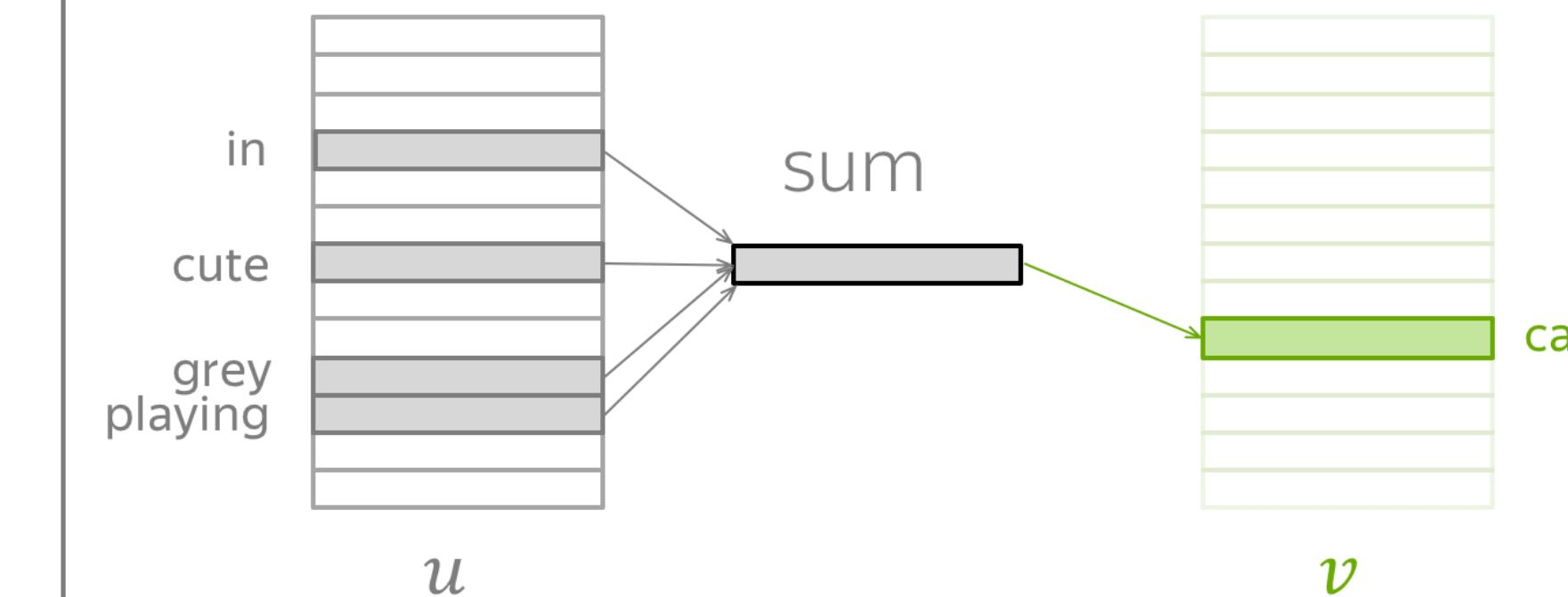
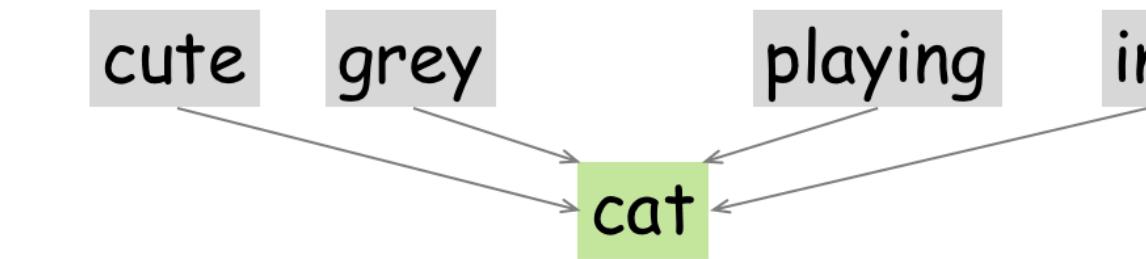
# Word2Vec

Хотим: перевести слова в числовой вид

... I saw a cute grey cat playing in the garden ...



Skip-Gram: from central predict context  
(one at a time)



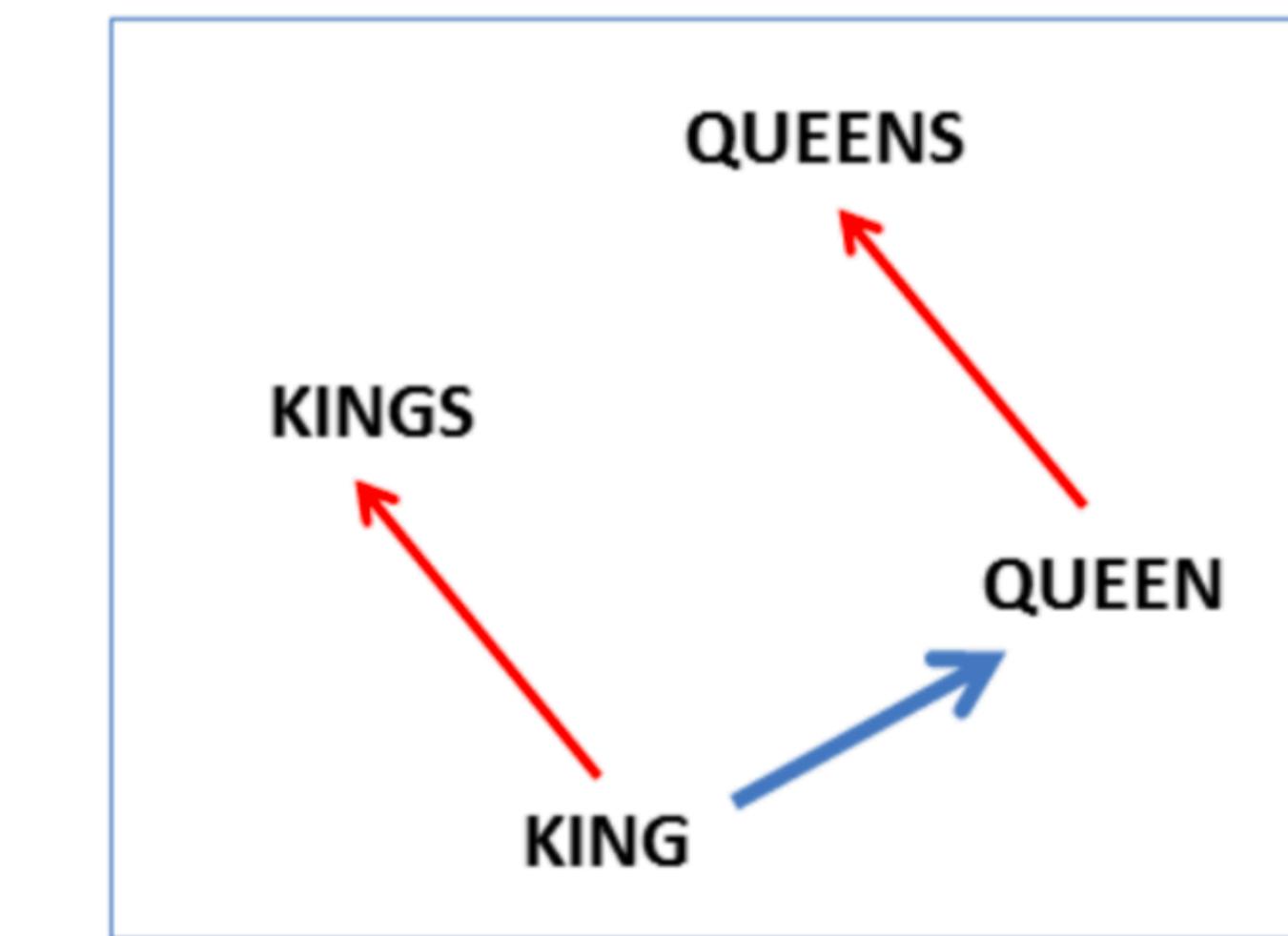
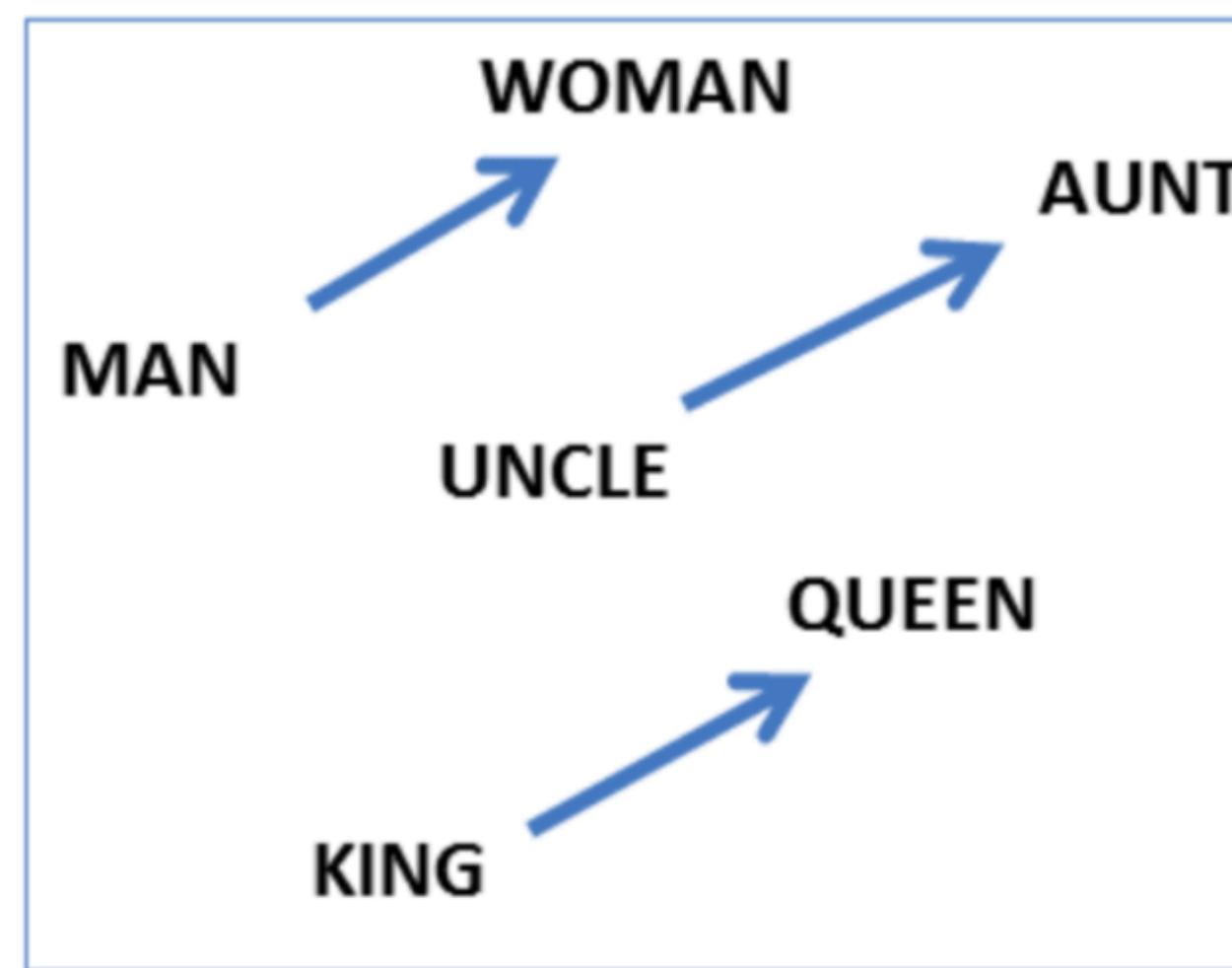
CBOW: from sum of context predict central

[Image credit](#)

# Word2Vec

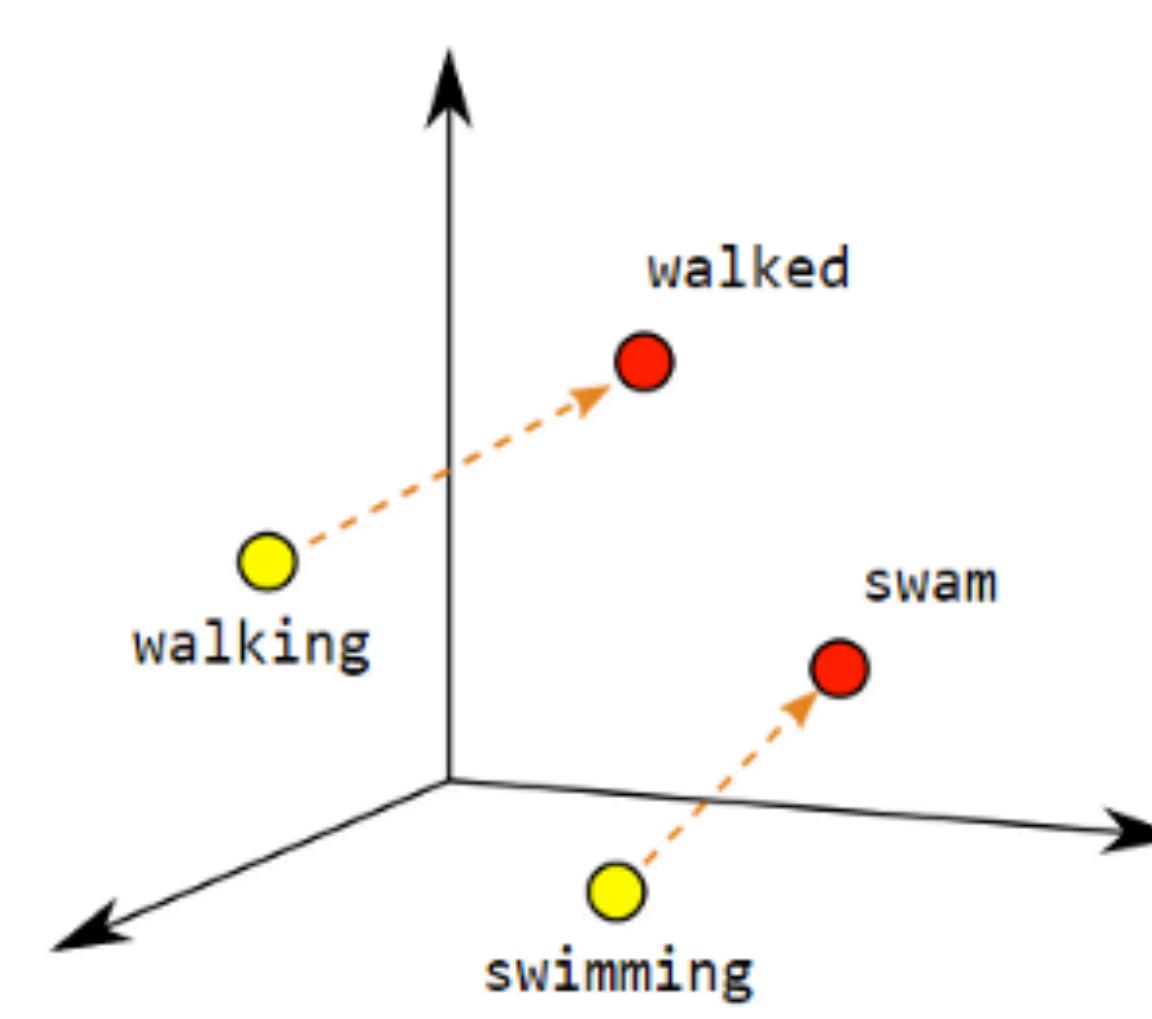
semantic:  $v(\text{king}) - v(\text{man}) + v(\text{woman}) \approx v(\text{queen})$

syntactic:  $v(\text{kings}) - v(\text{king}) + v(\text{queen}) \approx v(\text{queens})$

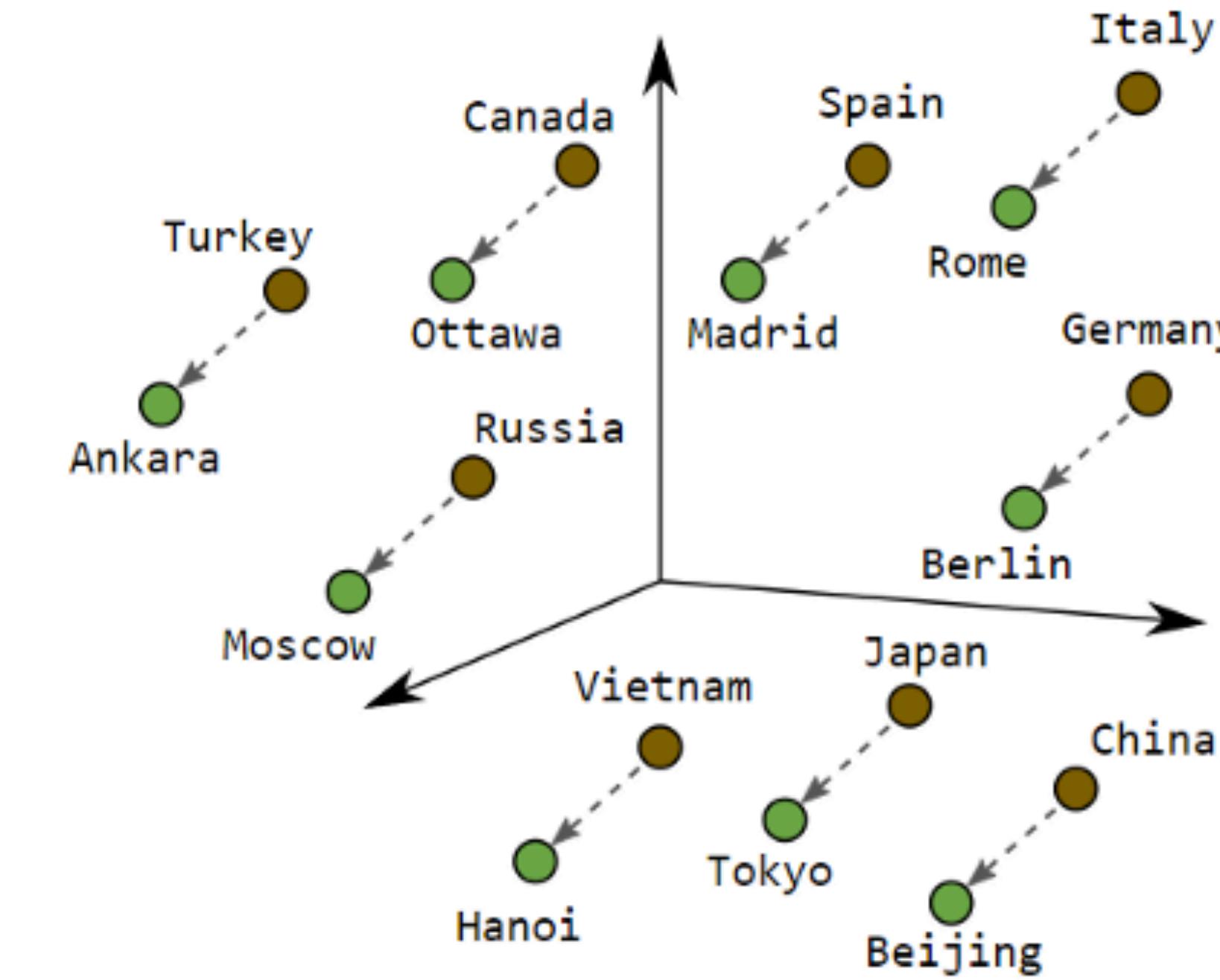


[Image credit](#)

# Word2Vec



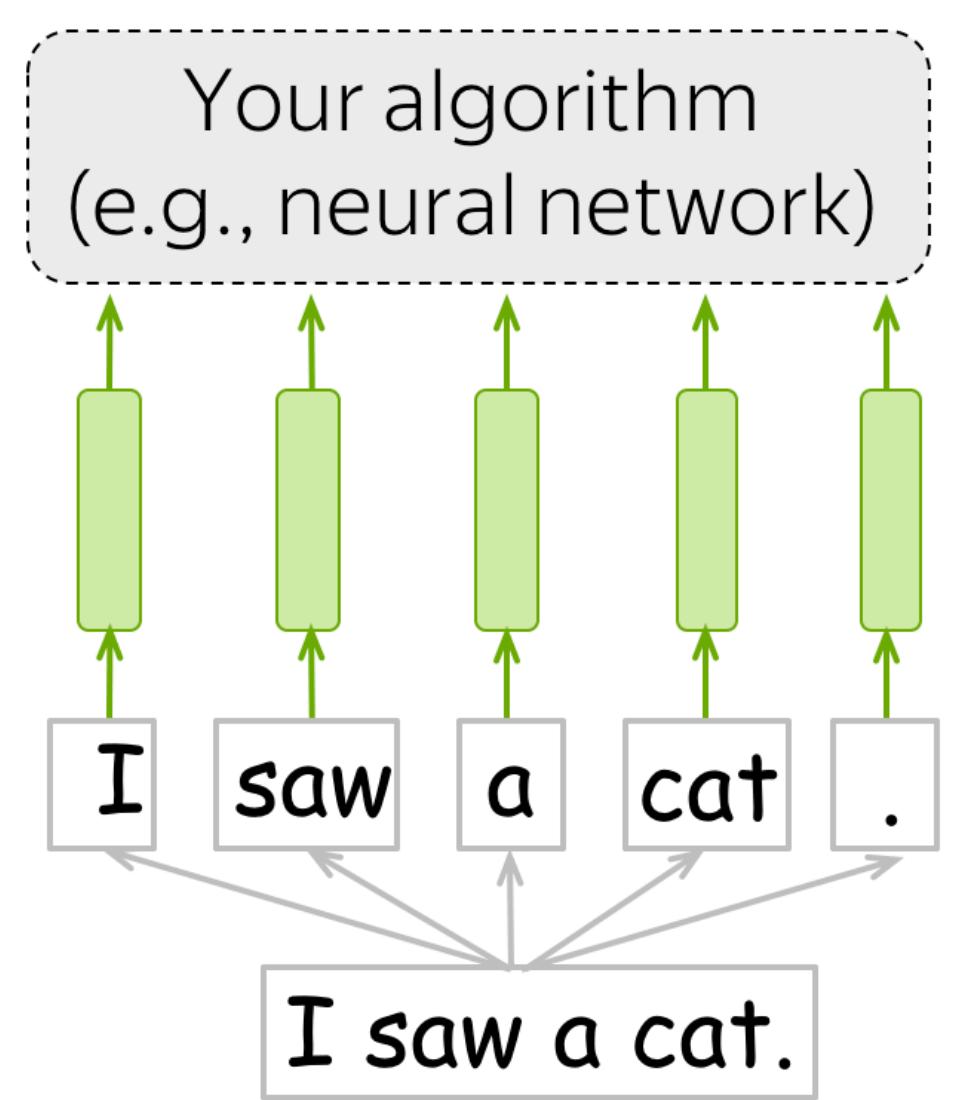
Verb Tense



Country-Capital

[Image credit](#)

# Word2Vec



Any algorithm for solving a task

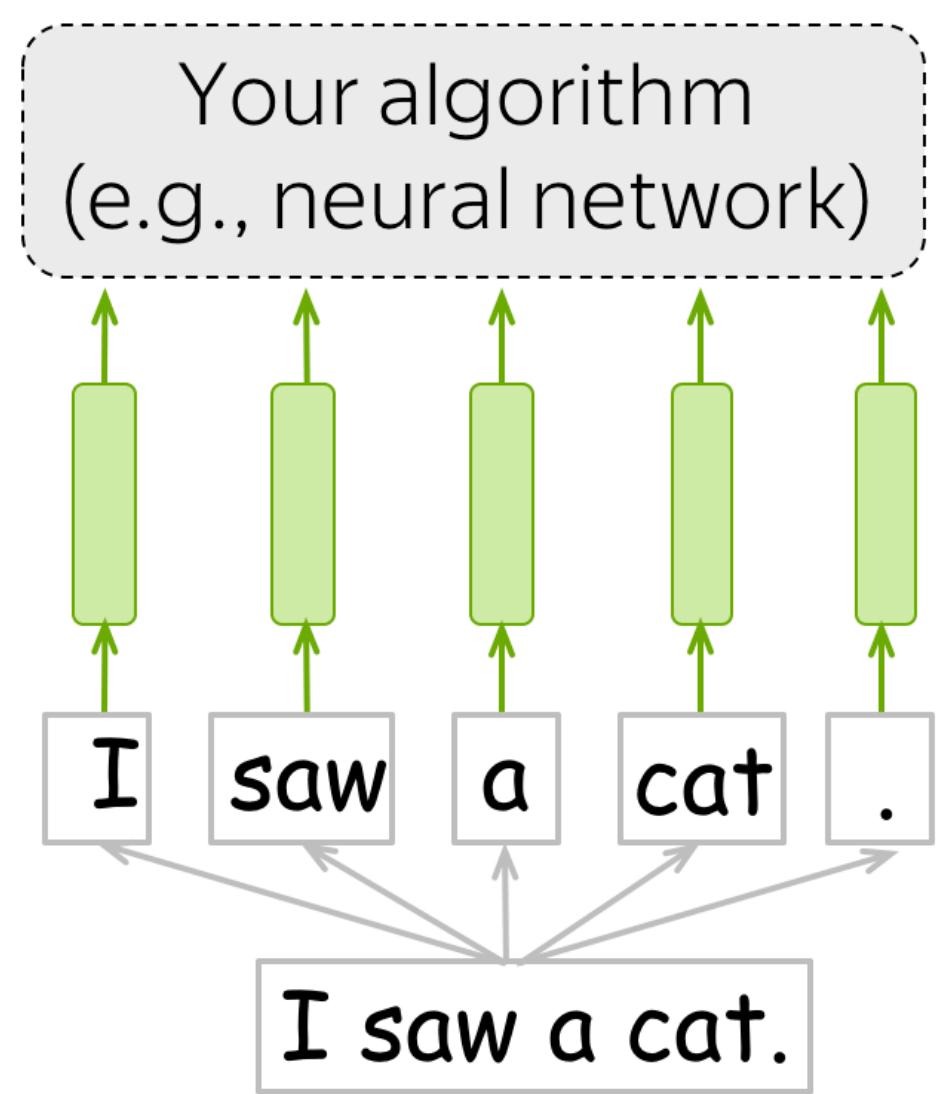
Word representation - vector  
(input for your model/algorithm)

Sequence of tokens

Text (your input)

[Image credit](#)

# Word2Vec

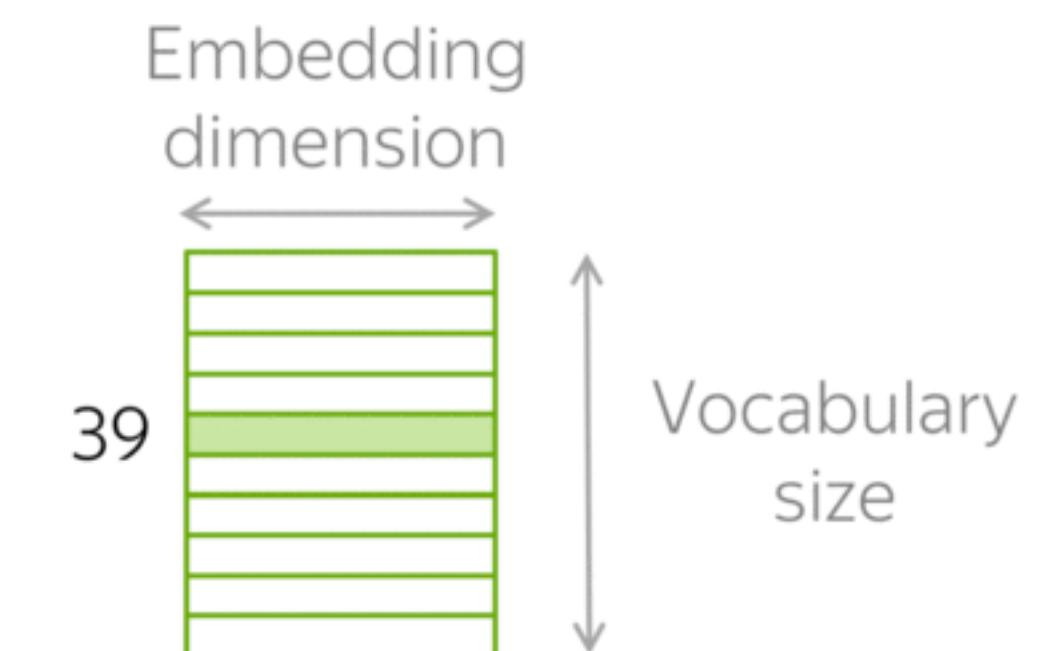
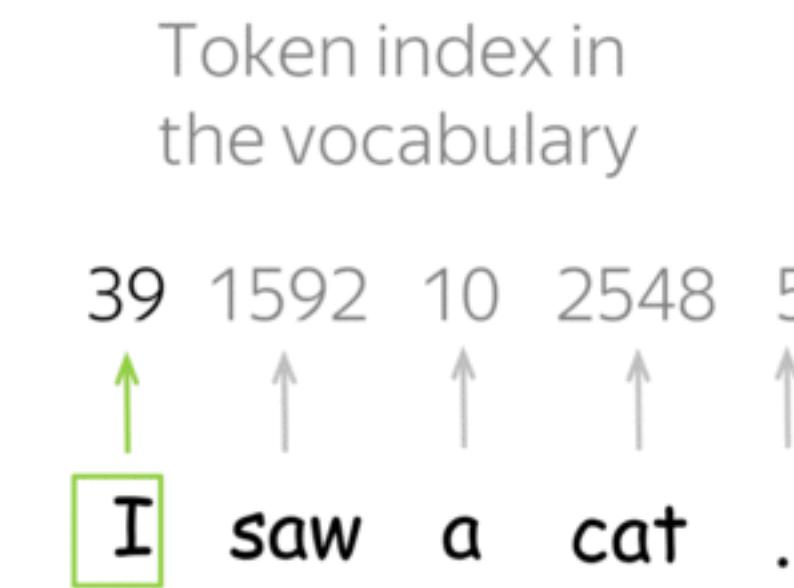


Any algorithm for solving a task

Word representation - vector  
(input for your model/algorithm)

Sequence of tokens

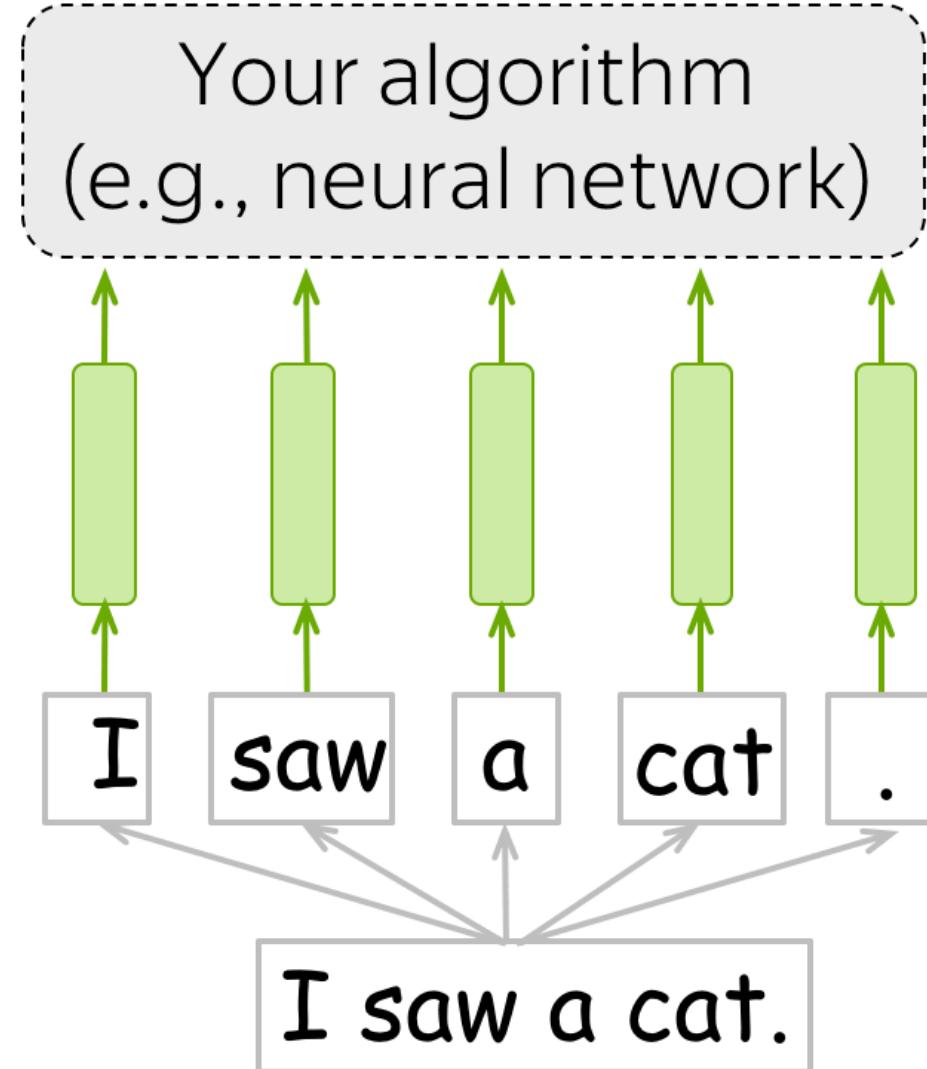
Text (your input)



[Image credit](#)

# Word2Vec

Что делать со словами не из словаря (out-of-vocab)?



Any algorithm for solving a task

Word representation - vector  
(input for your model/algorithm)

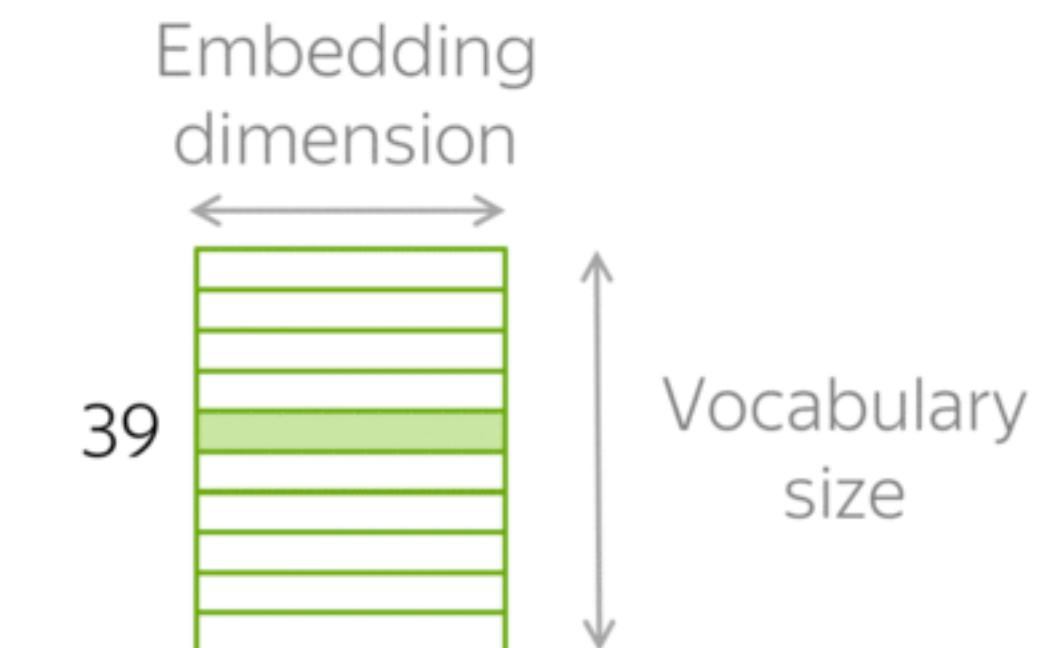
Sequence of tokens

Text (your input)

Token index in  
the vocabulary

39 1592 10 2548 5  
I saw a cat .

I saw a UNK .  
I saw a &%!



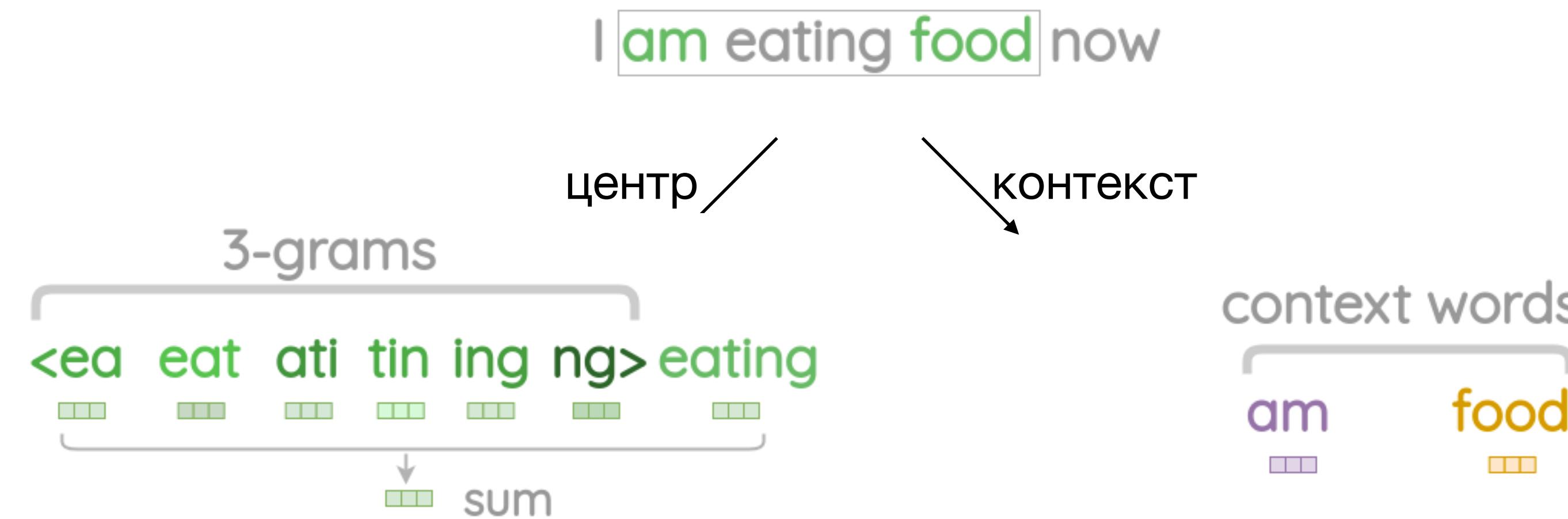
not in the  
vocabulary

Image credit

# FastText

Что делать со словами не из словаря (out-of-vocab)?

Идея: использовать n-grams



[Image credit](#)

# FastText

Что делать со словами не из словаря (out-of-vocab)?

Идея: использовать n-grams

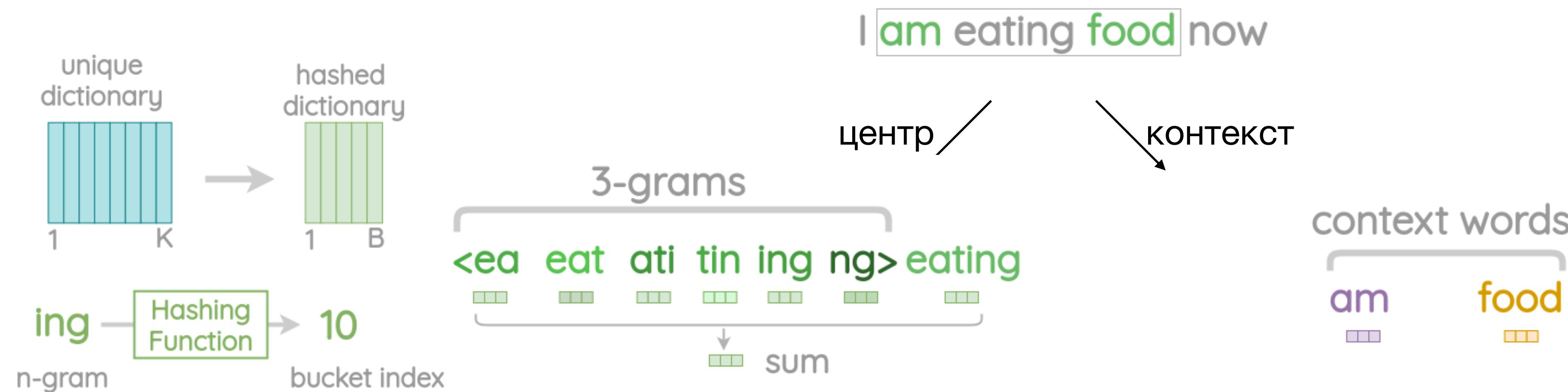


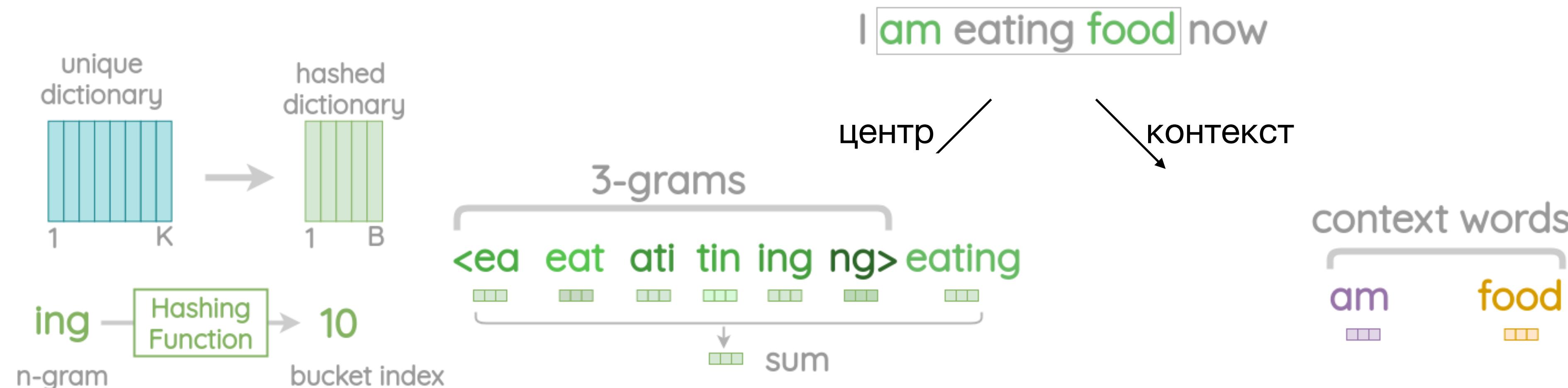
Image credit

# FastText

Что делать со словами не из словаря (out-of-vocab)?

Идея: использовать n-grams

Медленнее обучается, но лучше представления для OOV слов



[Image credit](#)

# Представления текста

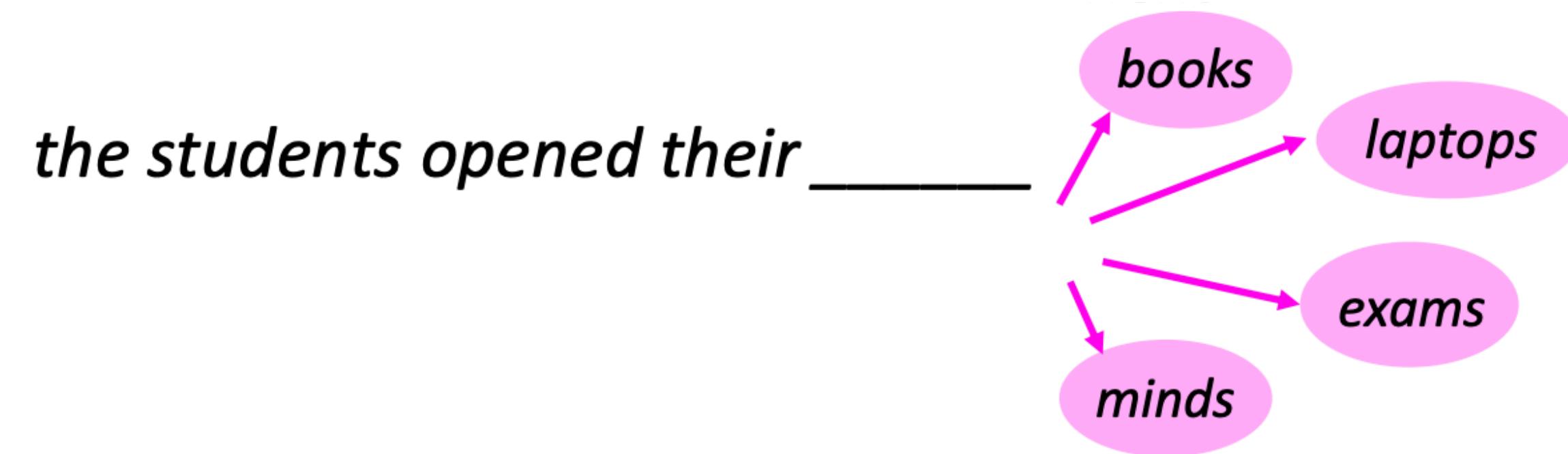
**Word2Vec, FastText, Glove есть предобученные для разных языков!**

Есть имплементации, можно обучить на своем корпусе (gensim и др.)

# **Языковые модели (LM)**

# Языковые модели (Language Models)

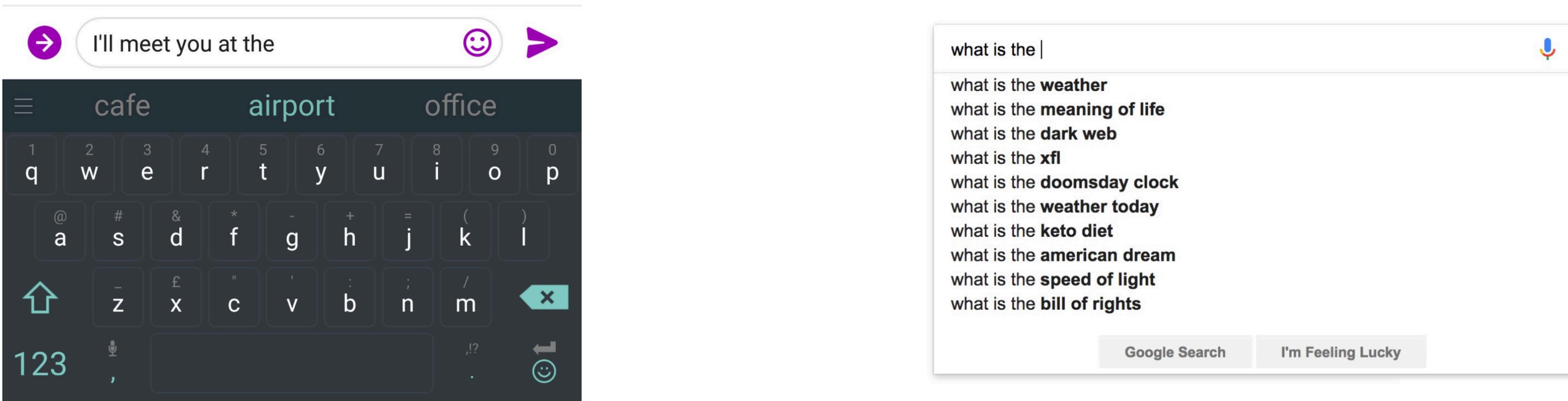
Задача: предсказать следующее слово по предыдущим



Авторегрессионная модель  $p(x) = \prod_{i=1}^n p(x_i | x_1, \dots, x_{i-1})$

# Языковые модели (Language Models)

Задача: предсказать следующее слово по предыдущим



# Языковые модели (Language Models)

Задача: предсказать следующее слово по предыдущим

Используем нейросеть и проход окном

Output distribution

$$\hat{y} = \text{softmax}(Uh + b_2) \in \mathbb{R}^{|V|}$$

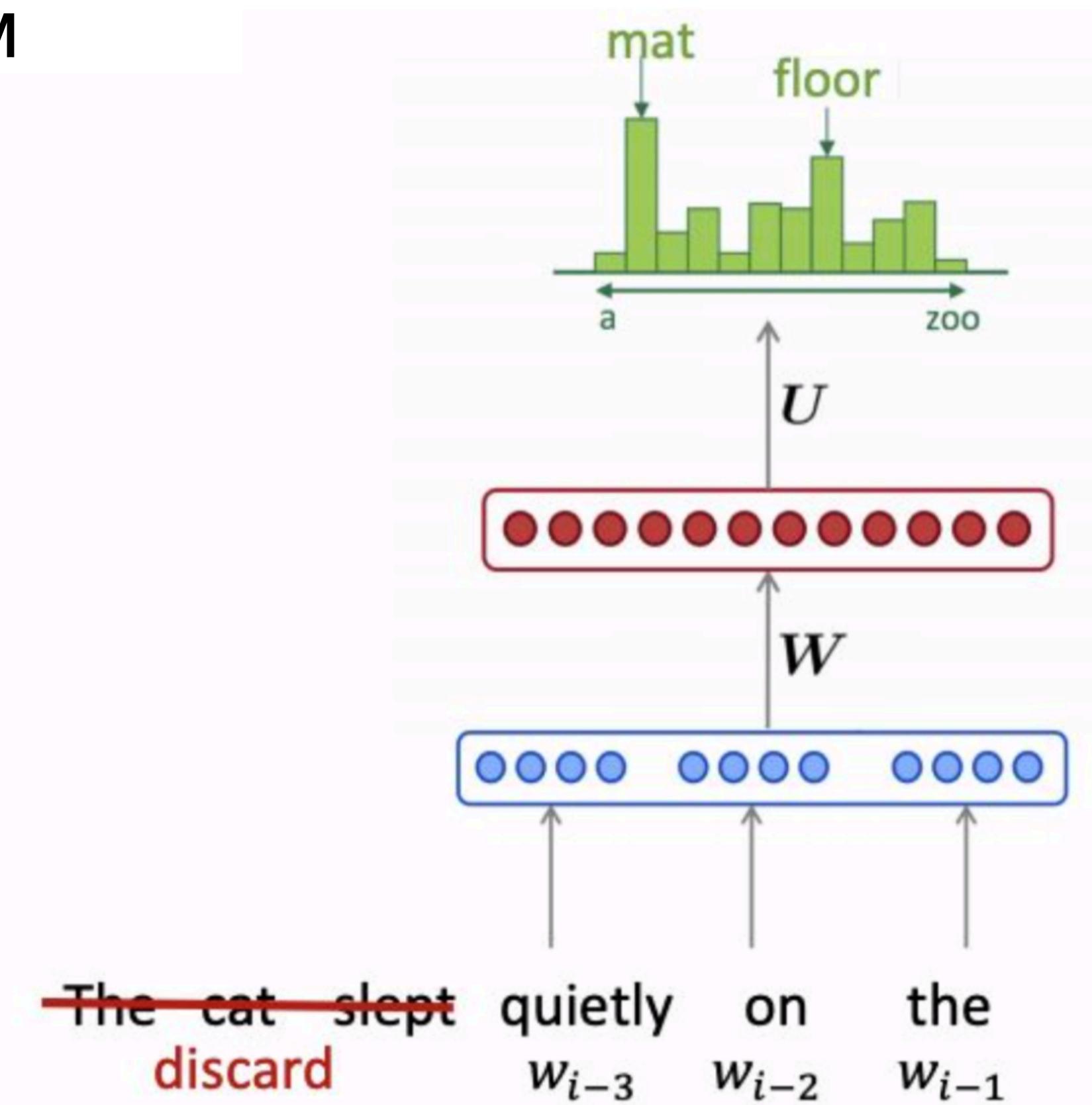
Hidden layer (or any feed-forward NN)

$$h = f(Wx + b)$$

Concatenate word embeddings

$$x = (x_{i-3}, x_{i-2}, x_{i-1})$$

Word embeddings



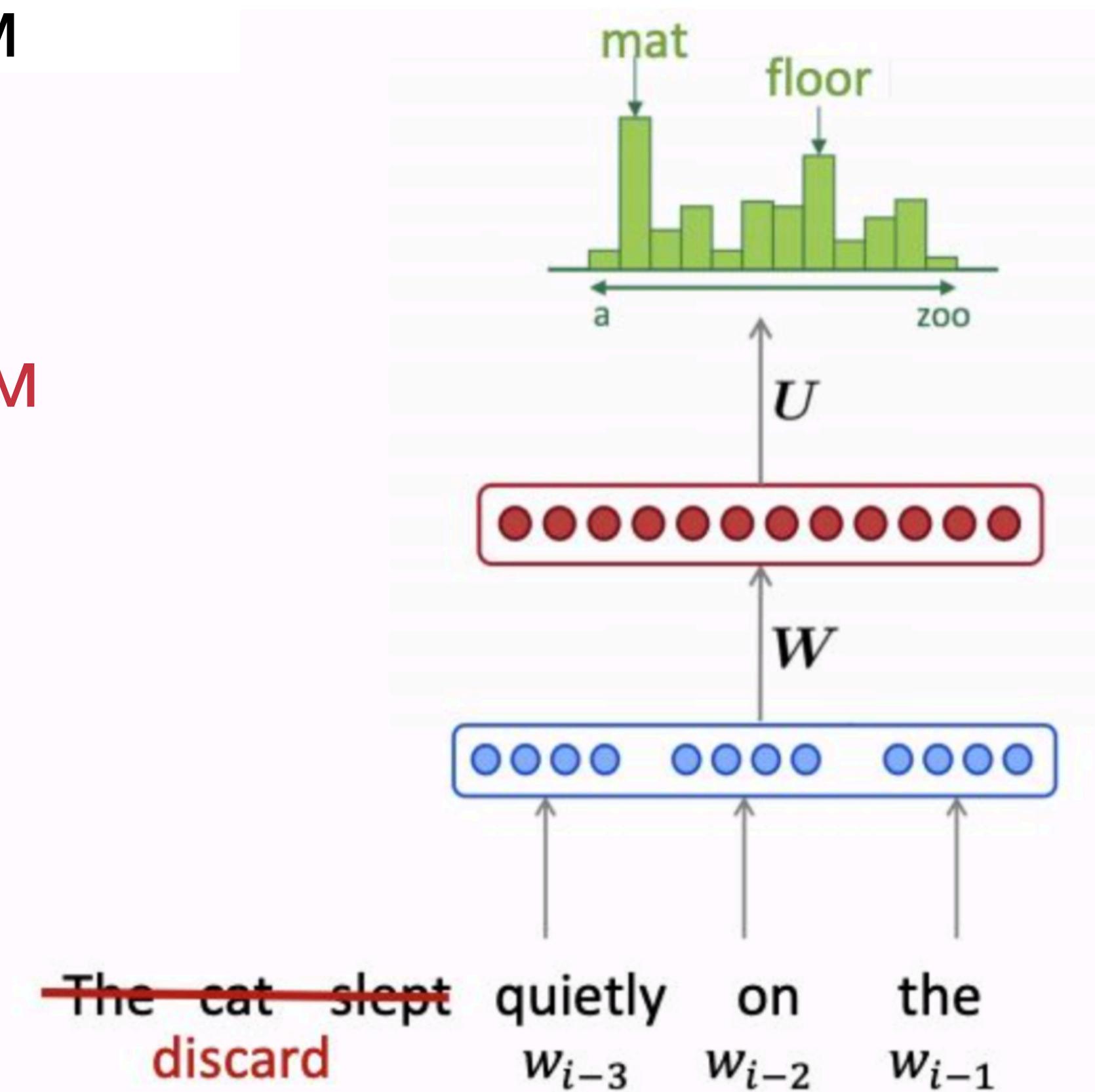
# Языковые модели (Language Models)

Задача: предсказать следующее слово по предыдущим

Используем нейросеть и проход окном

Проблемы:

- Размер окна не может быть большим
- Не получаем весь контекст



# Языковые модели (Language Models)

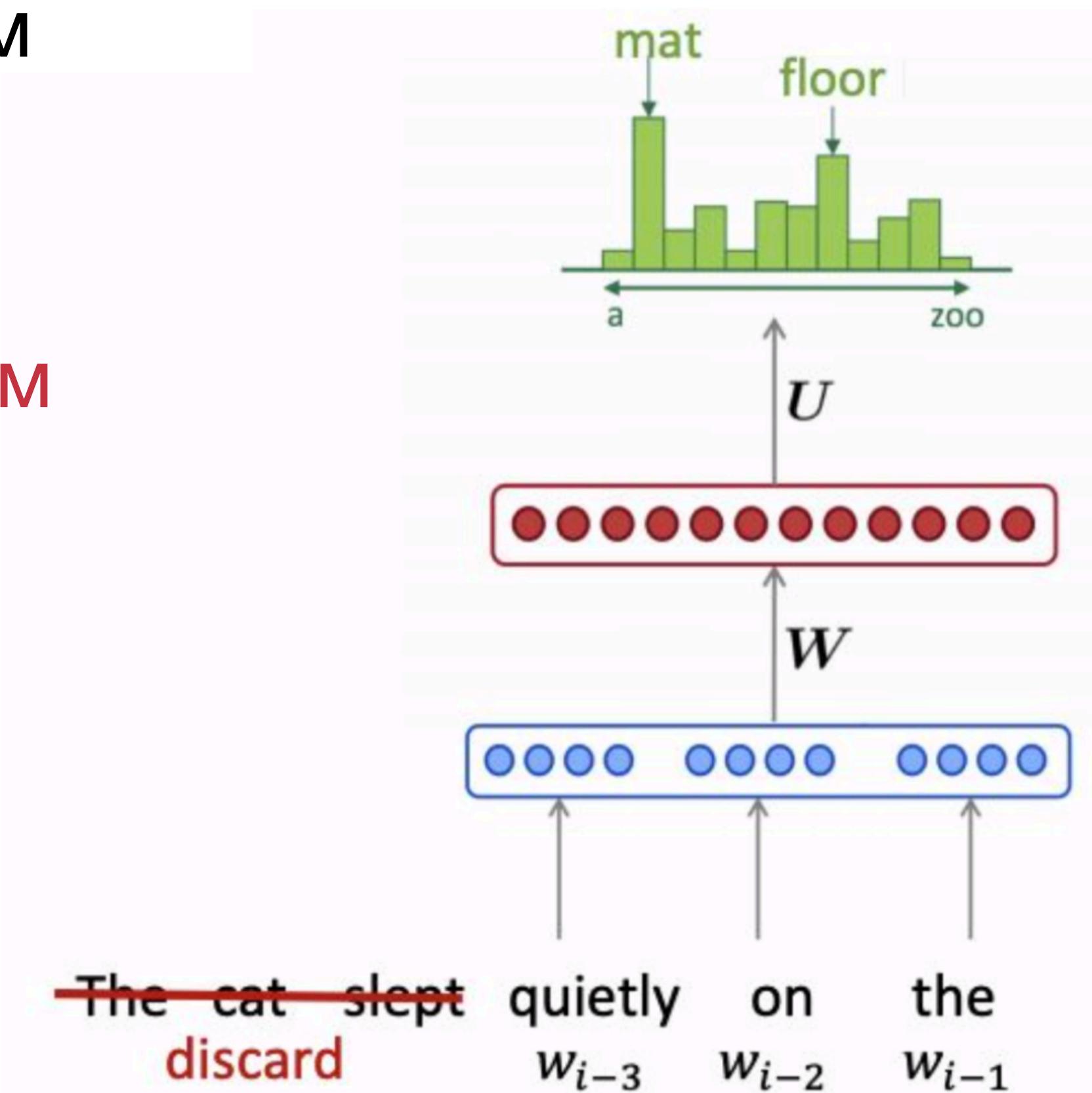
Задача: предсказать следующее слово по предыдущим

Используем нейросеть и проход окном

Проблемы:

- Размер окна не может быть большим
- Не получаем весь контекст

Нужна архитектура, которая может обрабатывать текст любой длины



# Рекуррентные нейросети (RNN)

**output distribution**

$$\hat{y}^{(t)} = \text{softmax}(\mathbf{U}\mathbf{h}^{(t)} + \mathbf{b}_2) \in \mathbb{R}^{|V|}$$

**hidden states**

$$\mathbf{h}^{(t)} = \sigma(\mathbf{W}_h \mathbf{h}^{(t-1)} + \mathbf{W}_e \mathbf{e}^{(t)} + \mathbf{b}_1)$$

$\mathbf{h}^{(0)}$  is the initial hidden state

**word embeddings**

$$\mathbf{e}^{(t)} = \mathbf{E}\mathbf{x}^{(t)}$$

**words / one-hot vectors**

$$\mathbf{x}^{(t)} \in \mathbb{R}^{|V|}$$

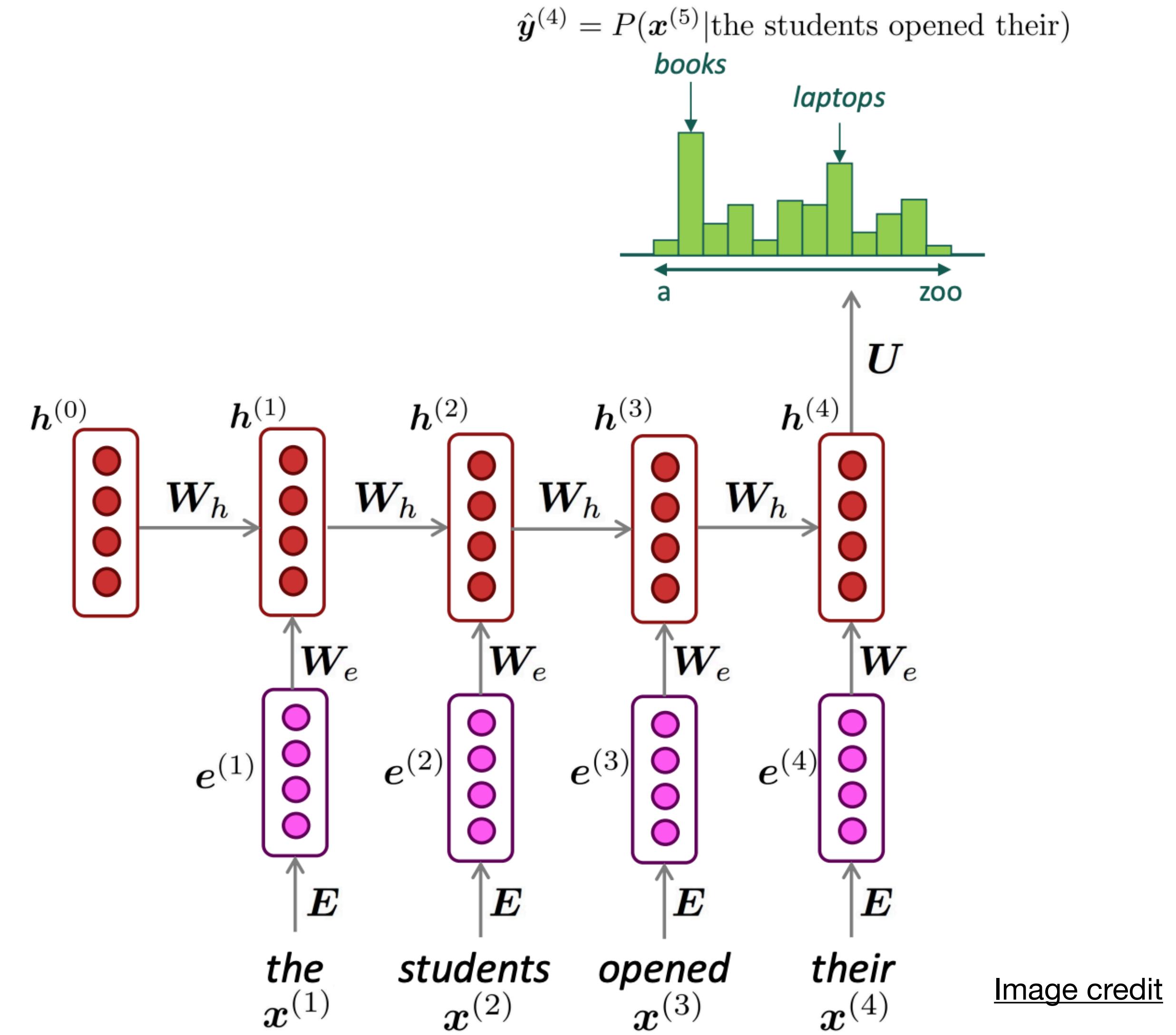


Image credit

# Рекуррентные нейросети (RNN)

В теории получаем весь контекст

output distribution

$$\hat{y}^{(t)} = \text{softmax}(\mathbf{U}\mathbf{h}^{(t)} + \mathbf{b}_2) \in \mathbb{R}^{|V|}$$

hidden states

$$\mathbf{h}^{(t)} = \sigma(\mathbf{W}_h \mathbf{h}^{(t-1)} + \mathbf{W}_e \mathbf{e}^{(t)} + \mathbf{b}_1)$$

$\mathbf{h}^{(0)}$  is the initial hidden state

word embeddings

$$\mathbf{e}^{(t)} = \mathbf{E}\mathbf{x}^{(t)}$$

words / one-hot vectors

$$\mathbf{x}^{(t)} \in \mathbb{R}^{|V|}$$

$$\hat{y}^{(4)} = P(\mathbf{x}^{(5)} | \text{the students opened their})$$

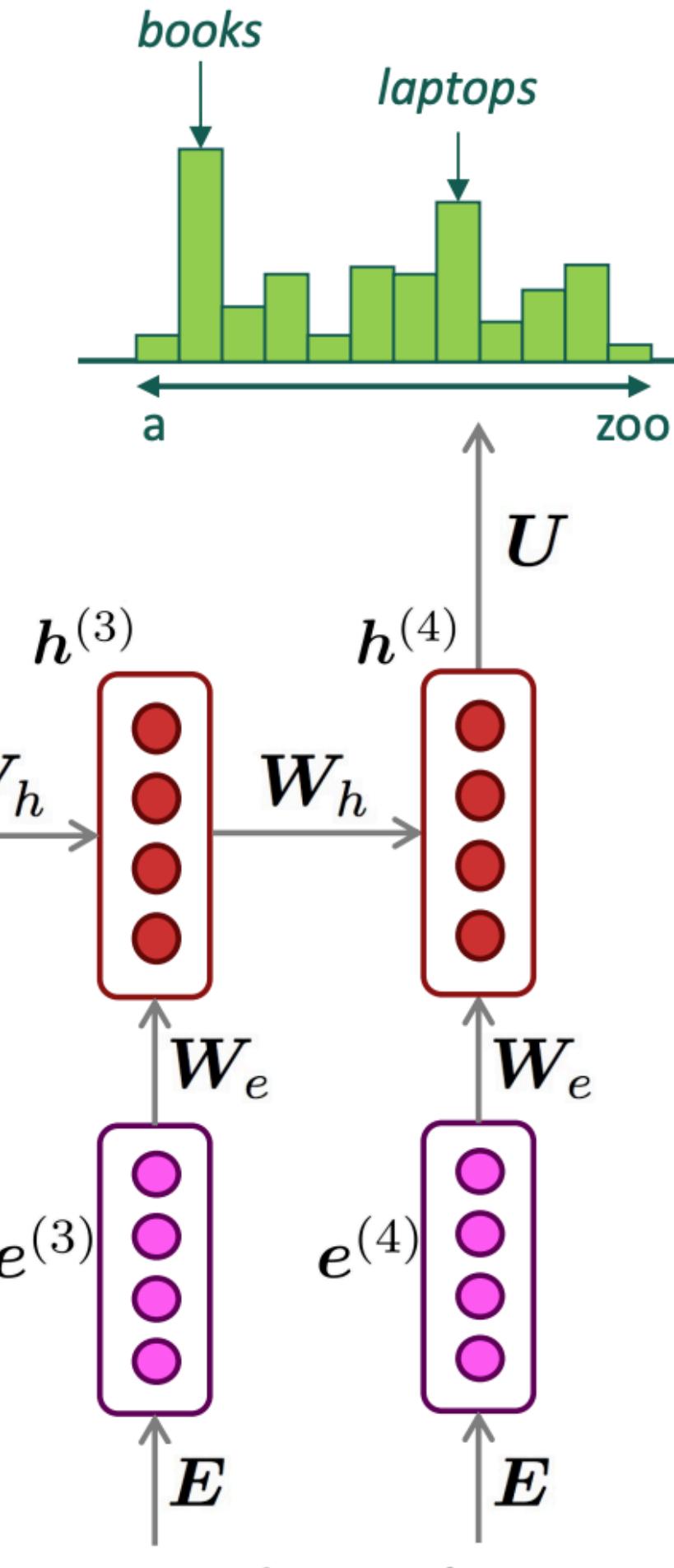


Image credit

# Рекуррентные нейросети (RNN)

## Обучение

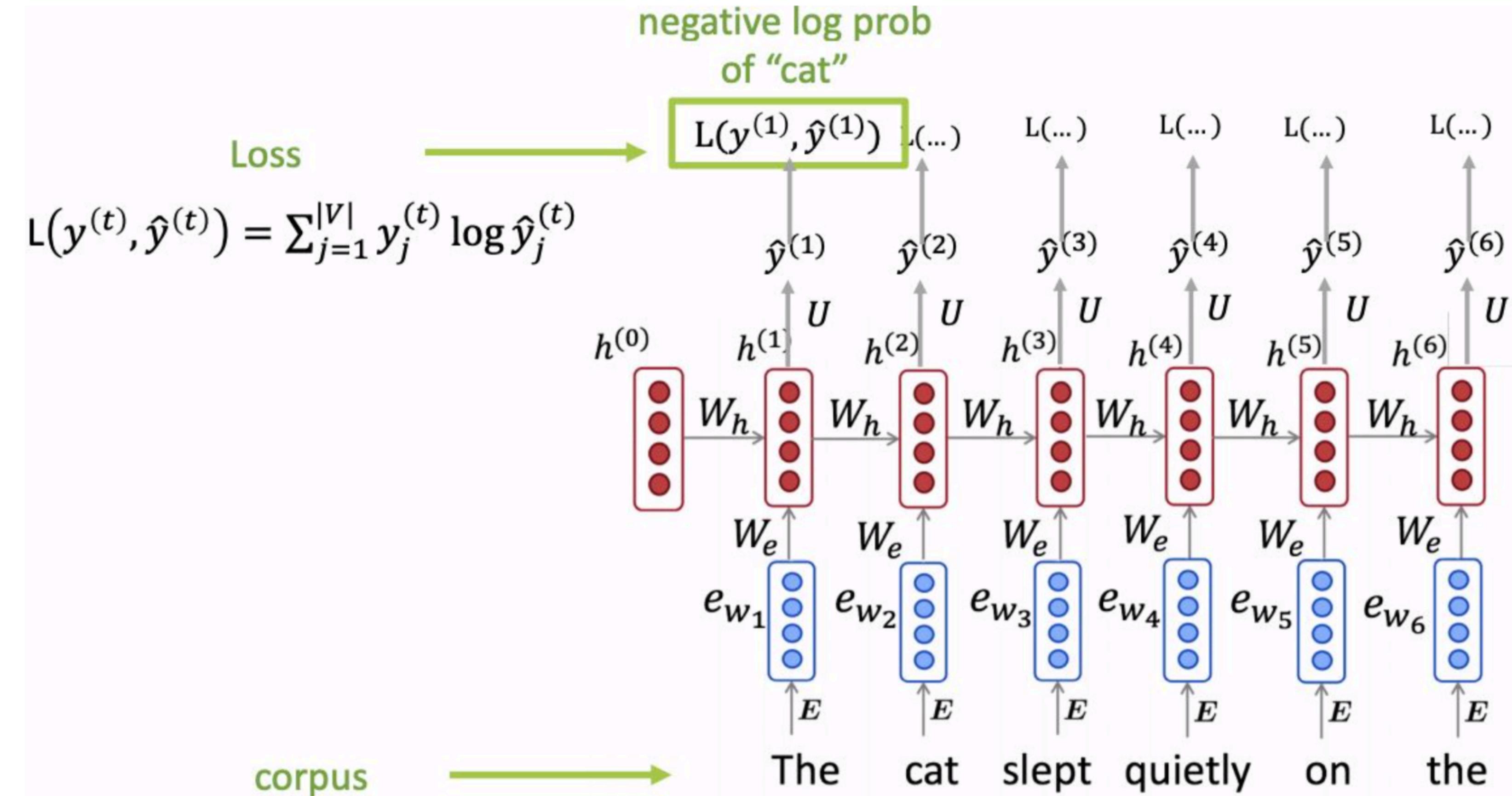


Image credit

# Рекуррентные нейросети (RNN)

## Обучение

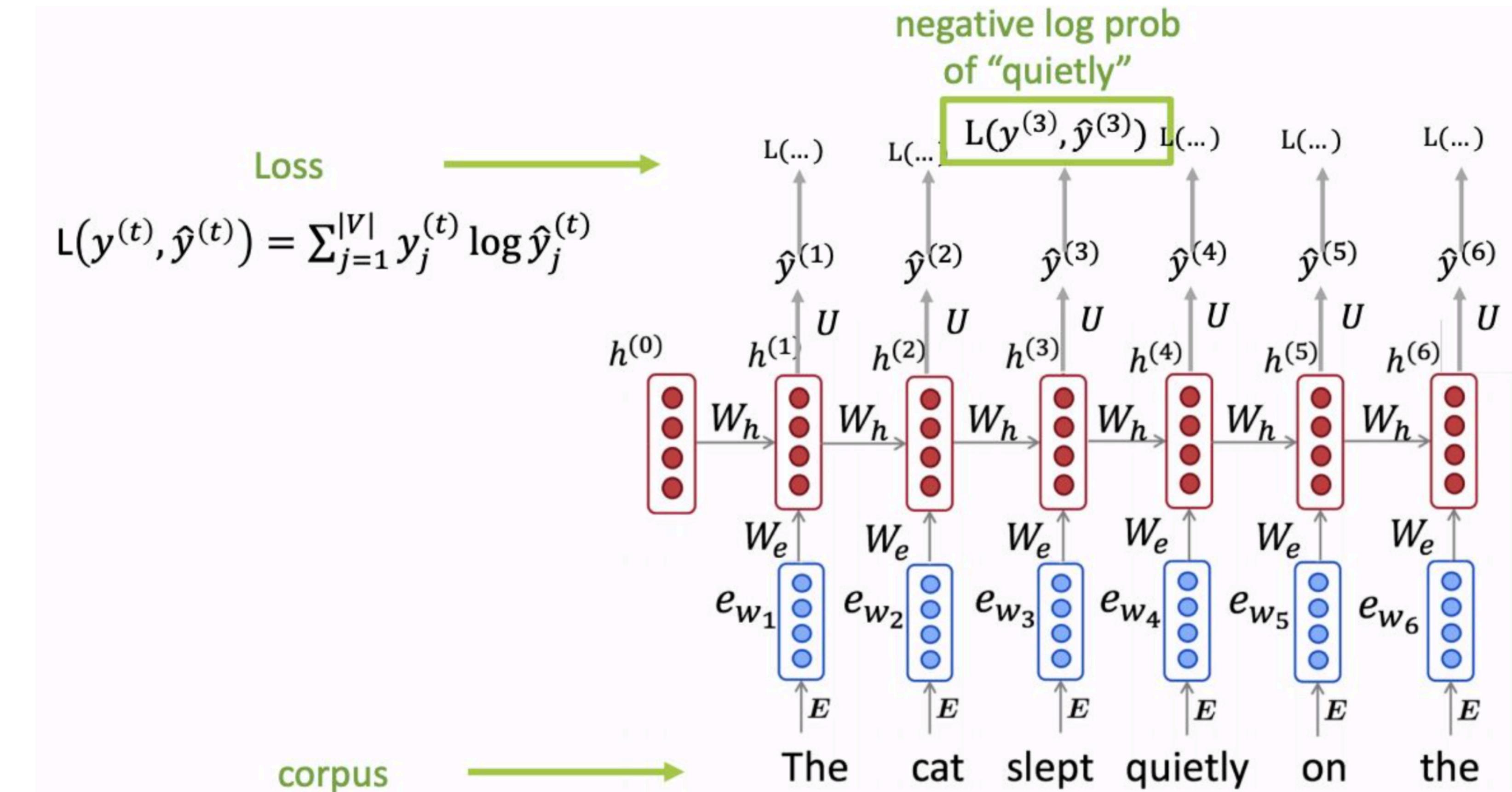


Image credit

# Рекуррентные нейросети (RNN)

## Обучение

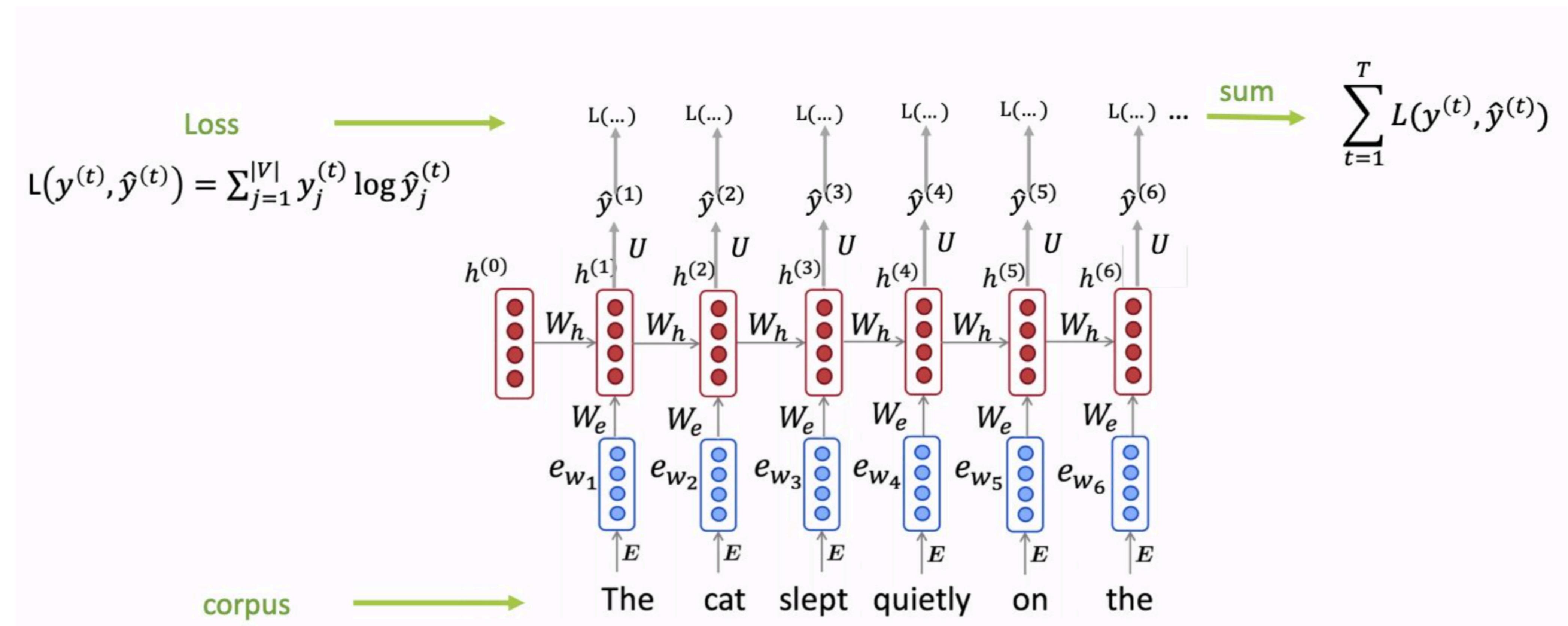
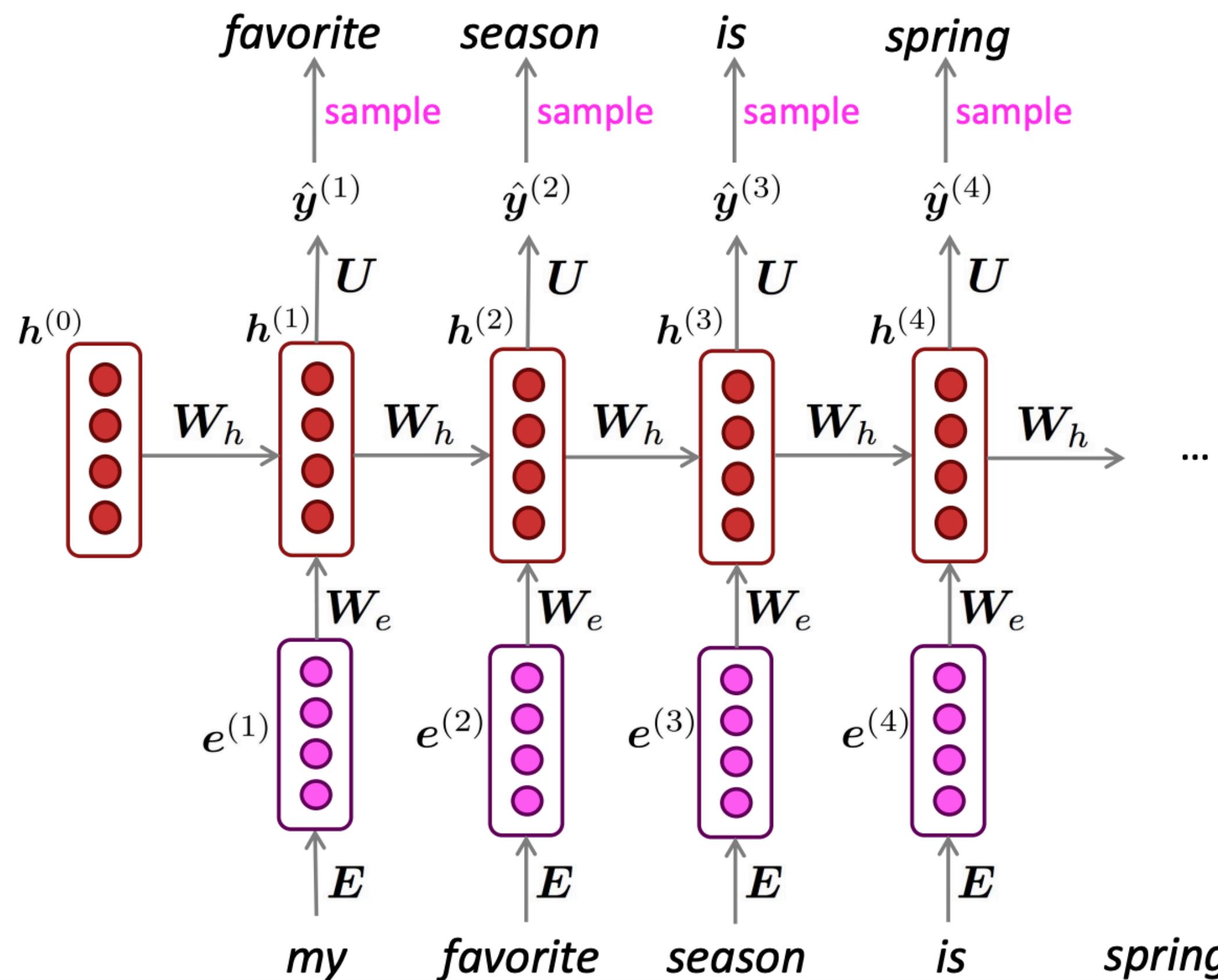


Image credit

# Рекуррентные нейросети (RNN)

## Генерация текста (language modeling)



“Sorry,” Harry shouted, panicking—“I’ll leave those brooms in London, are they?”

“No idea,” said Nearly Headless Nick, casting low close by Cedric, carrying the last bit of treacle Charms, from Harry’s shoulder, and to answer him the common room perched upon it, four arms held a shining knob from when the spider hadn’t felt it seemed. He reached the teams too.

Image credit

# Рекуррентные нейросети (RNN)

Можно использовать для других задач!

output distribution

$$\hat{y}^{(t)} = \text{softmax}(\mathbf{U}\mathbf{h}^{(t)} + \mathbf{b}_2) \in \mathbb{R}^{|V|}$$

hidden states

$$\mathbf{h}^{(t)} = \sigma(\mathbf{W}_h \mathbf{h}^{(t-1)} + \mathbf{W}_e \mathbf{e}^{(t)} + \mathbf{b}_1)$$

$\mathbf{h}^{(0)}$  is the initial hidden state

word embeddings

$$\mathbf{e}^{(t)} = \mathbf{E}\mathbf{x}^{(t)}$$

words / one-hot vectors

$$\mathbf{x}^{(t)} \in \mathbb{R}^{|V|}$$

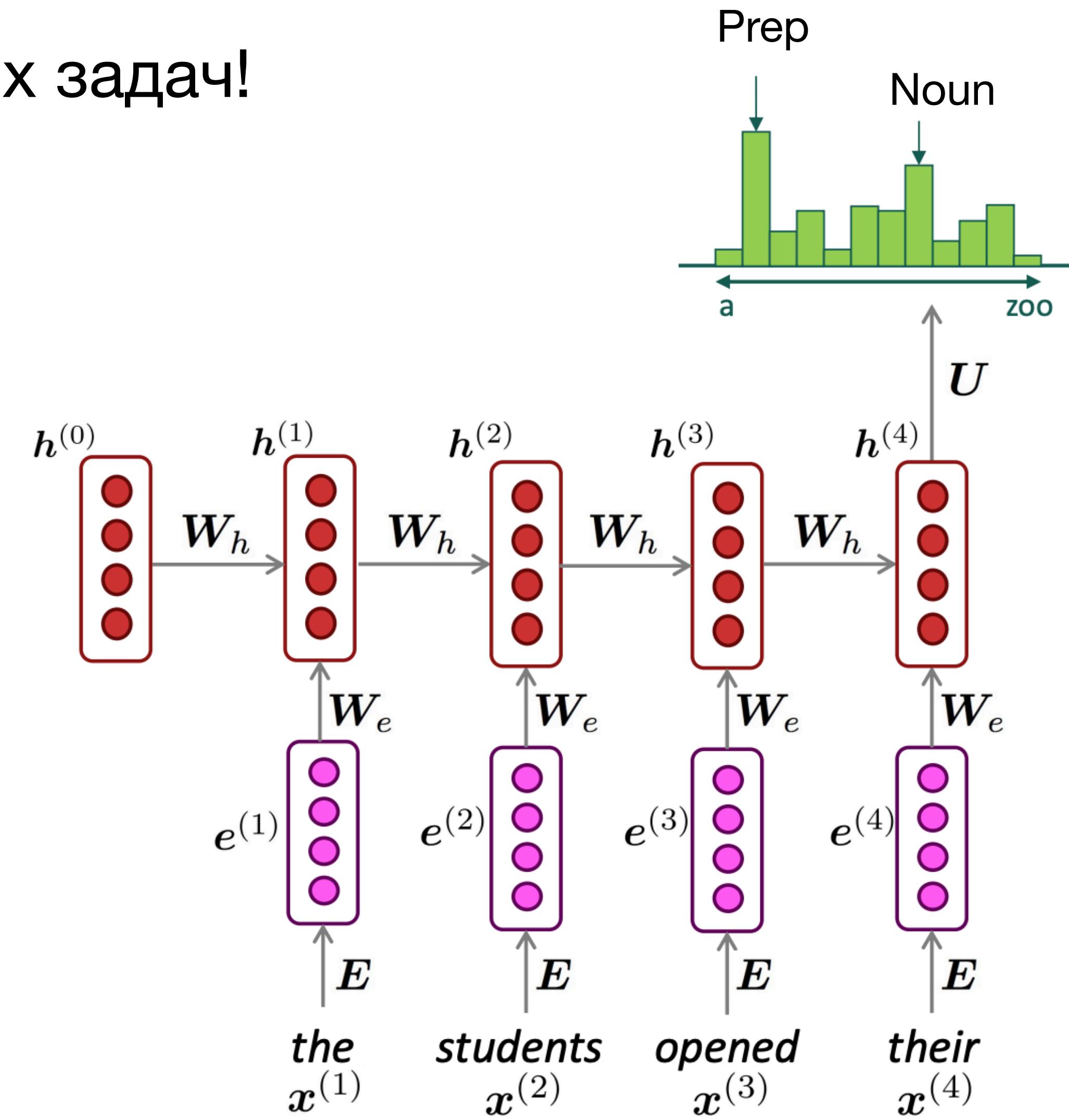
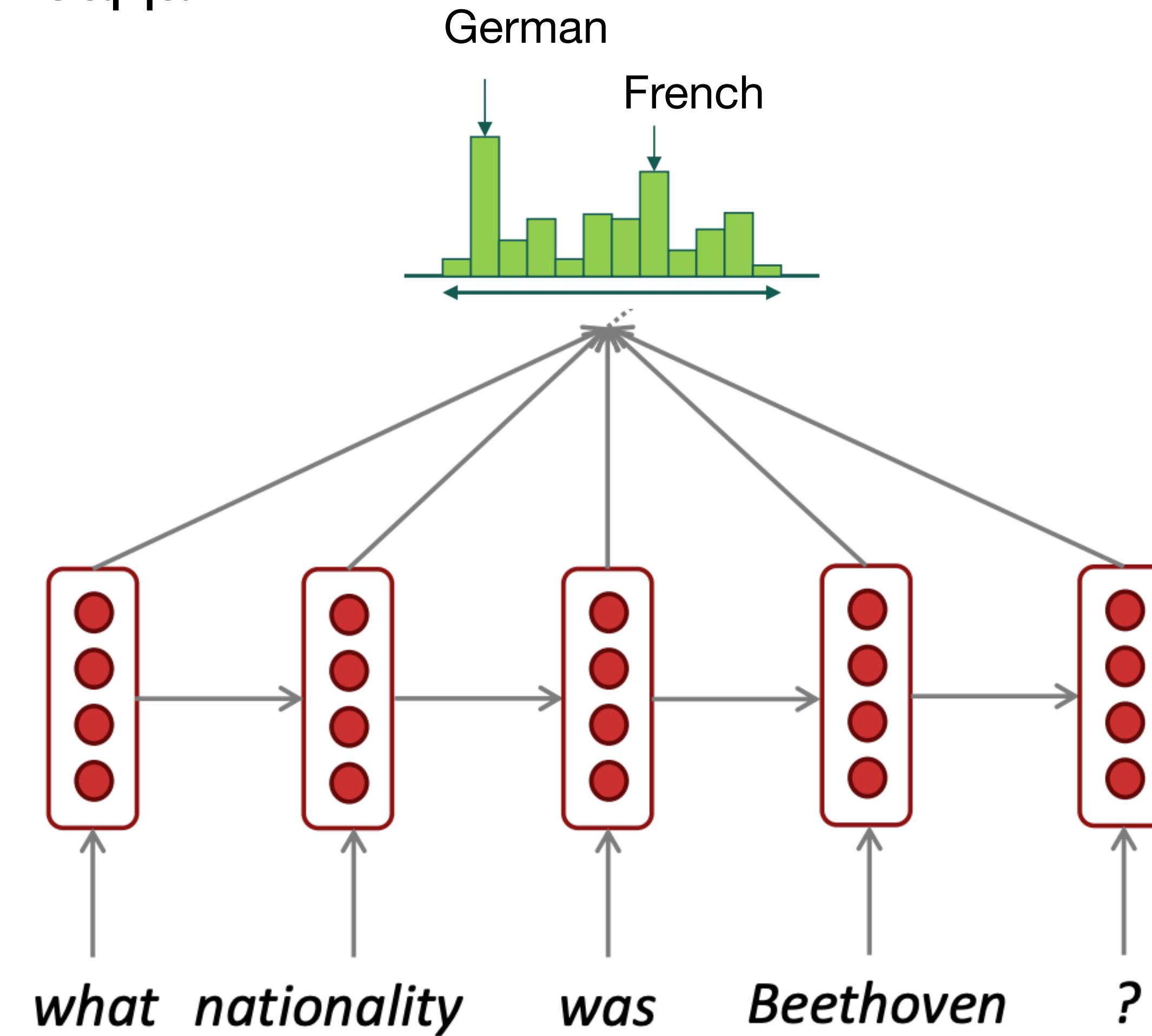


Image credit

# Рекуррентные нейросети (RNN)

Можно использовать для других задач!

Sentence classification

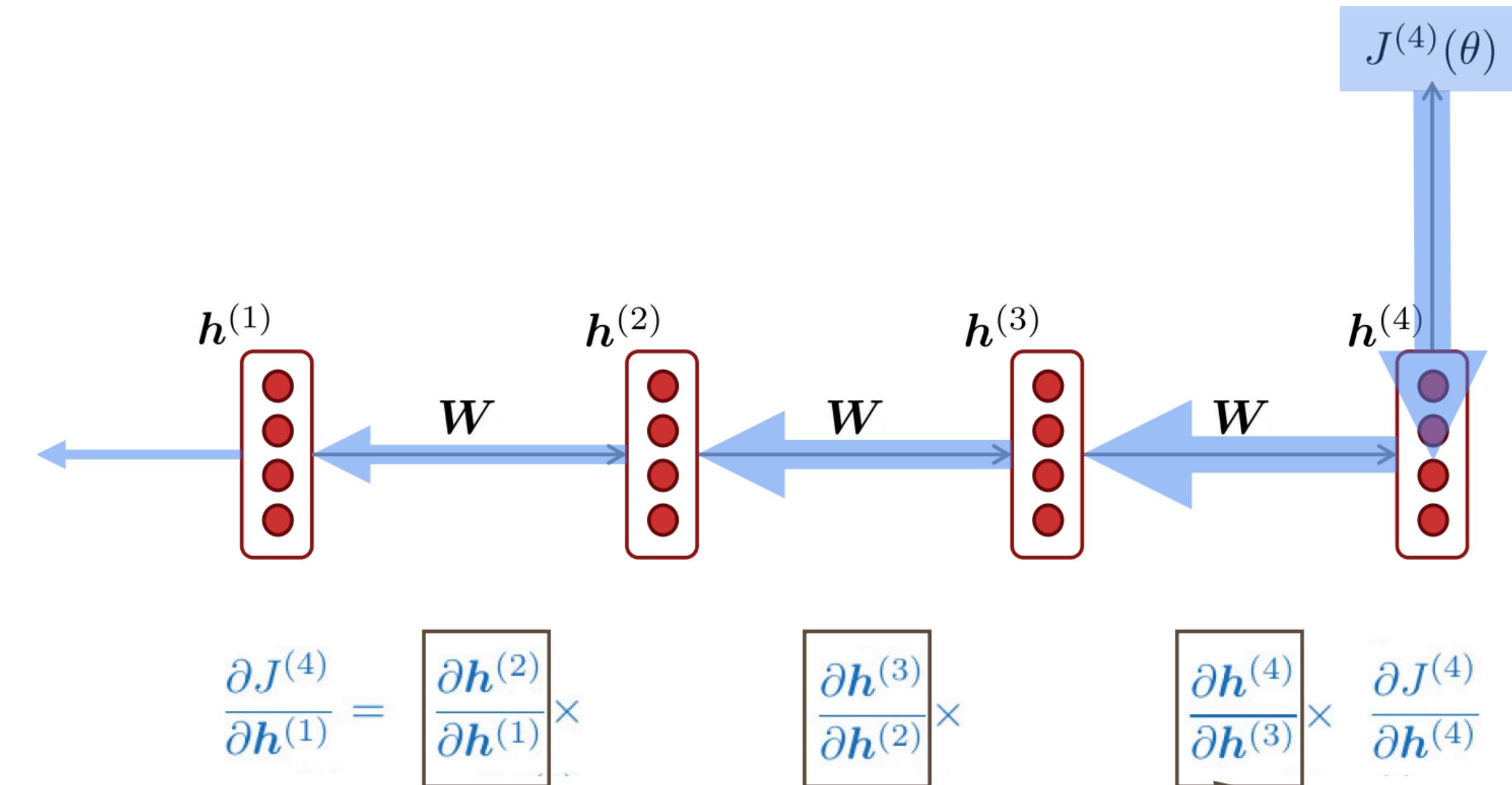


[Image credit](#)

# Рекуррентные нейросети (RNN)

Проблемы?

- Медленно
- Vanishing gradients



**LM task:** When she tried to print her tickets, she found that the printer was out of toner.  
She went to the stationery store to buy more toner. It was very overpriced. After  
installing the toner into the printer, she finally printed her \_\_\_\_\_

Image credit

# Рекуррентные нейросети (RNN)

## Проблемы?

- Медленно
- Vanishing gradients
- Exploding gradients? Solution: gradients clipping

---

### Algorithm 1 Pseudo-code for norm clipping

---

```
 $\hat{\mathbf{g}} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}$ 
if  $\|\hat{\mathbf{g}}\| \geq threshold$  then
     $\hat{\mathbf{g}} \leftarrow \frac{threshold}{\|\hat{\mathbf{g}}\|} \hat{\mathbf{g}}$ 
end if
```

---

# LSTM

Vanilla RNN постоянно переписывает  $\mathbf{h}^{(t)} = \sigma(\mathbf{W}_h \mathbf{h}^{(t-1)} + \mathbf{W}_x \mathbf{x}^{(t)} + \mathbf{b})$

Идея: добавить “память”  $\mathbf{c}^{(t)}$

Новая информация

$$\tilde{\mathbf{c}}^{(t)} = \tanh(\mathbf{W}_c \mathbf{h}^{(t-1)} + \mathbf{U}_c \mathbf{x}^{(t)} + \mathbf{b}_c)$$

$$\mathbf{c}^{(t)} = \mathbf{f}^{(t)} \circ \mathbf{c}^{(t-1)} + \mathbf{i}^{(t)} \circ \tilde{\mathbf{c}}^{(t)}$$

$$\mathbf{h}^{(t)} = \mathbf{o}^{(t)} \circ \tanh \mathbf{c}^{(t)}$$

All these are vectors of same length  $n$

# LSTM

Vanilla RNN постоянно переписывает  $\mathbf{h}^{(t)} = \sigma(\mathbf{W}_h \mathbf{h}^{(t-1)} + \mathbf{W}_x \mathbf{x}^{(t)} + \mathbf{b})$

Идея: добавить “память”  $\mathbf{c}^{(t)}$

Forget gate: контролируем сколько забыть

Input gate: контролируем сколько записать

$$f^{(t)} = \sigma(\mathbf{W}_f \mathbf{h}^{(t-1)} + \mathbf{U}_f \mathbf{x}^{(t)} + \mathbf{b}_f)$$
$$i^{(t)} = \sigma(\mathbf{W}_i \mathbf{h}^{(t-1)} + \mathbf{U}_i \mathbf{x}^{(t)} + \mathbf{b}_i)$$

Sigmoid function: all gate values are between 0 and 1

Новая информация

Можно записать новую и стереть старую

$$\tilde{\mathbf{c}}^{(t)} = \tanh(\mathbf{W}_c \mathbf{h}^{(t-1)} + \mathbf{U}_c \mathbf{x}^{(t)} + \mathbf{b}_c)$$
$$\mathbf{c}^{(t)} = f^{(t)} \circ \mathbf{c}^{(t-1)} + i^{(t)} \circ \tilde{\mathbf{c}}^{(t)}$$
$$\mathbf{h}^{(t)} = o^{(t)} \circ \tanh \mathbf{c}^{(t)}$$

All these are vectors of same length  $n$

# LSTM

Vanilla RNN постоянно переписывает  $\mathbf{h}^{(t)} = \sigma(\mathbf{W}_h \mathbf{h}^{(t-1)} + \mathbf{W}_x \mathbf{x}^{(t)} + \mathbf{b})$

Идея: добавить “память”  $\mathbf{c}^{(t)}$

Forget gate: контролируем сколько забыть

Input gate: контролируем сколько записать

Output gate: контролируем сколько считать

Sigmoid function: all gate values are between 0 and 1

$$f^{(t)} = \sigma(\mathbf{W}_f \mathbf{h}^{(t-1)} + \mathbf{U}_f \mathbf{x}^{(t)} + \mathbf{b}_f)$$

$$i^{(t)} = \sigma(\mathbf{W}_i \mathbf{h}^{(t-1)} + \mathbf{U}_i \mathbf{x}^{(t)} + \mathbf{b}_i)$$

$$o^{(t)} = \sigma(\mathbf{W}_o \mathbf{h}^{(t-1)} + \mathbf{U}_o \mathbf{x}^{(t)} + \mathbf{b}_o)$$

Новая информация

Можно записать новую и стереть старую

Считывание

$$\tilde{\mathbf{c}}^{(t)} = \tanh(\mathbf{W}_c \mathbf{h}^{(t-1)} + \mathbf{U}_c \mathbf{x}^{(t)} + \mathbf{b}_c)$$

$$\mathbf{c}^{(t)} = f^{(t)} \circ \mathbf{c}^{(t-1)} + i^{(t)} \circ \tilde{\mathbf{c}}^{(t)}$$

$$\mathbf{h}^{(t)} = o^{(t)} \circ \tanh \mathbf{c}^{(t)}$$

All these are vectors of same length  $n$

# GRU

Упрощенная версия LSTM

Update gate

$$\mathbf{u}^{(t)} = \sigma \left( \mathbf{W}_u \mathbf{h}^{(t-1)} + \mathbf{U}_u \mathbf{x}^{(t)} + \mathbf{b}_u \right)$$

Reset gate

$$\mathbf{r}^{(t)} = \sigma \left( \mathbf{W}_r \mathbf{h}^{(t-1)} + \mathbf{U}_r \mathbf{x}^{(t)} + \mathbf{b}_r \right)$$

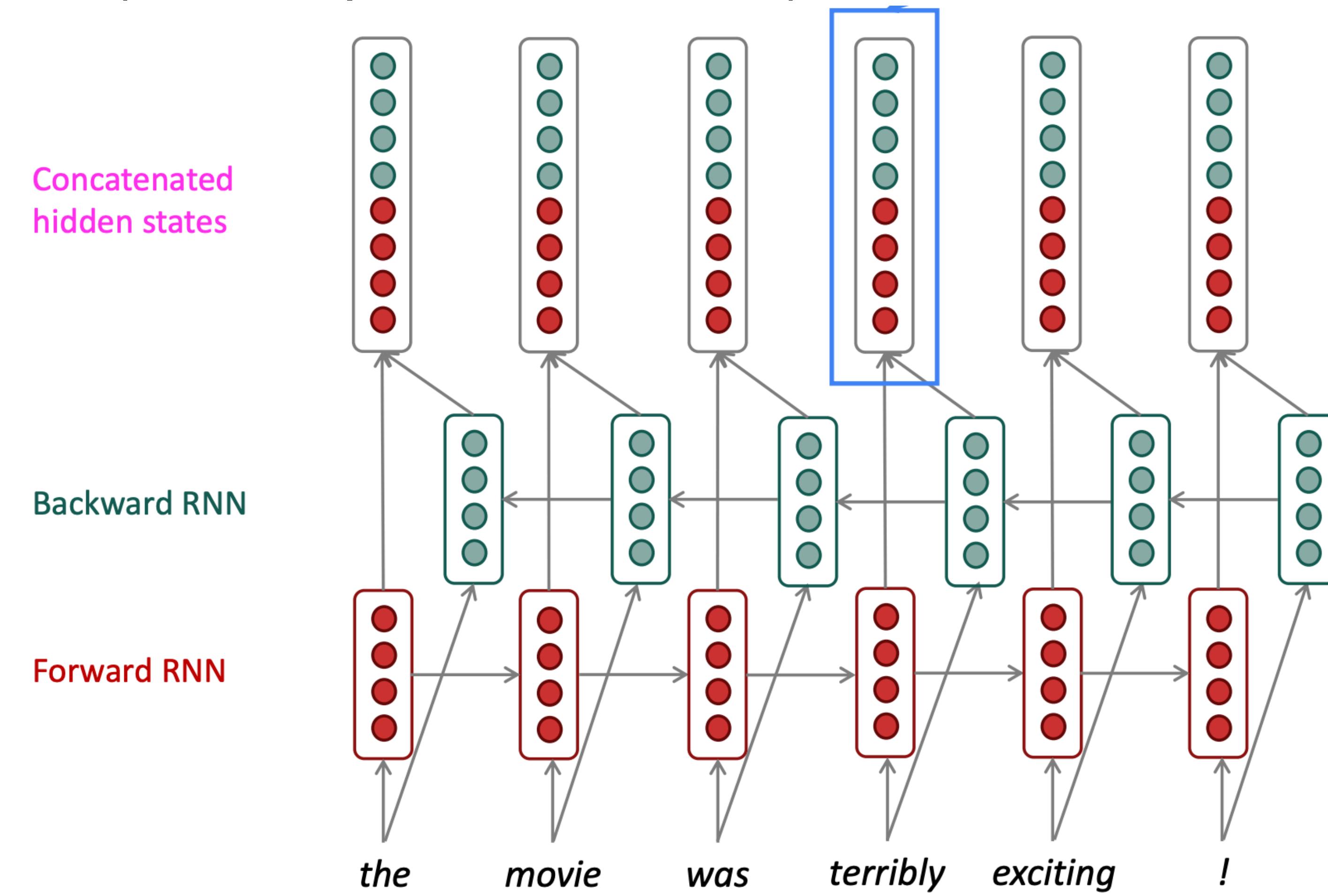
$$\begin{aligned}\tilde{\mathbf{h}}^{(t)} &= \tanh \left( \mathbf{W}_h (\mathbf{r}^{(t)} \circ \mathbf{h}^{(t-1)}) + \mathbf{U}_h \mathbf{x}^{(t)} + \mathbf{b}_h \right) \\ \mathbf{h}^{(t)} &= (1 - \mathbf{u}^{(t)}) \circ \mathbf{h}^{(t-1)} + \mathbf{u}^{(t)} \circ \tilde{\mathbf{h}}^{(t)}\end{aligned}$$

# Рекуррентные нейросети (RNN)

**GRU** и **LSTM** - уменьшают проблему vanishing gradients (необязательно обновлять каждый раз информацию) - использовать их!

# Рекуррентные нейросети (RNN)

Улучшения: bidirectional RNN (если порядок не важен)



# Рекуррентные нейросети (RNN)

Улучшения: stacked RNN

