

# Автоматическая обработка текстов

## Нейронные сети

Лекция 6

Емельянов А. А.  
login-const@mail.ru

# Биологическая модель нейрона

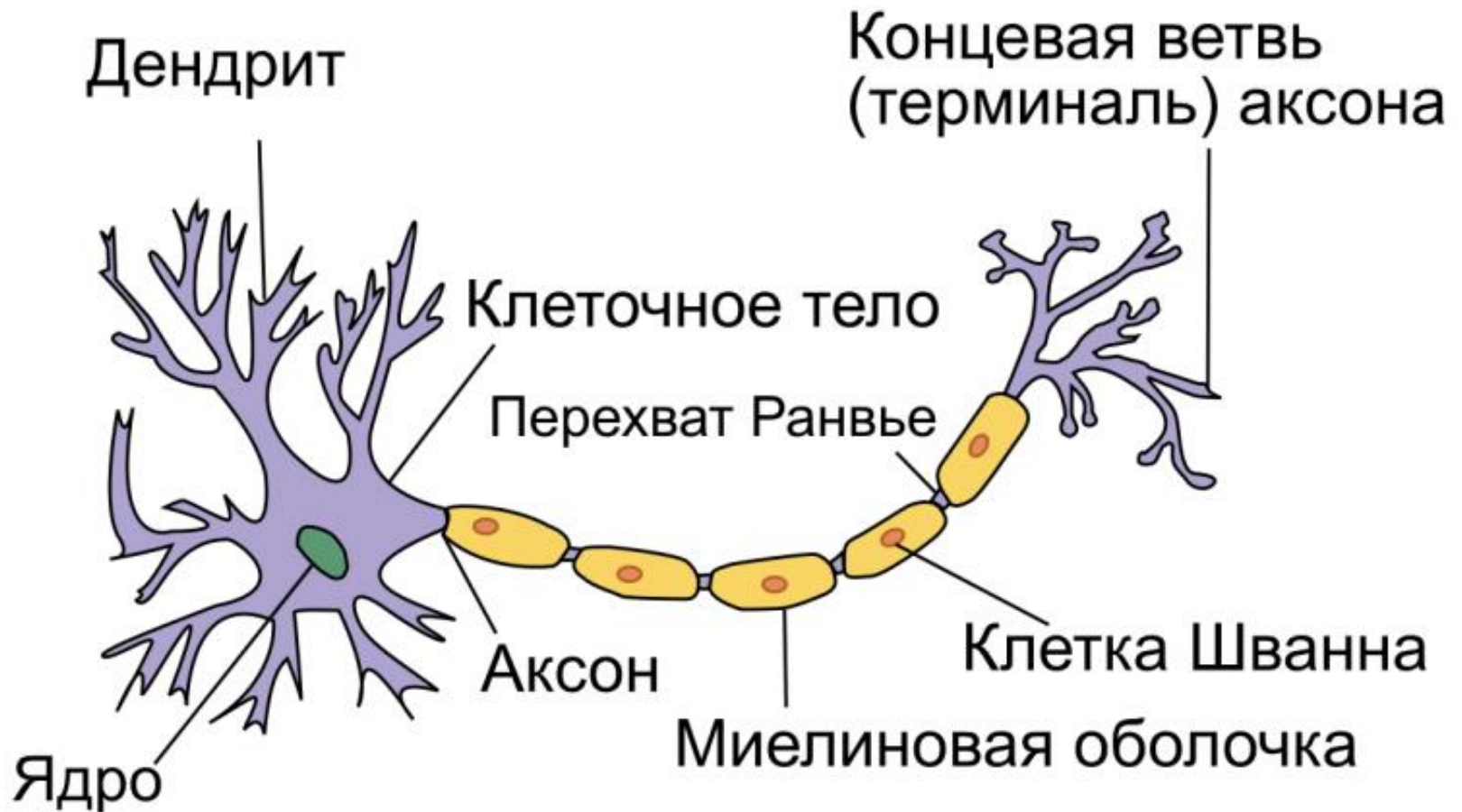


Рис.: Структура нейрона

# Биологическая модель нейрона

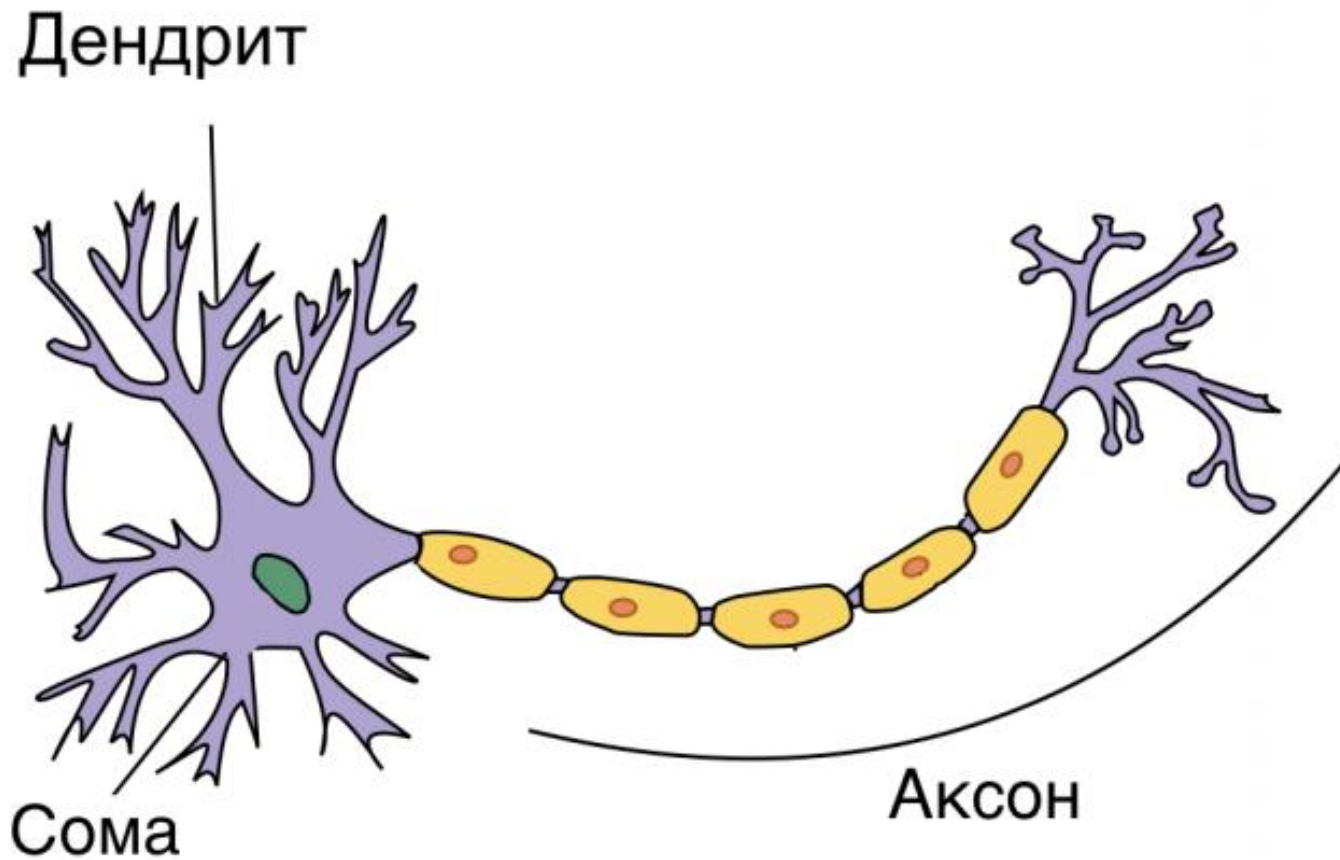


Рис.: Структура нейрона

# Биологическая модель нейрона

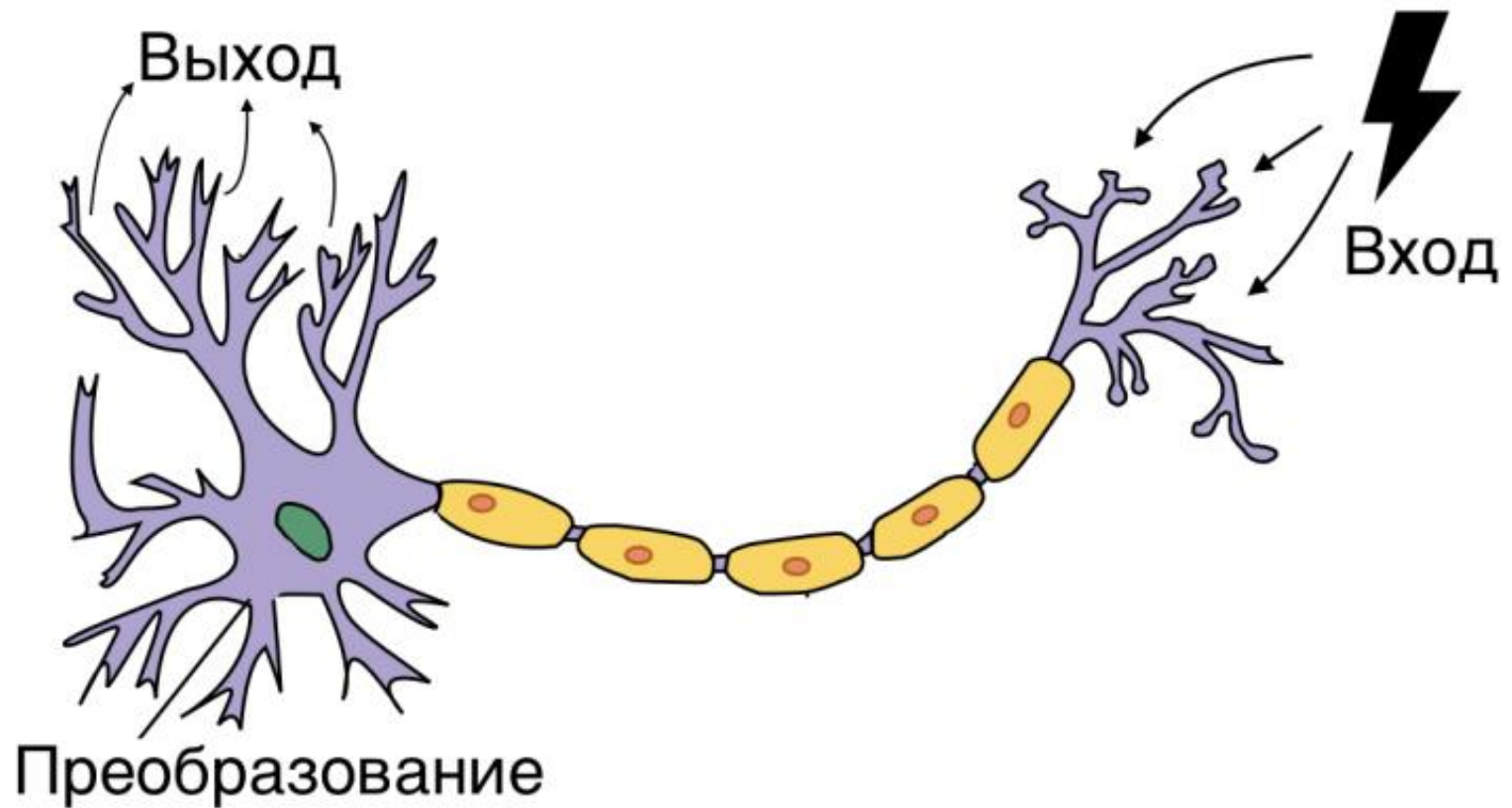


Рис.: Структура нейрона

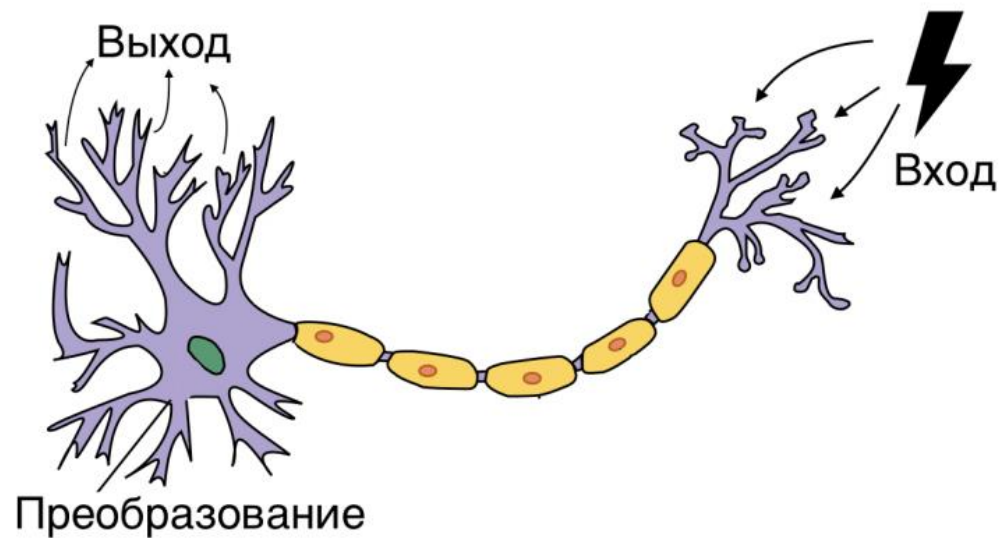
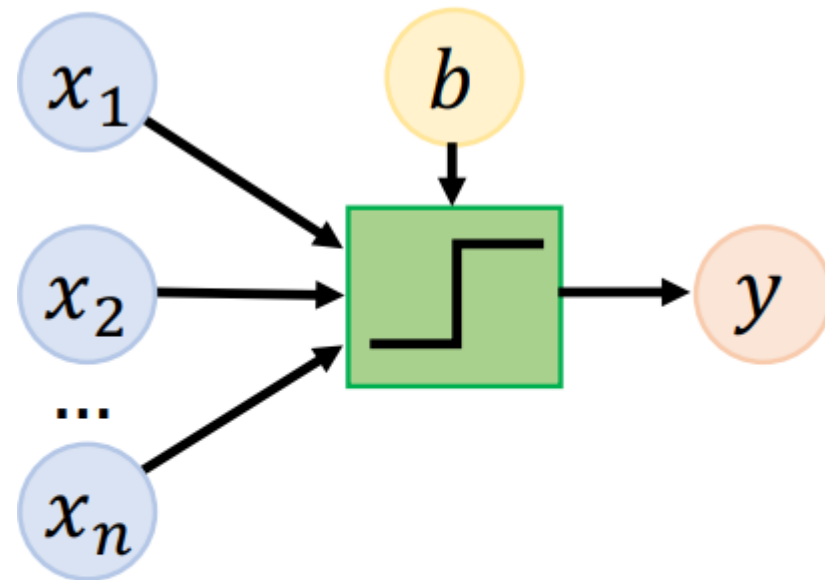


Рис.: Структура нейрона

- Выходной сигнал посылается при достижении определенного уровня входного сигнала. Модель:

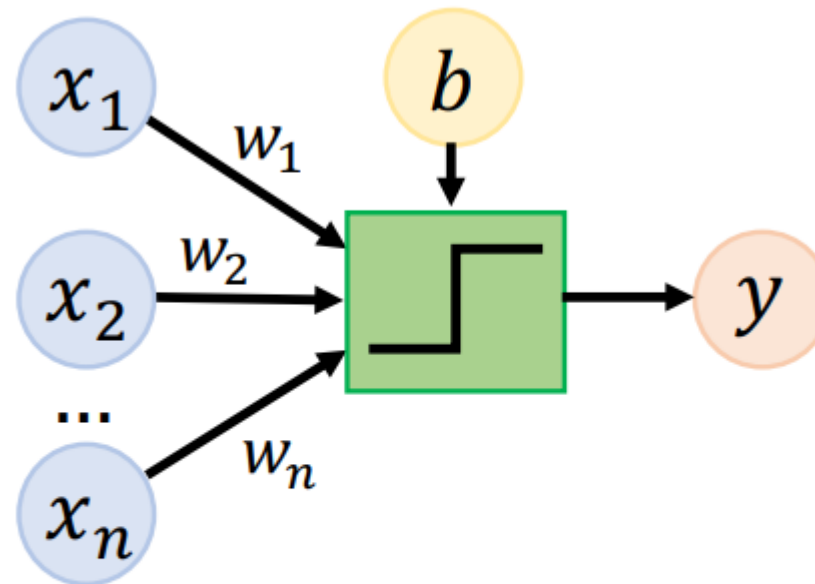
$$y = \begin{cases} 1, & \sum_{i=1}^N x_i > b \\ 0, & \text{иначе} \end{cases} = \mathbb{I}\left[\sum_{i=1}^N x_i > b\right]$$

# Модель 1: схема



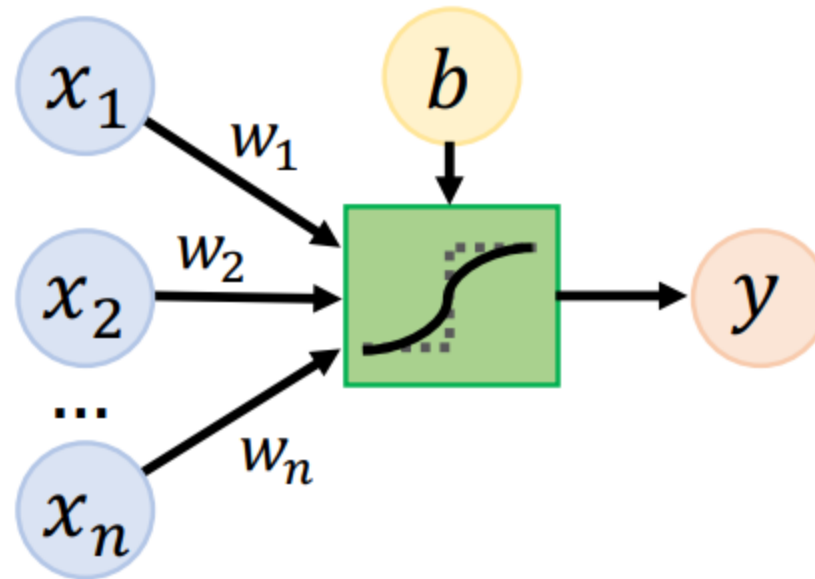
$$y = \mathbb{I}\left[\sum_{i=1}^N x_i > b\right]$$

## Модель 2: Чувствительность нейронов



$$y = \mathbb{I}\left[\sum_{i=1}^N w_i x_i > b\right] = \mathbb{I}[w^T x > b]$$

# Модель 3: Непрерывная активация

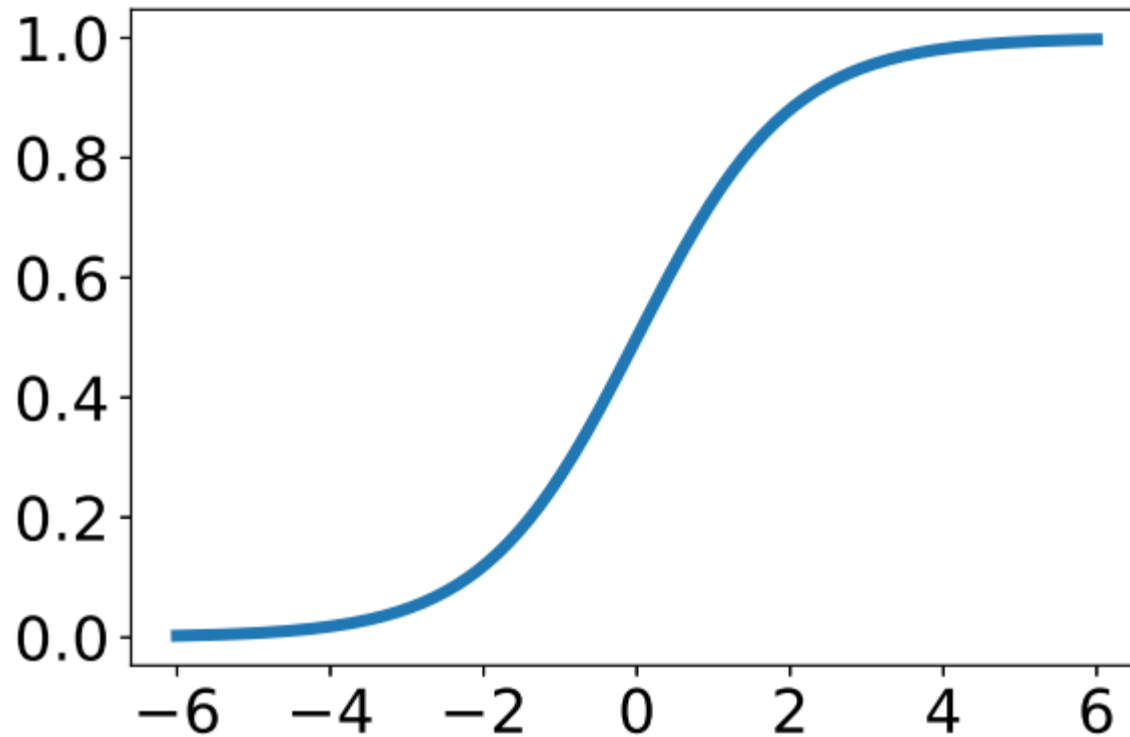


$$y = \sigma\left[\sum_{i=1}^N w_i x_i - b\right] = \sigma[w^T x - b]$$

- Параметры нейрона:  $w$  — веса,  $b$  — смещение.

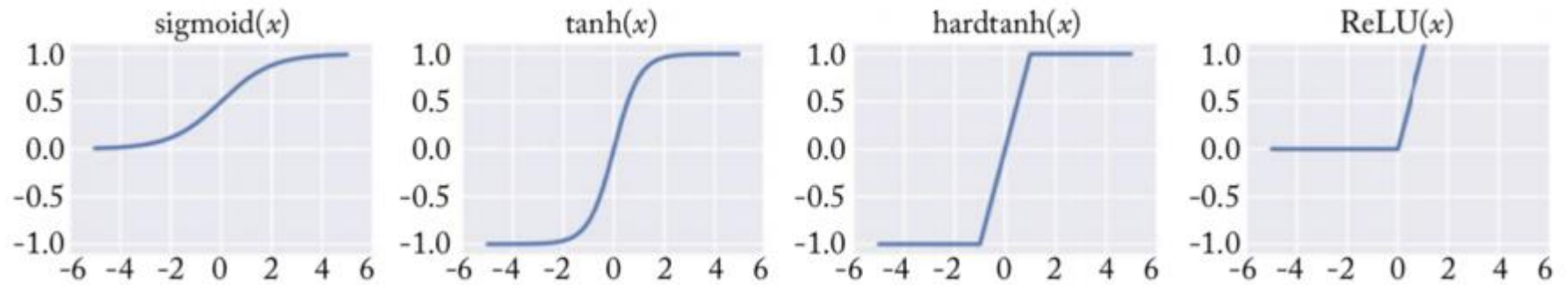


# Функция активации

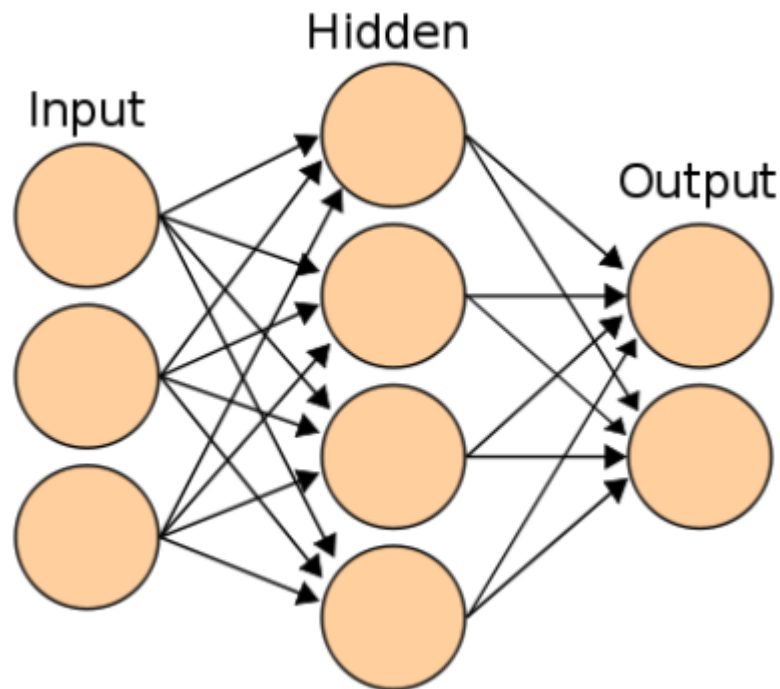


$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

# Нелинейные функции активации



# Сети с одним скрытым слоем



- **Теорема (универсальный аппроксиматор)** Любую непрерывную на компакте функцию можно равномерно приблизить нейронной сетью с одним скрытым слоем.

# Как обучить нейронную сеть?

- Обучить нейронную сеть — подобрать значения всех настраиваемых параметров (веса и смещения). Два этапа:
  1. Задать функцию потерь  $L$
  2. Подобрать веса, минимизирующие  $L$ .

# Многослойные сети прямого распространения

$$\text{NN}_{\text{MLP2}}(x) = y$$

$$h_1 = g^1(xW^1 + b^1)$$

$$h_2 = g^2(h^1W^2 + b^2)$$

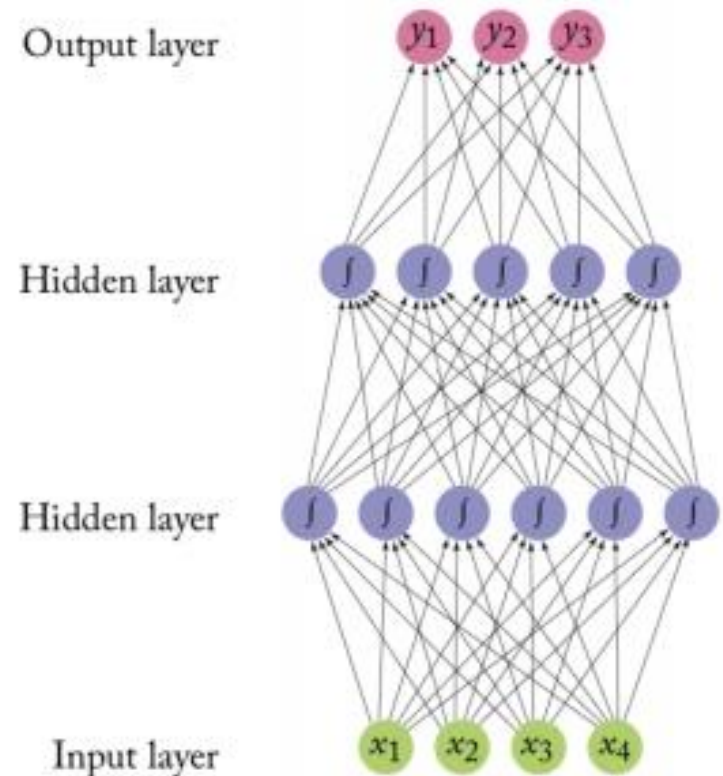
$$y = h^2W^3$$

$$x \in \mathbb{R}^{d_{in}}, y \in \mathbb{R}^{d_{out}}$$

$$W^1 \in \mathbb{R}^{d_{in} \times d_1}, b^1 \in \mathbb{R}^{d_1}$$

$$W^2 \in \mathbb{R}^{d_1 \times d_2}, b^2 \in \mathbb{R}^{d_2}$$

$$W^3 \in \mathbb{R}^{d_2 \times d_{out}}$$



# Векторное представление текста

- Мешок слов [Bag of Words, BoW]
  - $|word \in V| = N$  – словарь
  - $x \in D$  – документ,  $|x| = k$
  - $\bar{x}$  –  $N$ -мерный вектор,  $\bar{x}_i = f(word_i, x_i)$ , в котором  $k$  ненулевых компонент.
- Распределенное представление слов [Continuous Bag of Words, CBoW])
  - one-hot кодировка: каждое слово word –  $N$ -мерный вектор.
  - плотные вектора – эмбединги: каждое слово word –  $d$ -мерный вектор.
  - $CBOW(x) = \frac{1}{k} \sum_i^k E_i$ , матрица эмбедингов:  $E$ .

- Как учесть часть речи или регистр слова?
- Специальные аффиксы?
- Является ли оно именованной сущностью?

# Классификатор на основе сетей прямого распространения

- На последнем слое сети находится функция активации *softmax* для классификации на  $K$  классов:

$$z = W_3 l_3 + b_3 \in \mathbb{R}^K$$

- То есть, на выходном слое находится  $K$  нейронов, каждый соответствует своему классу.
- Для итоговой классификации:

$$y_j = \text{softmax}(z_j) = \frac{e^{z_j}}{\sum_{k \in K} e^{z_k}}$$

- $y_j$  может быть интерпретировано как вероятность класса  $j$
- Как выбрать число скрытых слоев?
  - Например,  $2000 \rightarrow 1000 \rightarrow 500 \rightarrow 100$



# Проблемы полносвязных нейронных сетей

**Проблемы:**

# Проблемы полносвязных нейронных сетей

## Проблемы:

- Vanishing/Exploding gradients

# Проблемы полносвязных нейронных сетей

## Проблемы:

- Vanishing/Exploding gradients
- Требуется огромное количество нейронов

# Проблемы полносвязных нейронных сетей

## Проблемы:

- Vanishing/Exploding gradients
- Требуется огромное количество нейронов
- Серьезное переобучение

# Проблемы полносвязных нейронных сетей

## Проблемы:

- Vanishing/Exploding gradients
- Требуется огромное количество нейронов
- Серьезное переобучение

Возможное **решение**

# Проблемы полносвязных нейронных сетей

## Проблемы:

- Vanishing/Exploding gradients
- Требуется огромное количество нейронов
- Серьезное переобучение

Возможное **решение** — предобработка данных.

# Проблемы полносвязных нейронных сетей

## Проблемы:

- Vanishing/Exploding gradients
- Требуется огромное количество нейронов
- Серьезное переобучение

Возможное **решение** — предобработка данных.

Возможное **решение** — введение новых типов слоев:

# Проблемы полносвязных нейронных сетей

## Проблемы:

- Vanishing/Exploding gradients
- Требуется огромное количество нейронов
- Серьезное переобучение

Возможное **решение** — предобработка данных.

Возможное **решение** — введение новых типов слоев:

- Сверточные слои
- Пулинг
- Dropout
- Нормализация



# dropout-регуляризация

$$\text{NN}_{\text{MLP2}}(x) = y$$

$$h_1 = g^1(xW^1 + b^1)$$

$$m^1 \sim \text{Bernouli}(r^1)$$

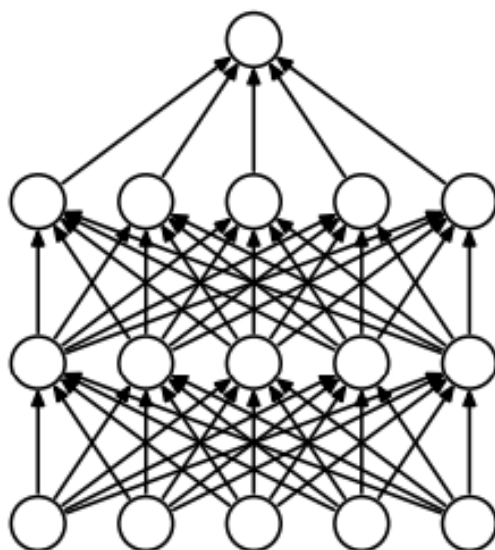
$$\hat{h}^1 = m^1 \odot h^1$$

$$h_2 = g^2(\hat{h}^1 W^2 + b^2)$$

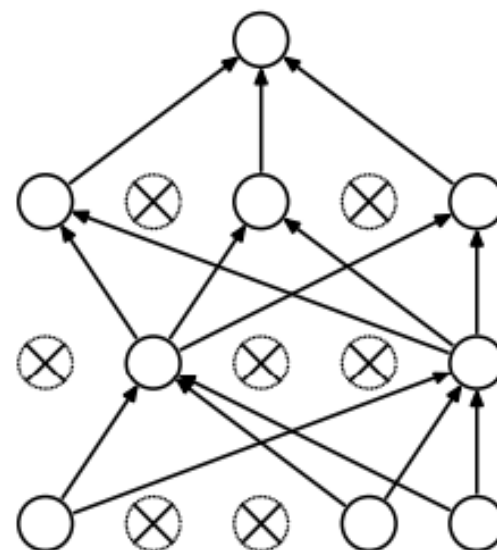
$$m^2 \sim \text{Bernouli}(r^2)$$

$$\hat{h}^2 = m^2 \odot h^2$$

$$y = \hat{h}^2 W^3$$



(a) Standard Neural Net



(b) After applying dropout.

Figure 1: Dropout Neural Net Model. **Left:** A standard neural net with 2 hidden layers. **Right:** An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped.

# Сверточные нейронные сети

**Сверточные нейронные сети** [англ. convolutional neural network]:

- Заимствованы из области компьютерного зрения.
- Пик популярности пришелся на 2014 (до +10% аккуратности в задачах классификации), со временем были вытеснены рекуррентными нейронными сетями.

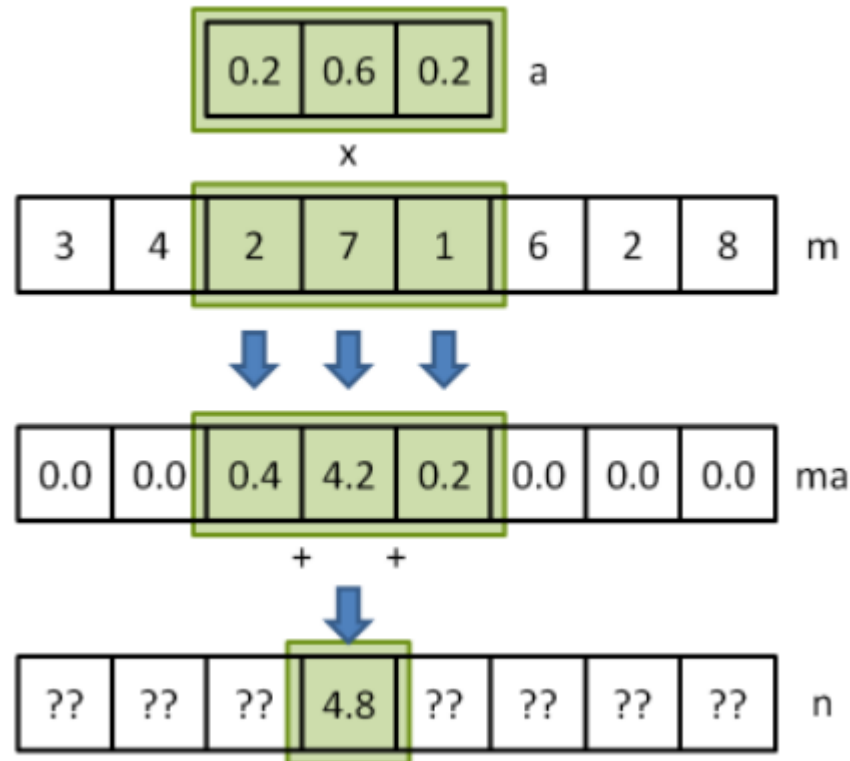
**Помогают справиться с двумя проблемами:**

- Часто входы бывают переменной длины (тексты, абзацы, предложения)
- Если использовать подход, основанный на представлении предложениями окнами, то:
  - число параметров увеличивается,
  - нужно подбирать размер окна.

# Одномерная свертка

## Определение

- Результатом операции свертки массива  $m$  с ядром  $a$  называется сигнал  $n$ :  $n[k] = \sum_{i=-w}^w m[k+i]a[-i]$ . Обозначение:  $n = m * a$ .



# Padding

- Нулевой отступ

0	0	A	B	C	0	0
---	---	---	---	---	---	---

- Продолжение границы

A	A	A	B	C	C	C
---	---	---	---	---	---	---

- Зеркальный отступ

B	A	A	B	C	C	B
---	---	---	---	---	---	---

C	B	A	B	C	B	A
---	---	---	---	---	---	---

- Циклический отступ

B	C	A	B	C	A	B
---	---	---	---	---	---	---

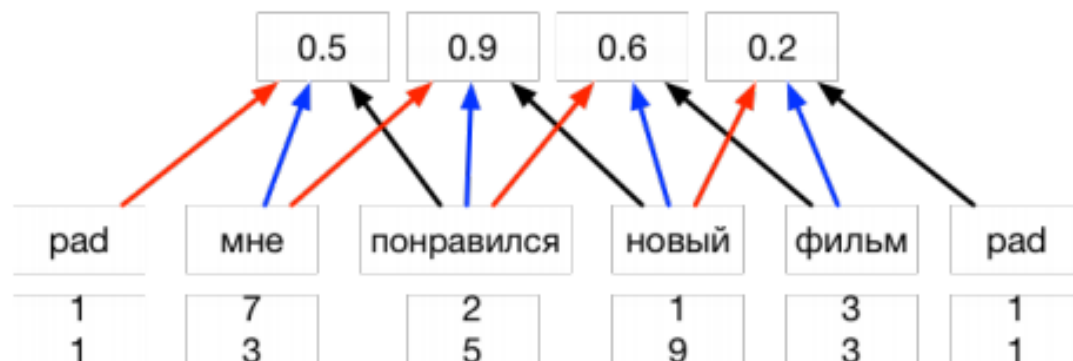
# Слой свертки

Фильтр [англ. filter]:

- $w_{1:n}$  – входная последовательность слов,
- $E_{w_i}$  – эмбединг слова  $w_i$ ,

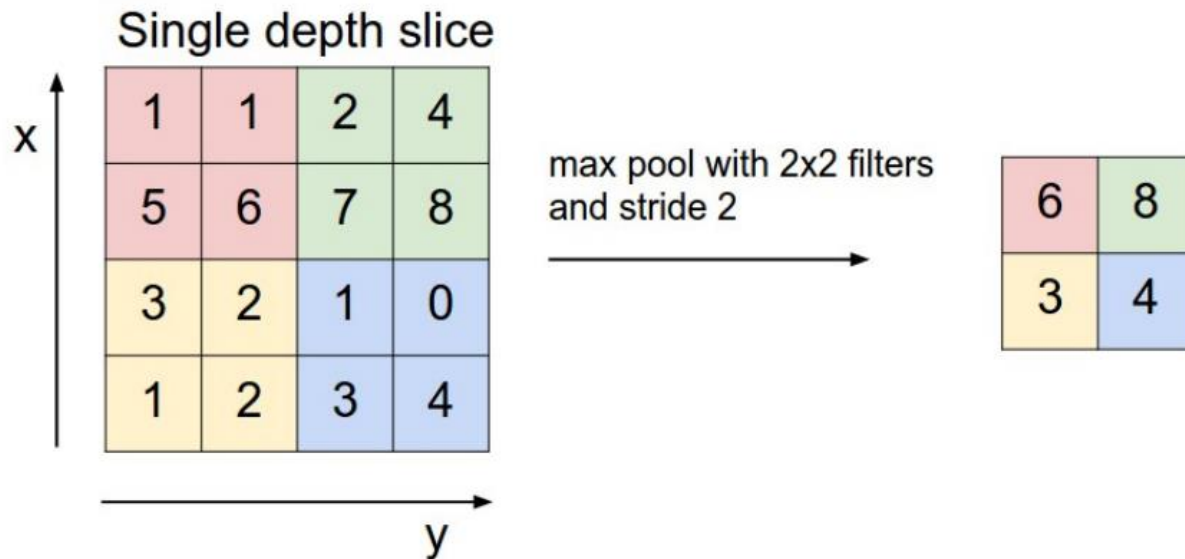
Фильтр:  $p_i = g(x_i u)$

$p_i \in \mathbb{R}, x_i \in \mathbb{R}^{k \cdot d_{emb}}, u \in \mathbb{R}^{k \cdot d_{emb}}$



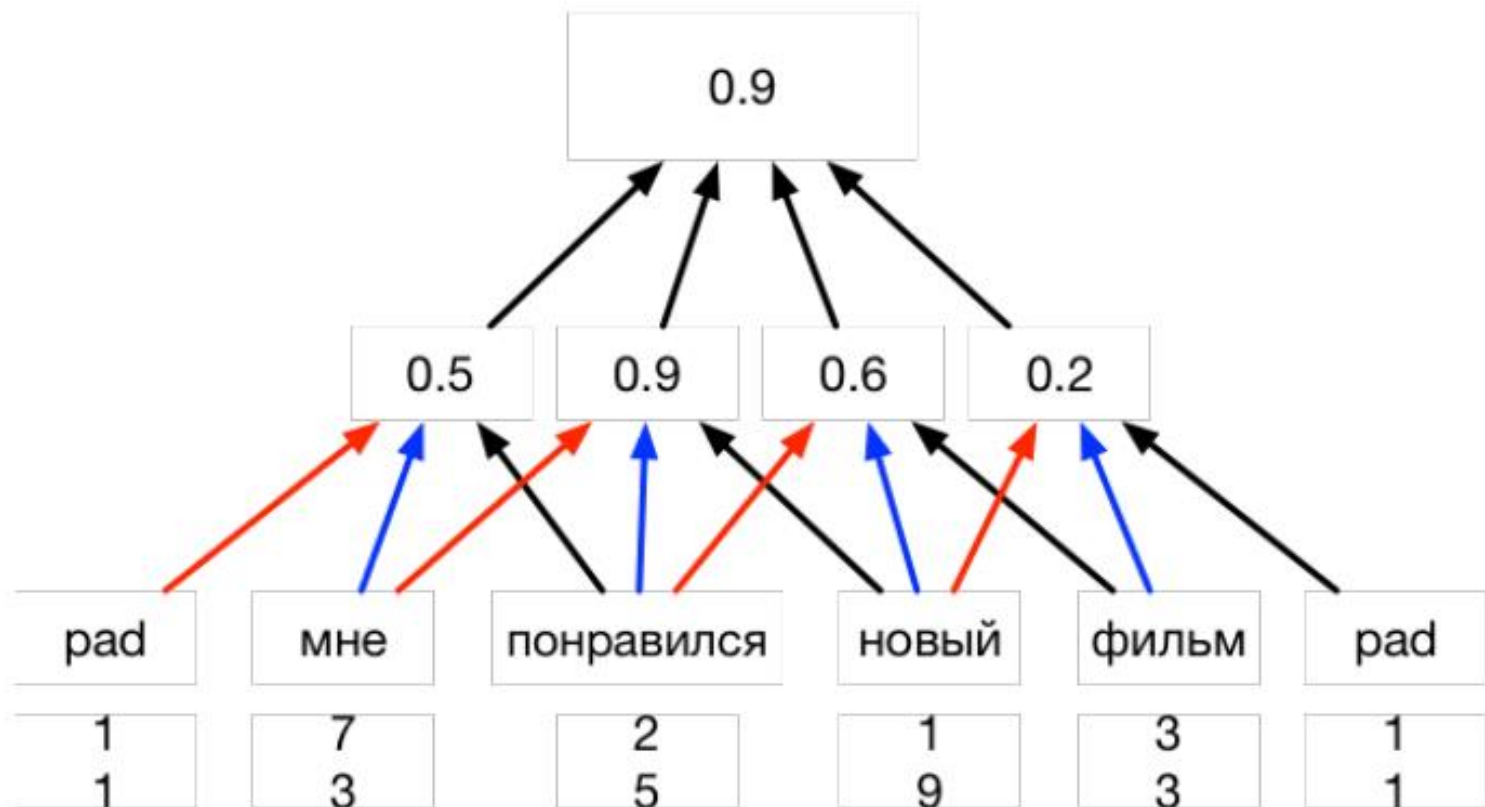
- Преобразуем каждое входное окно, но пока размерность входа не уменьшается!

# Pooling



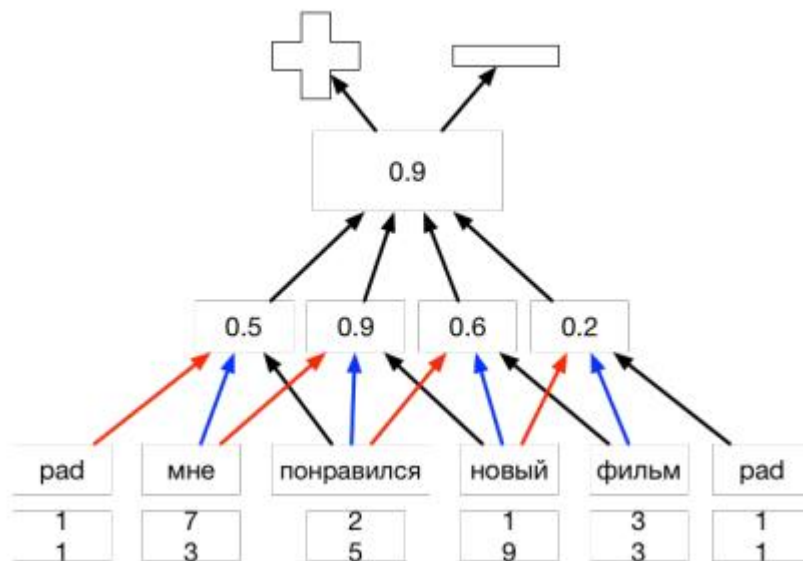
- Голосование: побеждают наиболее активные нейроны.
- Вырабатывается инвариантность к небольшим сдвигам (в рамках окна).
- Увеличение рецептивной области.
- Уменьшение вычислительных затрат
- Кроме max-пулинга: mean, weighted, root-mean-square, ...

# Pooling



# Классификатор на основе сверточной сети

- $y \in [0, 1]$  - истинные значения
- $\hat{y} = c$  - предсказанные значения.



- Для обучения сверточной сети можно использовать обычный алгоритм распространения ошибки.
- Одномерные фильтры – это сильное ограничение. Что делать, если  $c = 0.5$ ?



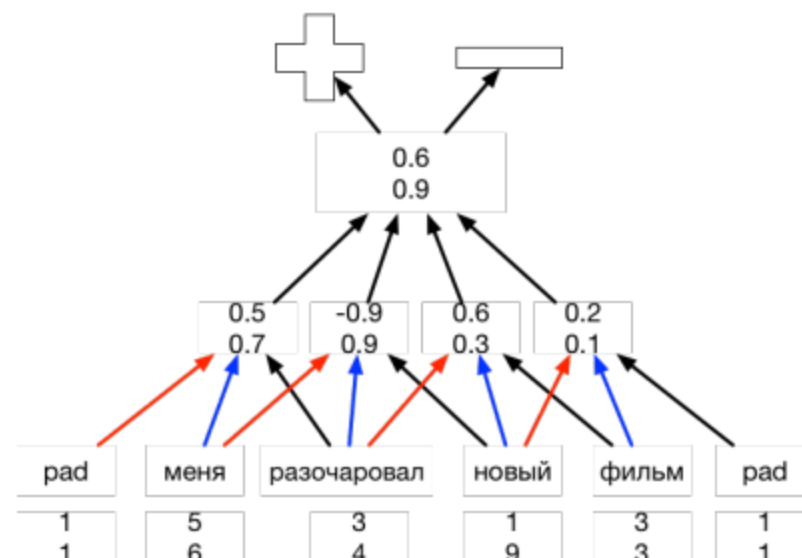
# Многомерные фильтры

Применяем фильтр  $l$  раз:

$$p_i = g(x_i \cdot U + b)$$

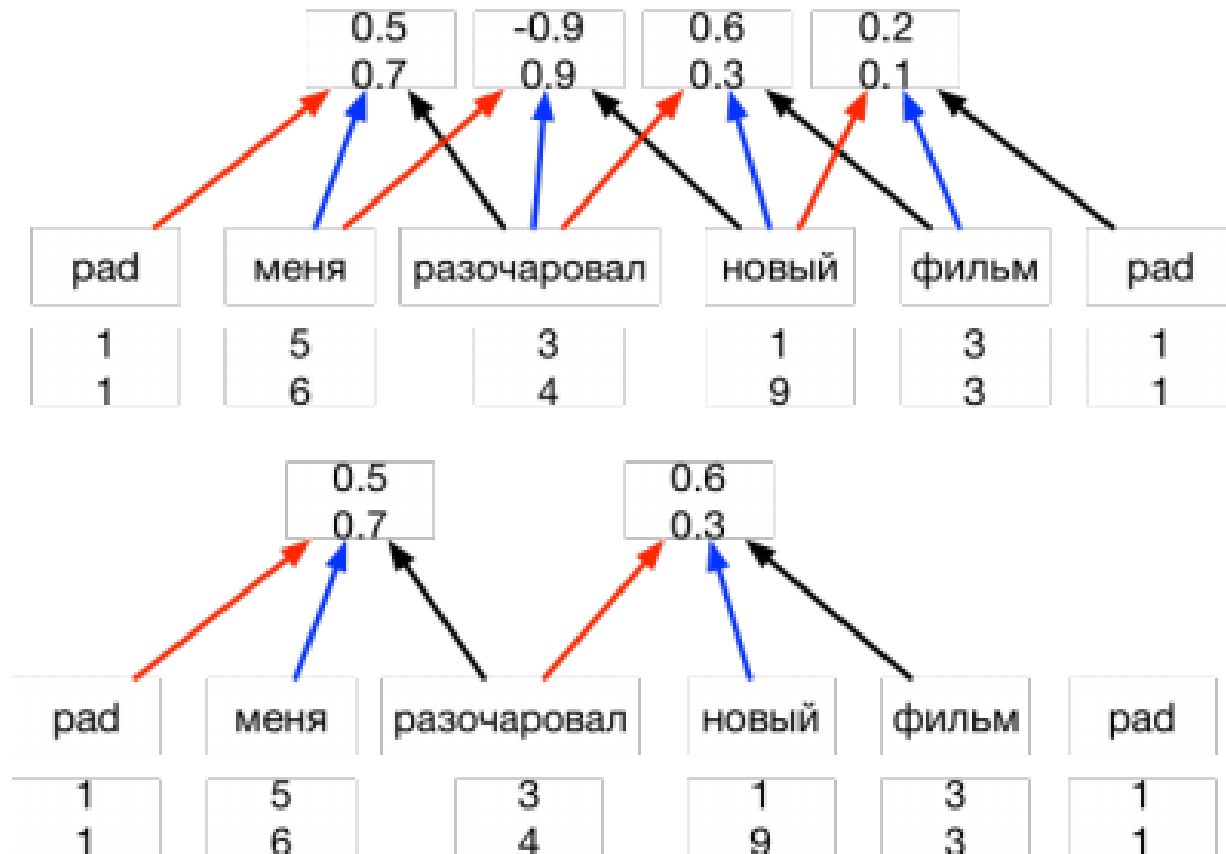
$$p_i \in \mathbb{R}^l, x_i \in \mathbb{R}^{k \cdot d_{emb}}$$

$$U \in \mathbb{R}^{k \cdot d_{emb} \times l}, b \in \mathbb{R}^l$$



# Шаг окна

- Можно использовать непересекающиеся окна, чтобы уменьшить объем вычисления



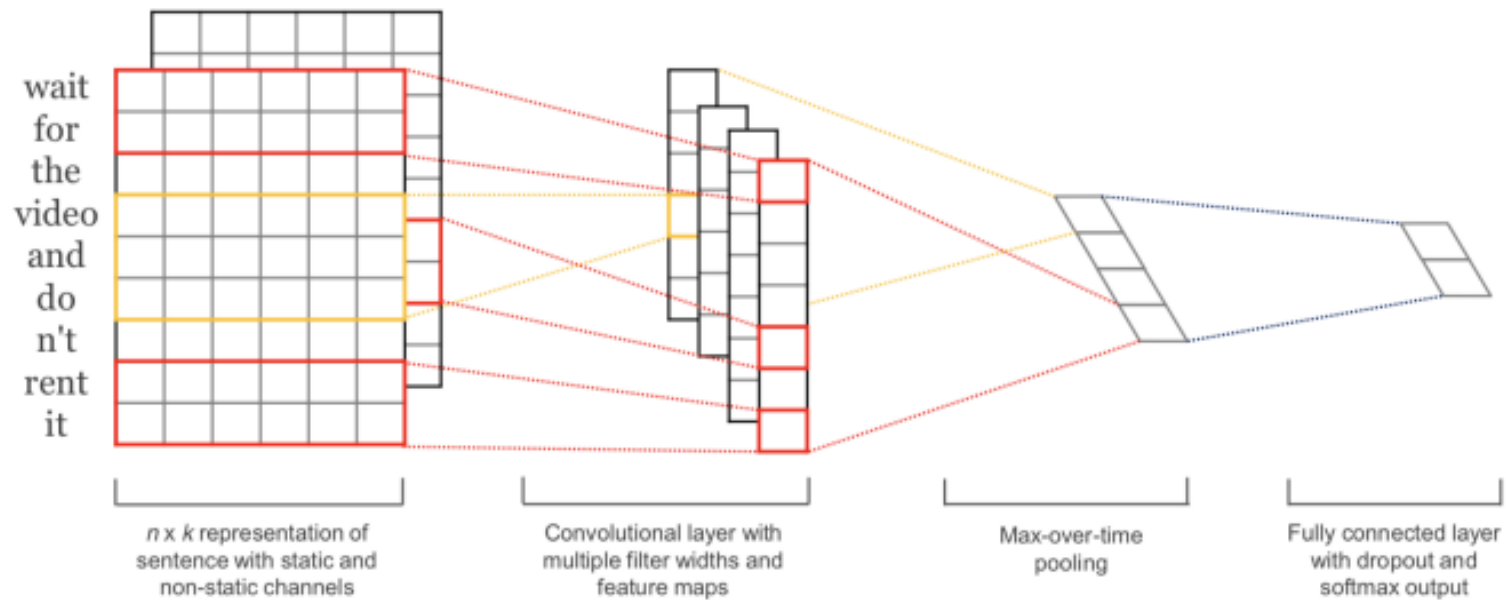
# Как выбирать вектора слов?

- Случайная инициализация (если нет обученных моделей word2vec, GloVe).
- word2vec, GloVe без обновления.
- word2vec, GloVe с обновлением на каждой эпохе (увеличивается количество параметров!)
- Несколько каналов: копируем два входа и
  - на один подаем word2vec и не обновляем эти входы во время обучения, на второй подаем word2vec и обновляем эти входы во время обучения
  - на один вход подаем word2vec, на второй – GloVe.

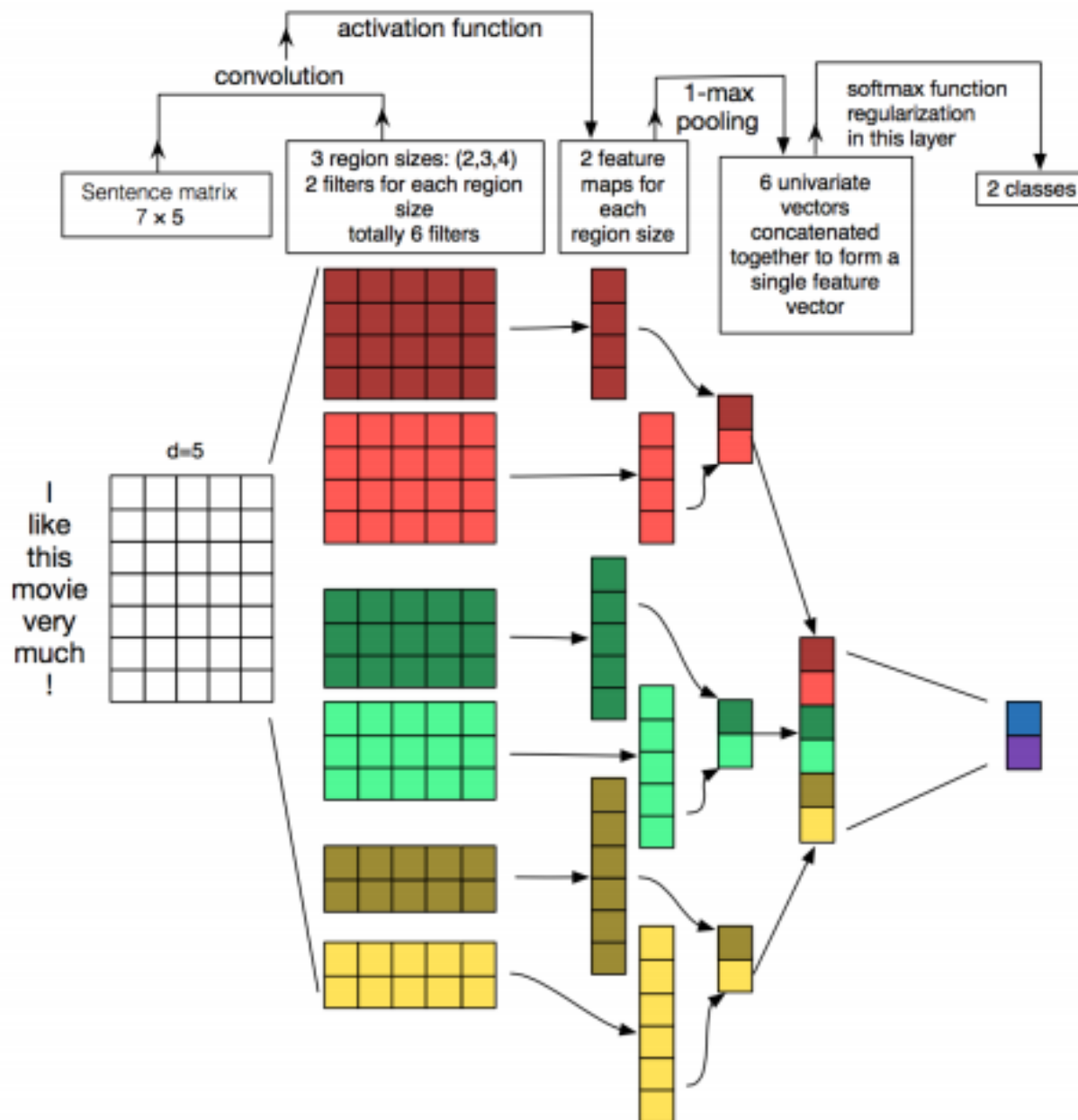
# Как использовать padding?

- [[мое первое короткое предложение], [второе очень длинное предложение, которое никогда не заканчивается], [третье предложение]]
- Неэффективный способ: одно предложение – одна эпоха
- Окружить все предложения балластными символами pad и сделать их одной длины
  - Надо убедиться, что max-пулинг не выберет значения, соответствующие pad.
  - Надо убрать выбросы, то есть, супер-длинные предложения, возникшие, например, из-за ошибок сегментатора

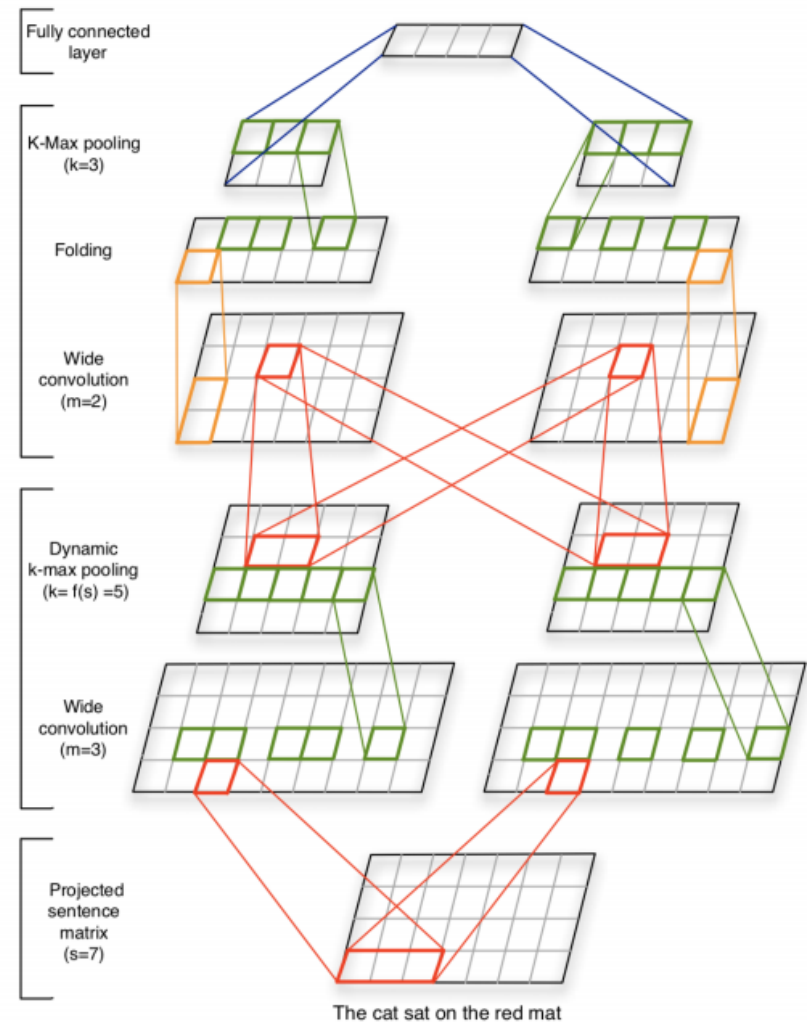
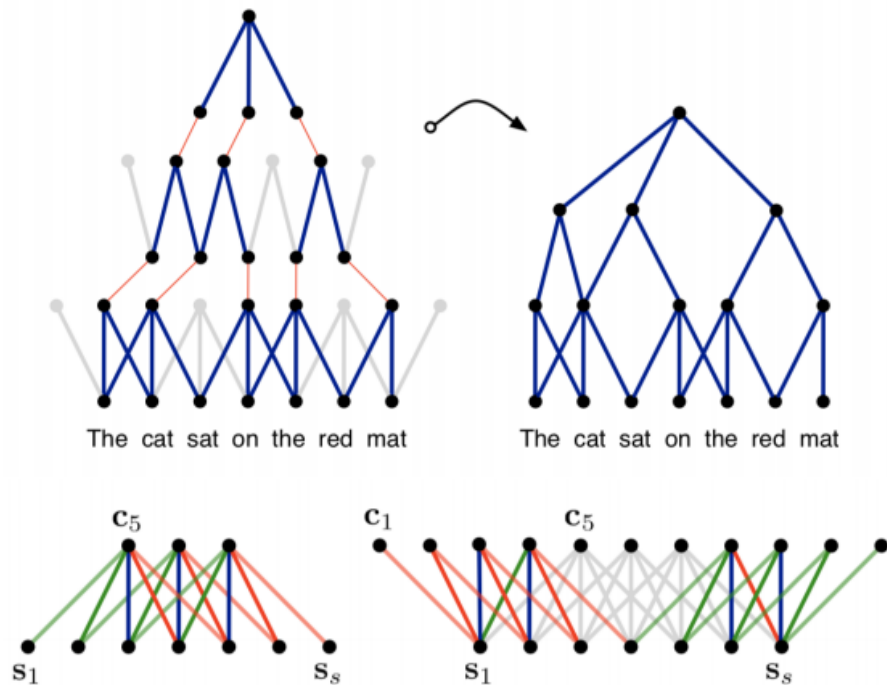
# CNN для классификации предложений [Kim14]



# CNN для классификации предложений [ZW15]

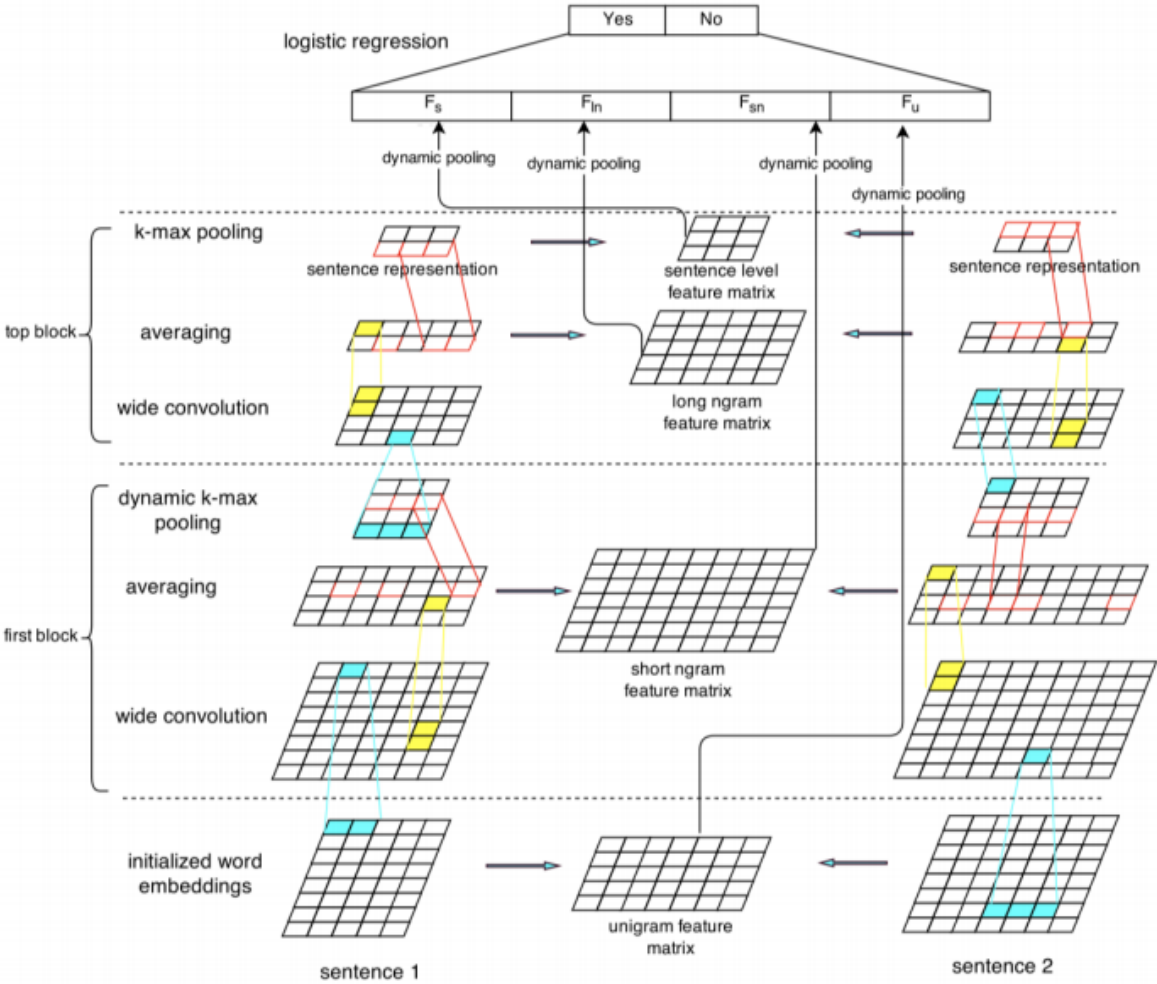


# CNN для классификации предложений [KGB14]



A1: Detroit  
manufacturers have  
raised vehicle prices by  
ten percent  
A2: GM, Ford and  
Chrysler have raised car  
prices by five percent

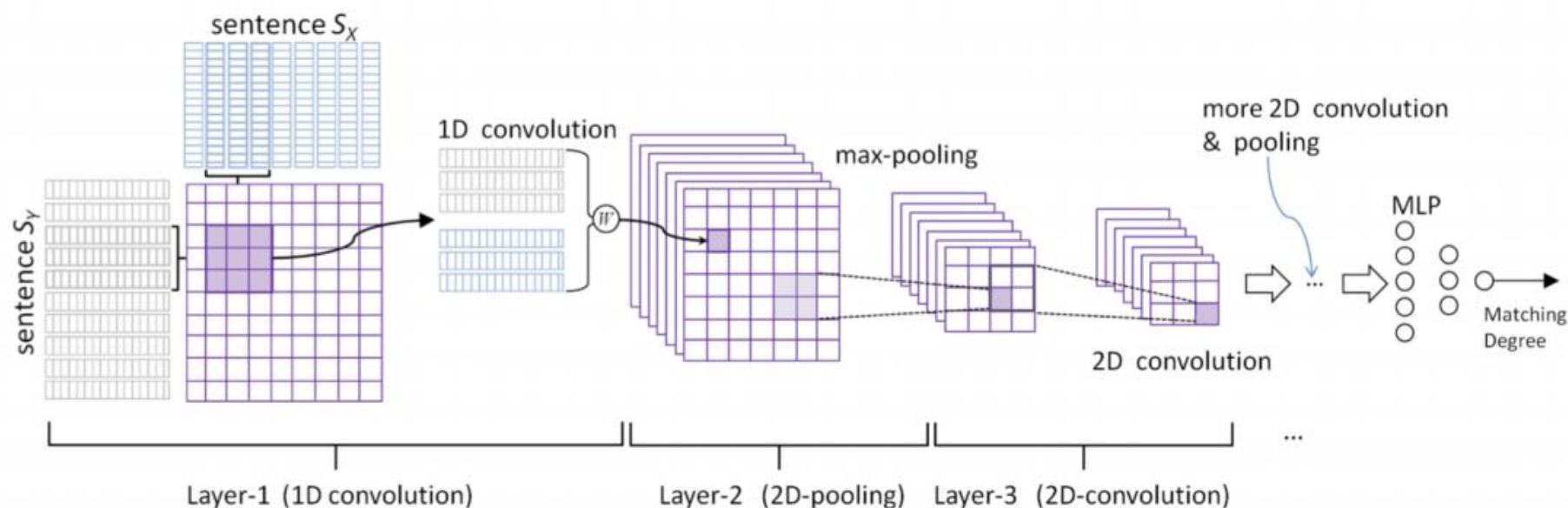
B1: Mary gave birth to  
a son in 2000  
B2: He is 18 years old  
and his mother is Mary





# CNN для определения близости между предложениями[HLLC14]

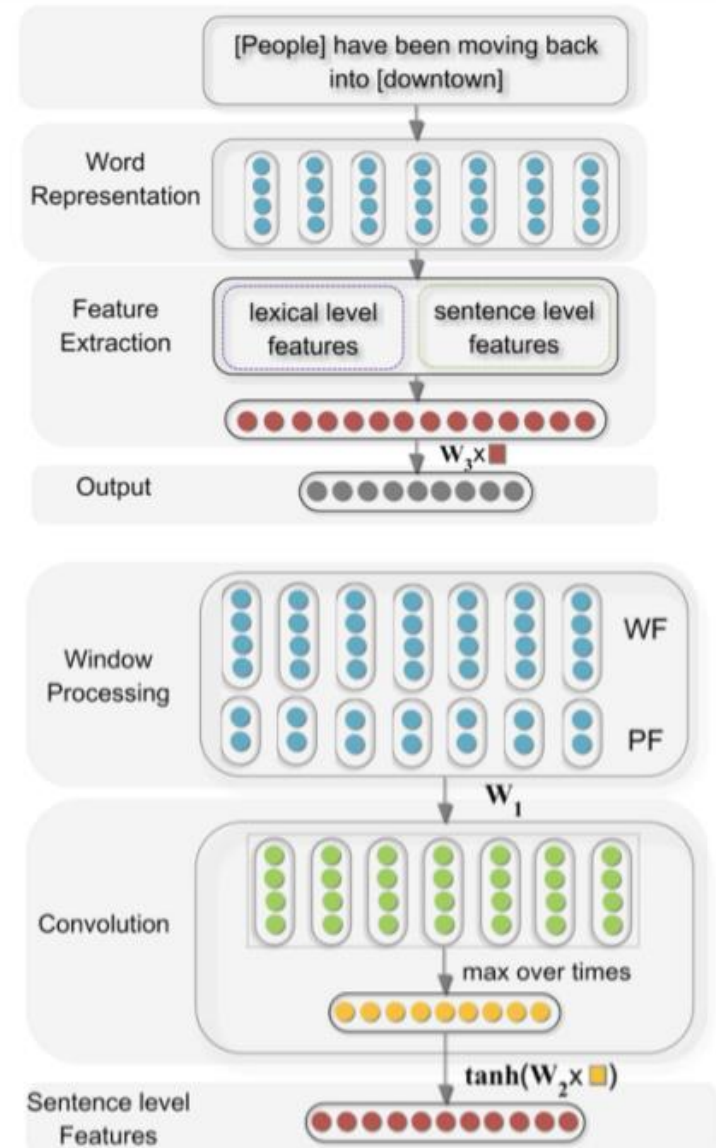
- Дополнение предложения [sentence completion]
- Определение ответа на вопрос [matching a response to a tweet]
- Определения парафраза [paraphrase detection]



# CNN для извлечения отношений [relation extraction] [ZLL+14]

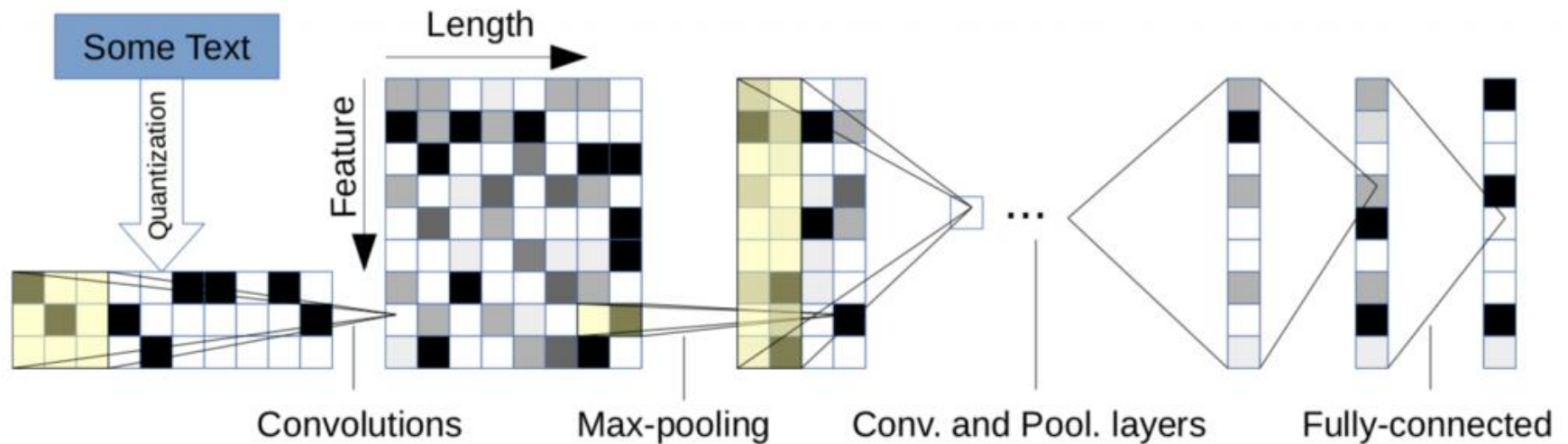
The [fire]<sub>e1</sub> inside WTC was caused by exploding [fuel]<sub>e2</sub>

The [company]<sub>e1</sub> fabricates [plastic chairs]<sub>e2</sub>



# Символьные CNN для классификации предложений [ZZL15]

- Представление текста: one-hot вектора для 70 алфавитных и неалфавитных символов



# Рекуррентные нейронные сети

- RNN позволяют уйти от Марковских допущений и позволяют учитывать предысторию произвольной длины.

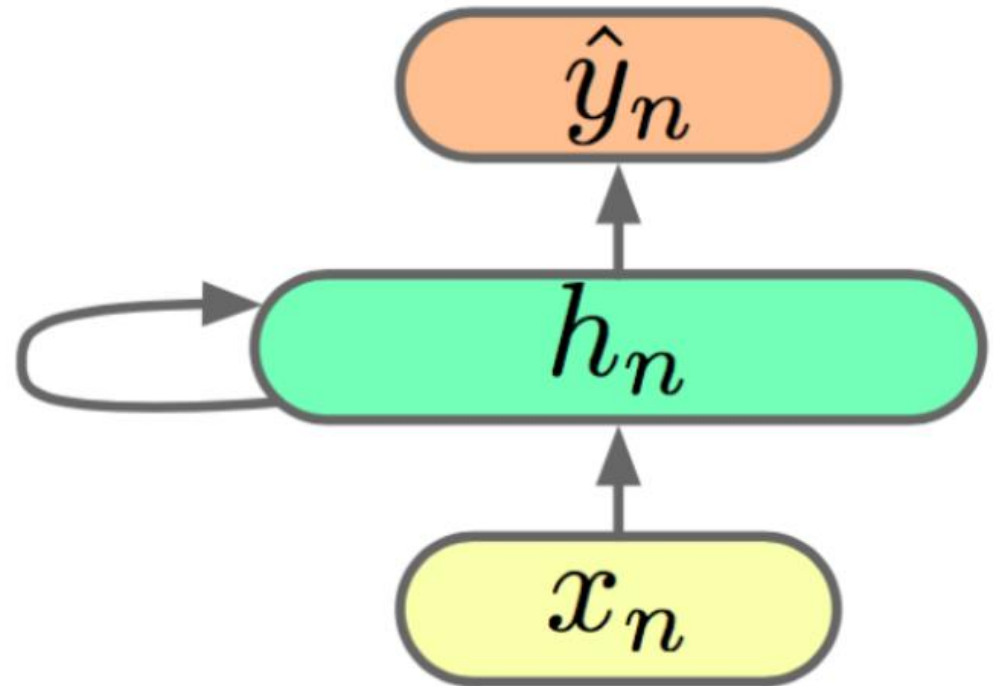
$$x_{1:n} = x_1, x_2, \dots, x_n, x_i \in \mathbb{R}^{d_{in}}$$

$$y_n = RNN(x_{1:n}), y_n \in \mathbb{R}^{d_{out}}$$

Для каждого префикса  $x_{1:i}$   $y_i$  – выходной вектор.

$$y_i = RNN(x_{1:i})$$

$$y_{1:n} = RNN^*(x_{1:n}), y_i \in \mathbb{R}^{d_{out}}$$



# Рекуррентные нейронные сети

$R$  – рекурсивная функция с двумя входами:  $x_i$  и  $s_{i-1}$  (вектор состояния)

$$RNN^*(x_1 : n, s_0) = y_1 : n$$

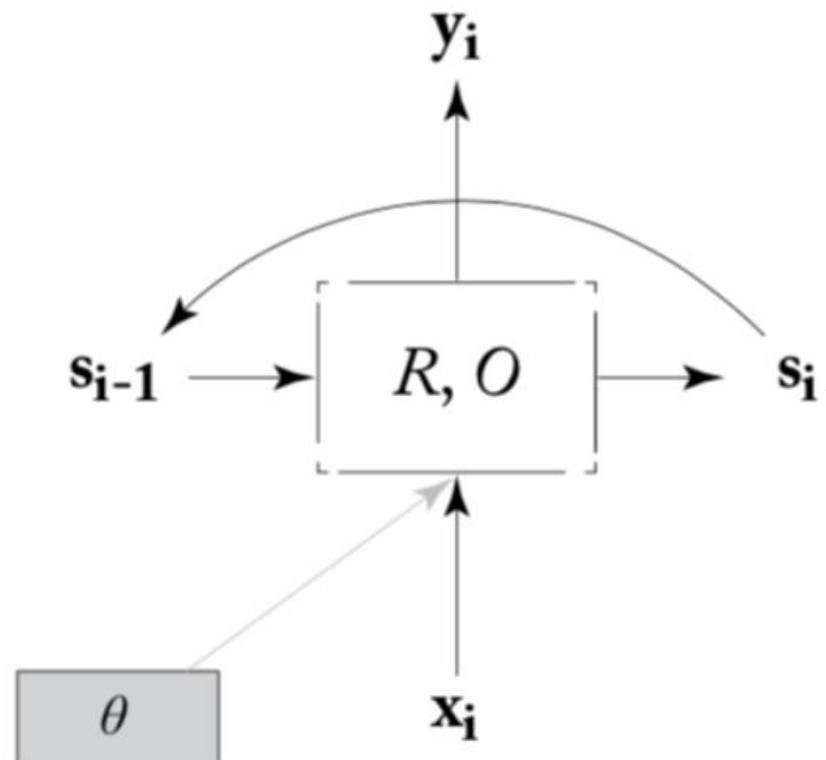
$$y_i = O(s_i)$$

$$s_i = R(s_{i-1}, x_i)$$

$$s_i = R(s_{i-1}, x_i) = g(s_{i-1}W^s + x_iW^x + b)$$

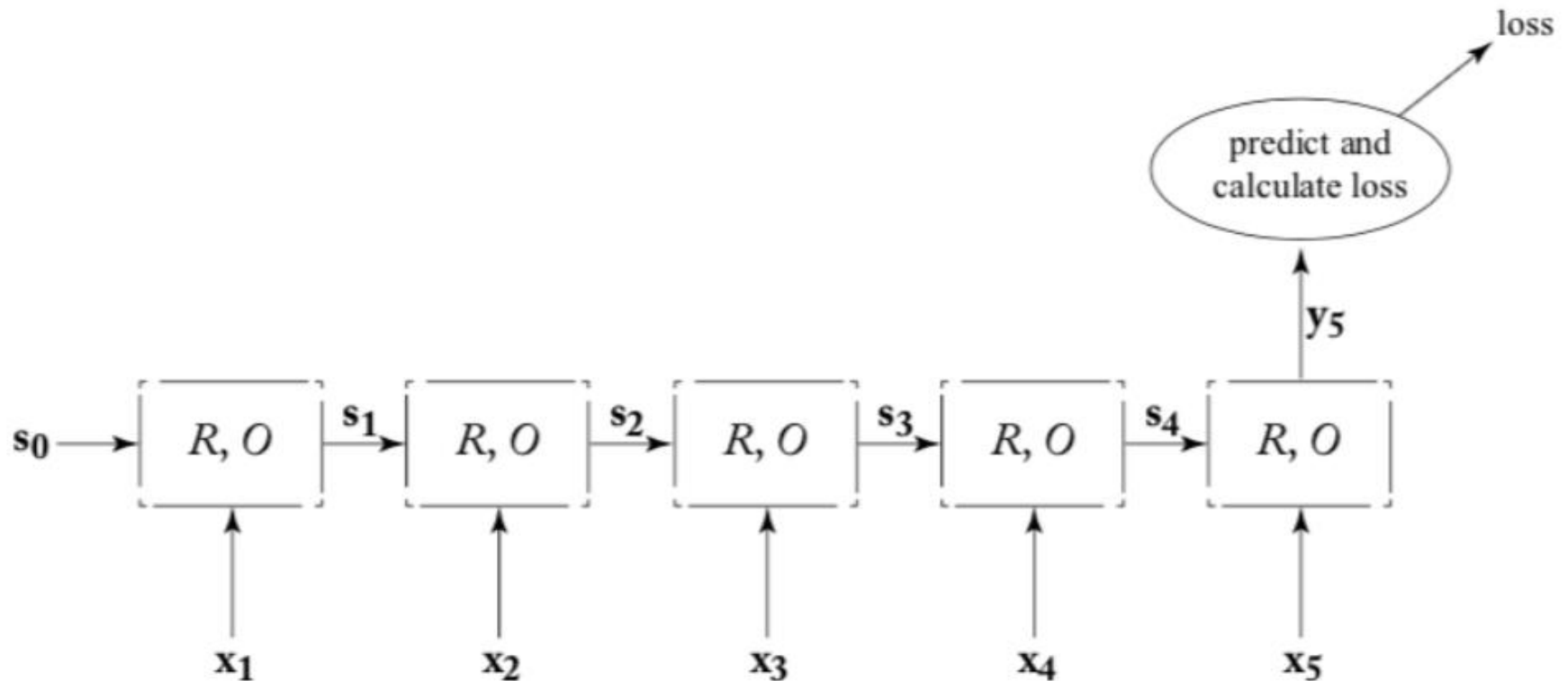
$$x_i \in \mathbb{R}^{d_{in}}, y_i \in \mathbb{R}^{d_{out}}, s_i \in \mathbb{R}^{d_{out}}$$

$$W^x \in \mathbb{R}^{d_{in} \times d_{in}}, W^s \in \mathbb{R}^{d_{out} \times d_{out}}$$



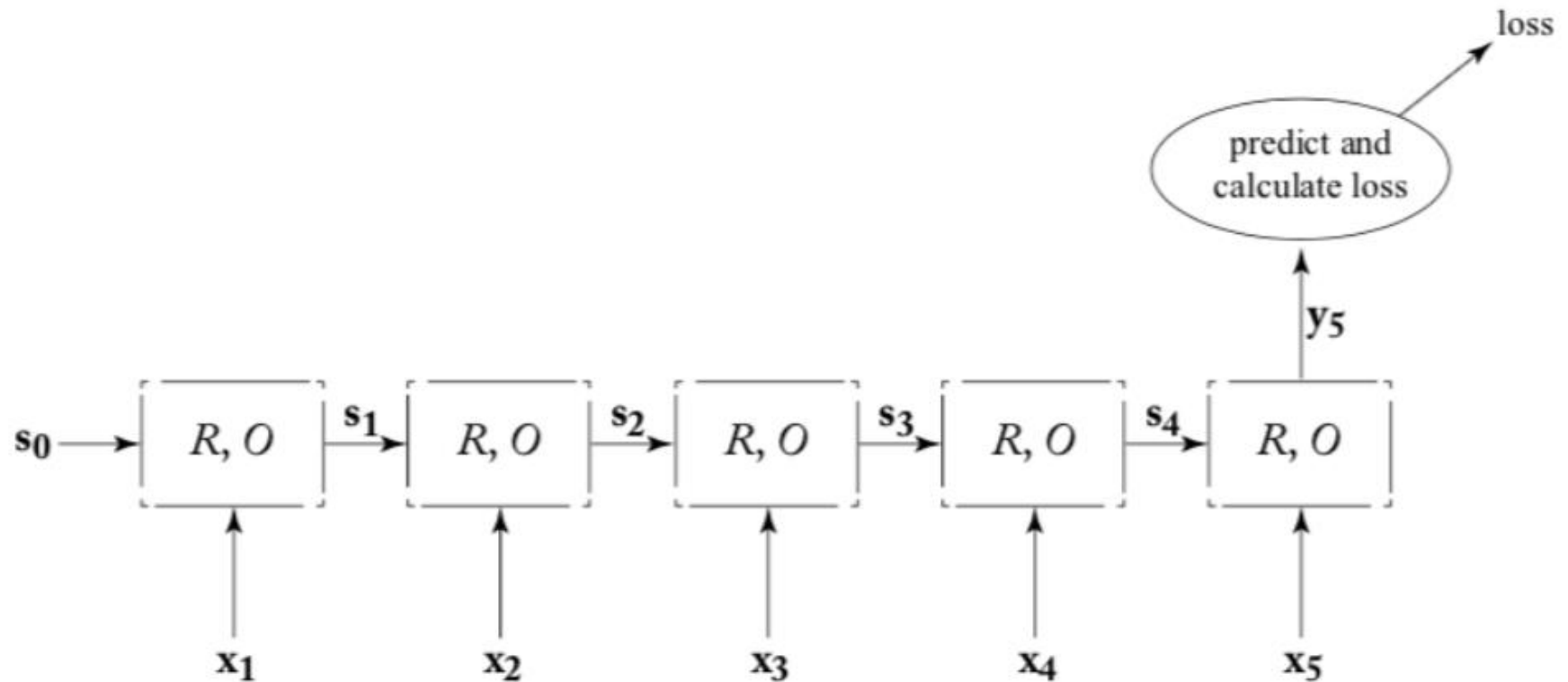
# Сценарии использования RNN

- Ассектор: только  $y_n$ , используемый для дальнейшего предсказания / классификации



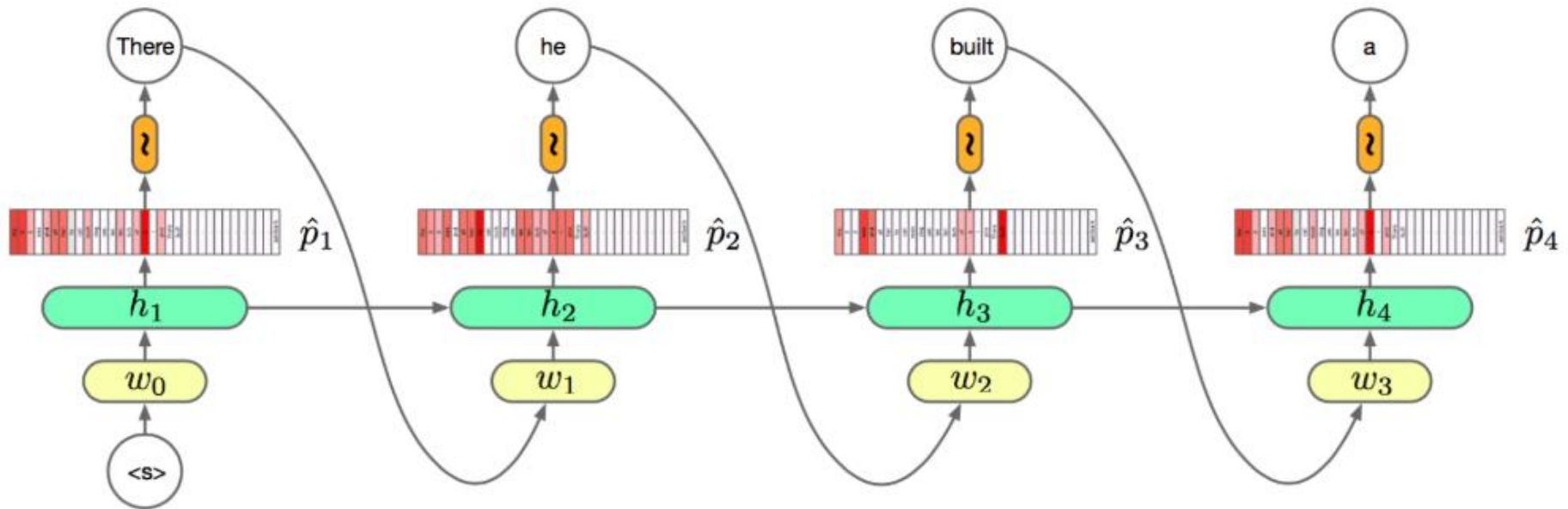
# Сценарии использования RNN

- Encoder: только  $y_n$



# Сценарии использования RNN

- Transducer: выход  $p_i$  для каждого входа  $w_0: w_i$  — языковые модели, sequence labelling





- **RNN** трудно обучать:
  - проблема исчезающего градиента.
- Уйти от нее помогают управляемые нейроны специального вида: **LSTM** и **GRU**.

- Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen, Convolutional neural network architectures for matching natural language sentences, Advances in neural information processing systems, 2014, pp. 2042–2050.
- Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom, A convolutional neural network for modelling sentences, arXiv preprint arXiv:1404.2188 (2014).
- Yoon Kim, Convolutional neural networks for sentence classification, arXiv preprint arXiv:1408.5882 (2014).
- Wenpeng Yin and Hinrich Schutze, " Convolutional neural network for paraphrase identification, Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015, pp. 901–911.

- Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao, Relation classification via convolutional deep neural network, Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, 2014, pp. 2335–2344.
- Ye Zhang and Byron Wallace, A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification, arXiv preprint arXiv:1510.03820 (2015).
- Xiang Zhang, Junbo Zhao, and Yann LeCun, Character-level convolutional networks for text classification, Advances in neural information processing systems, 2015, pp. 649–657.

**СПАСИБО ЗА ВНИМАНИЕ**