



Факультет компьютерных
наук

Аналитика больших
данных

Москва
2025

Классификация



Задача текстовой классификации

2 |

- Спам / не спам
- Тональность (sentiment)
- Тема документа / рубрика
- Токсичность / модерация
- Intent в чат-боте
- Авторство / стиль (по желанию)

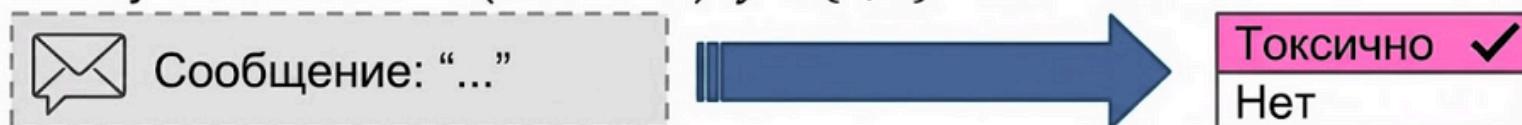




1. Какие бывают постановки задачи:
 - binary vs multi-class vs multi-label
2. Генеративный и дискриминативные модели
3. Классические алгоритмы: Naive Bayes, LogReg, SVM
4. Нейросетевые модели: RNN и CNN

Постановки задачи классификации: binary vs multi-class vs multi-label

Binary Classification (2 класса): $y \in \{0, 1\}$

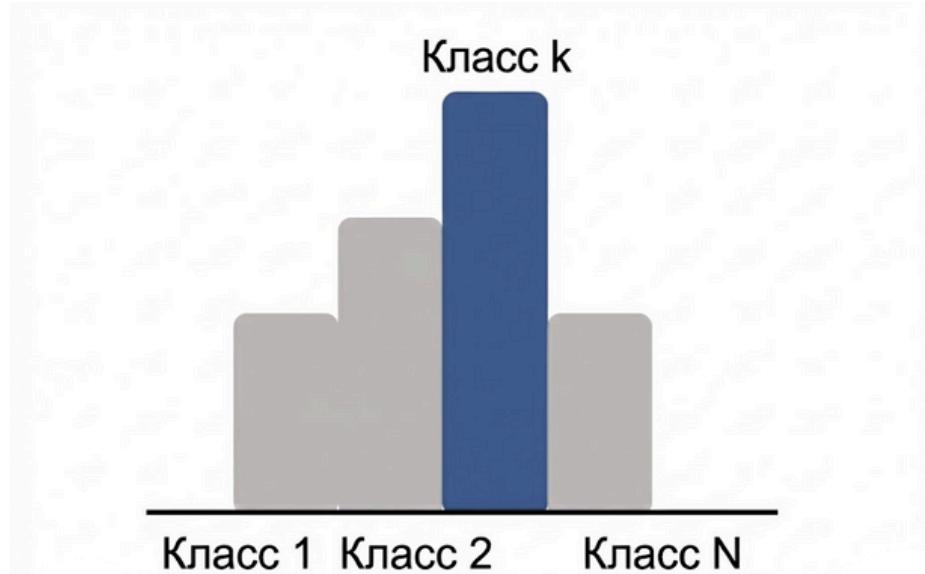


Multi-class Classification (1 из K): $y \in \{1, \dots, K\}$



Multi-label Classification (несколько из K): $y \in \{0, 1\}^K$




$$p(\text{Текст} = \text{Класс } k | \text{вектор текста})$$



Постановка задачи

Пусть $z \in \mathbb{R}^K$ - logits.

- Single-label (binary/multi-class):

$$p = \text{softmax}(z), \quad \sum_k p_k = 1$$

Решение: $\arg \max_k p_k$

- Multi-label:

$$p_k = \sigma(z_k) \text{ для каждого } k$$

Решение: $p_k > \tau$ - порог τ для каждой метки

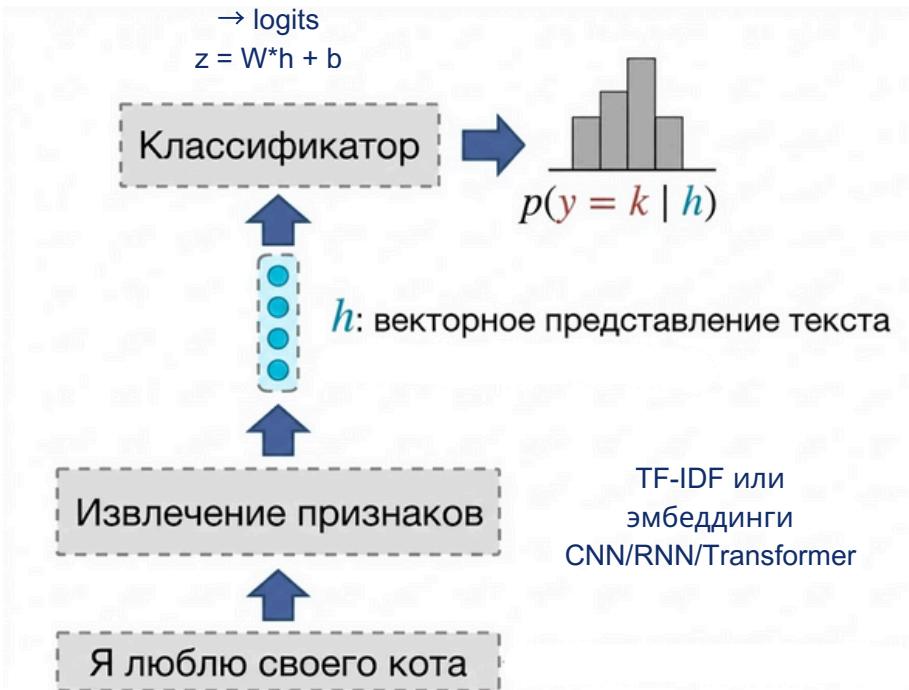
Лосс

- Cross-Entropy (single-label):

$$L = -\log p_y$$

- Binary Cross-Entropy по меткам (multi-label):

$$L = -\sum_{k=1}^K [y_k \log p_k + (1 - y_k) \log(1 - p_k)]$$





Softmax vs Sigmoid

в классификации текста

7

SOFTMAX

- Вход: logits $z \in \mathbb{R}^K$
- Выход: вектор вероятностей $p \in [0, 1]^K$
- $p_k = \exp(z_k) / \sum_j \exp(z_j)$
- Свойство: $\sum_k p_k = 1$ (классы конкурируют)
- Когда: single-label (multi-class, ровно один класс)
- Решение: $\arg \max_k p_k$
- Лосс: cross-entropy (NLL): $L = -\log p_y$

SIGMOID

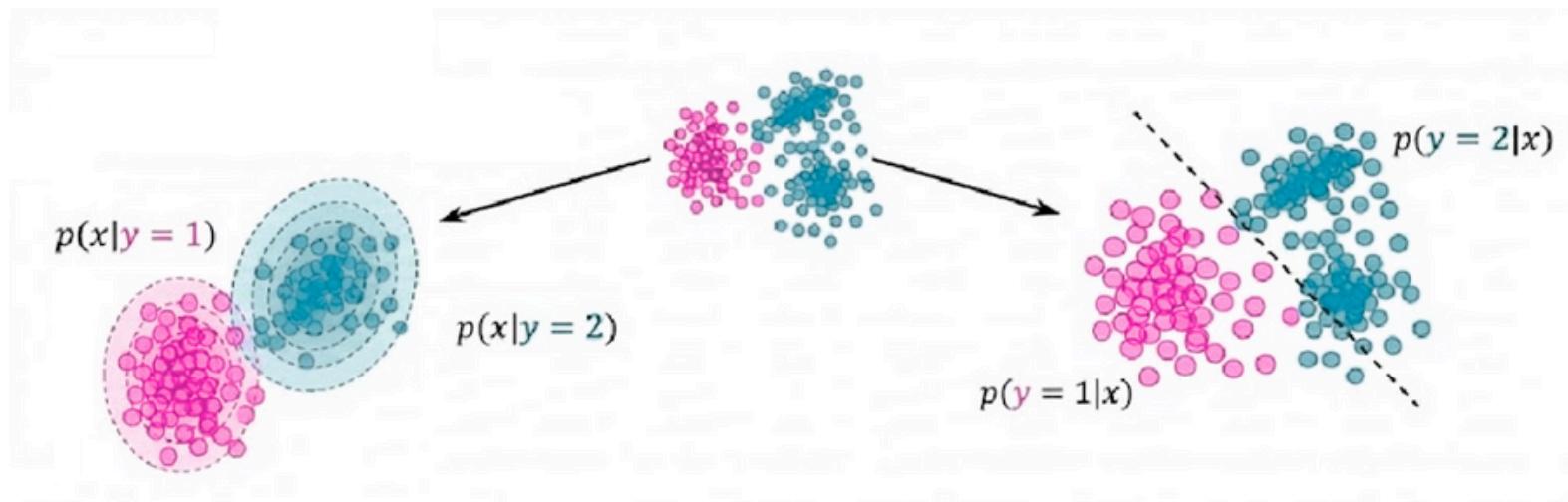
- Вход: logits $z \in \mathbb{R}^K$ (или один логит)
- Выход: независимые вероятности $p_k \in [0, 1]$
- $p_k = \sigma(z_k) = \frac{1}{1 + \exp(-z_k)}$
- Свойство: p_k НЕ суммируются (классы независимы)
- Когда: binary (да/нет) и multi-label (несколько меток)
- Решение: $p_k > \tau_k$ (пороги)
- Лосс: BCE по меткам:
$$L = -\sum_k [y_k \log p_k + (1 - y_k) \log(1 - p_k)]$$

logits ($z = [1.2, 0.3, -0.7]$) →

softmax($(z) = [0.64, 0.26, 0.10]$) (сумма=1) | sigmoid($(z) = [0.77, 0.57, 0.33]$) (независимо)

Binary - частный случай: $K=2$ (softmax) или $K=1$ (sigmoid)

x - объект (текст/признаки/эмбеддинг), y - класс.



оцениваем $p(x | y = 1)$ и $p(x | y = 2)$,

затем получаем $p(y | x) \propto p(x | y) p(y)$

сразу строим $p(y = 1 | x)$, $p(y = 2 | x)$



Naive Bayes (наивный Байес)

9

Интуиция: “какие слова типичны для класса?”

$$P(x \mid y = k) = P(x_1, \dots, x_n \mid y = k) \approx \prod_{t=1}^n P(x_t \mid y = k)$$

$$y^* = \arg \max_k P(x, y = k) = \arg \max_k P(y = k) \cdot P(x \mid y = k)$$

$$\log P(x, y = k) = \log P(y = k) + \sum_{t=1}^n \log P(x_t \mid y = k)$$



Naive Bayes (наивный Байес)

10

Интуиция: “какие слова типичны для класса?”

Как именно устроены признаки x_t

Bernoulli NB: $x_t \in \{0, 1\}$ - слово встретилось/не встретилось

Multinomial NB: $x_t \in \mathbb{N}$ - “сколько раз встретилось слово” (самый частый вариант для текстов)

Как оцениваются вероятности и почему нужно сглаживание

Иначе получаются нули.

- prior: $p(y = k) = \frac{N_k}{N}$
- likelihood (пример для MultinomialNB):

$$p(w | y = k) = \frac{N_{w,k} + \alpha}{\sum_{w' \in V} N_{w',k} + \alpha |V|}$$

где α - сглаживание Laplace/add- α



Что если слово ни разу не встречалось?

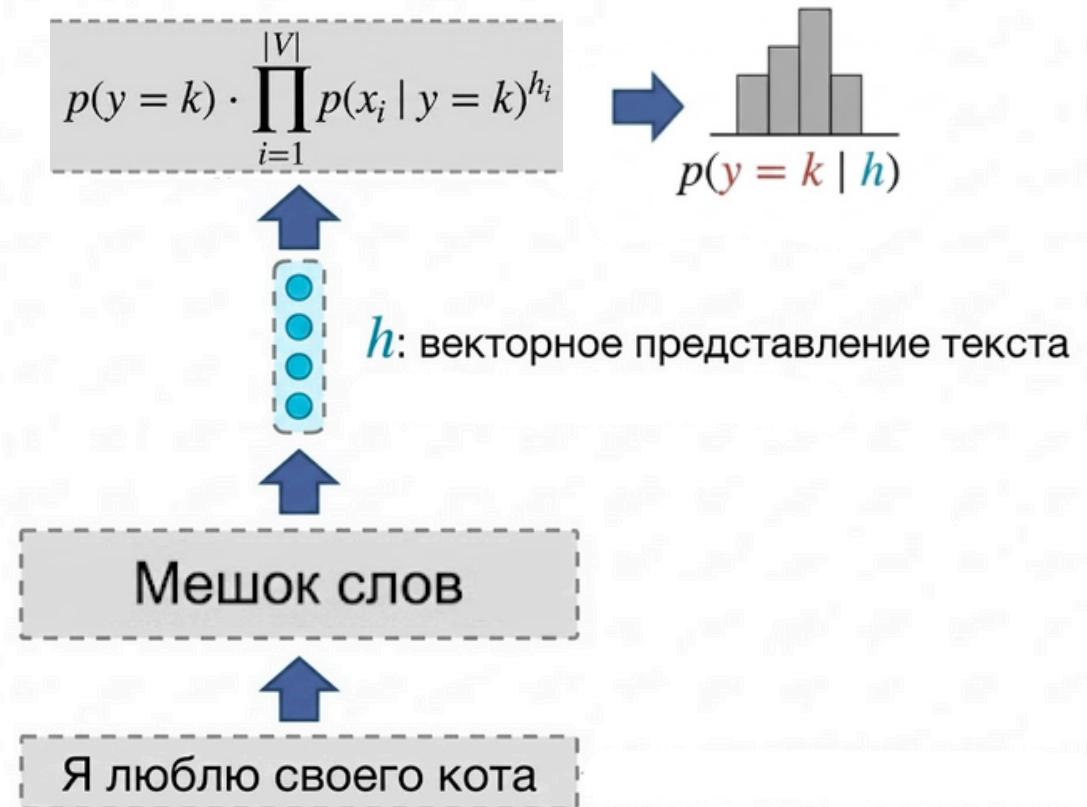
11

$$N(\text{кракозябrik}, y = +) = 0$$

$$p(\text{«очень милый кракозябrik»} \mid y = +) =$$

$$= p(\text{«очень»} \mid y = +) \cdot p(\text{«милый»} \mid y = +) \cdot p(\text{«кракозябrik»} \mid y = +) = 0$$

$$p(w \mid y = k) = \frac{N(w, y = k)}{\sum_{w' \in V} N(w', y = k)} = \frac{0}{\text{что-то положительное}} = 0$$



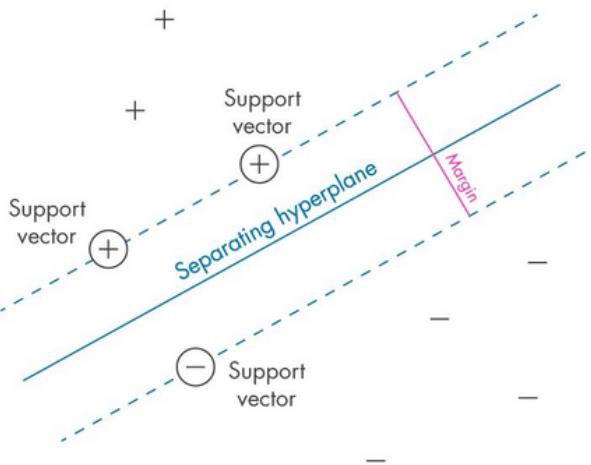


Практический вывод

13

- В тексте генеративные модели часто проще, быстрее и устойчивее на малых данных (Naive Bayes как baseline).
- Дискриминативные модели обычно дают лучшее качество при достаточных данных и правильных признаках/энкодере.
- И в обоих случаях на выходе мы приходим к одной цели: предсказать класс y по объекту x , но путь к этому разный.

- Идея: найти гиперплоскость, которая лучше всего разделяет классы и имеет максимальную маржу.
- Решение зависит только от части объектов - **support vectors** (точки у границы).
- Хорошо работает на разреженных признаках (BoW/TF-IDF) и в высоких размерностях.

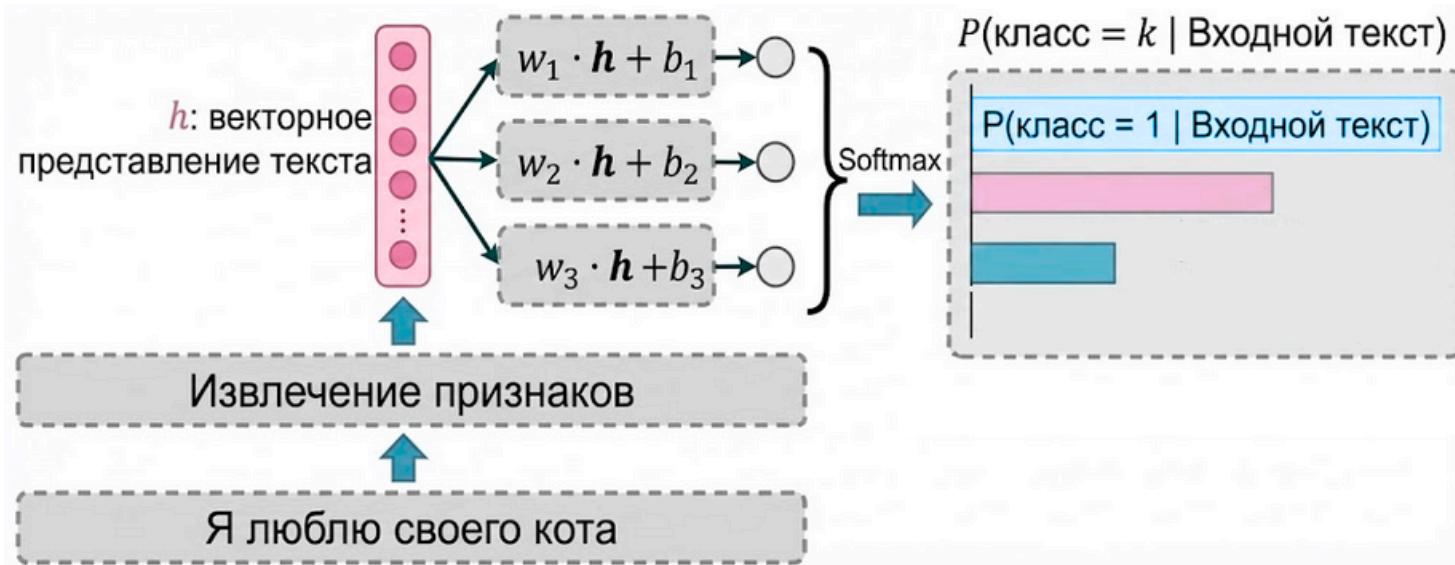


- Линейный классификатор:
 $f(x) = w^\top x + b, \quad \hat{y} = \text{sign}(f(x))$
- Оптимизация (в soft-margin варианте):
$$\min_{w,b} \frac{1}{2}|w|^2 + C \sum_i \max(0, 1 - y_i(w^\top x_i + b))$$
где $\max(0, 1 - y f(x))$ - hinge loss, C - компромисс "маржа vs ошибки".



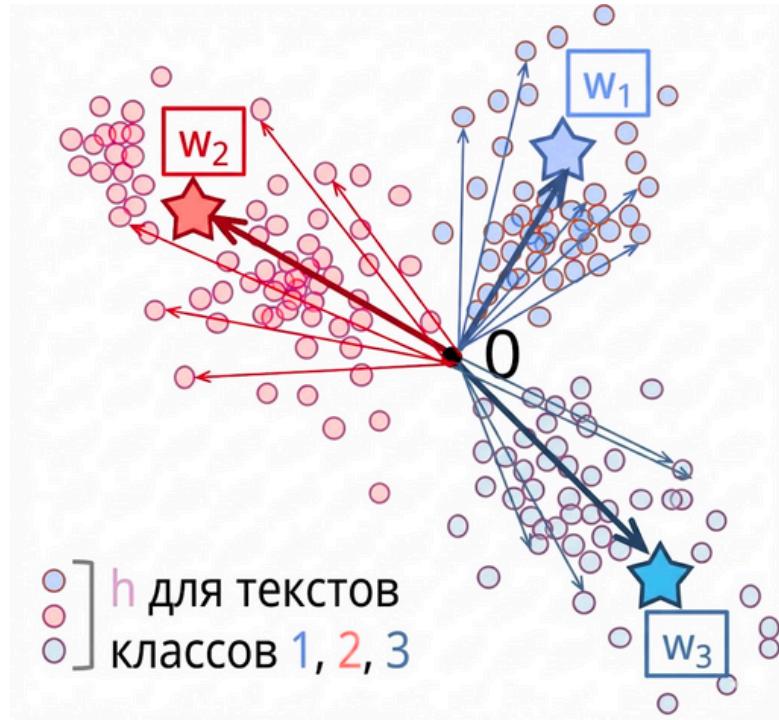
Logistic Regression

15



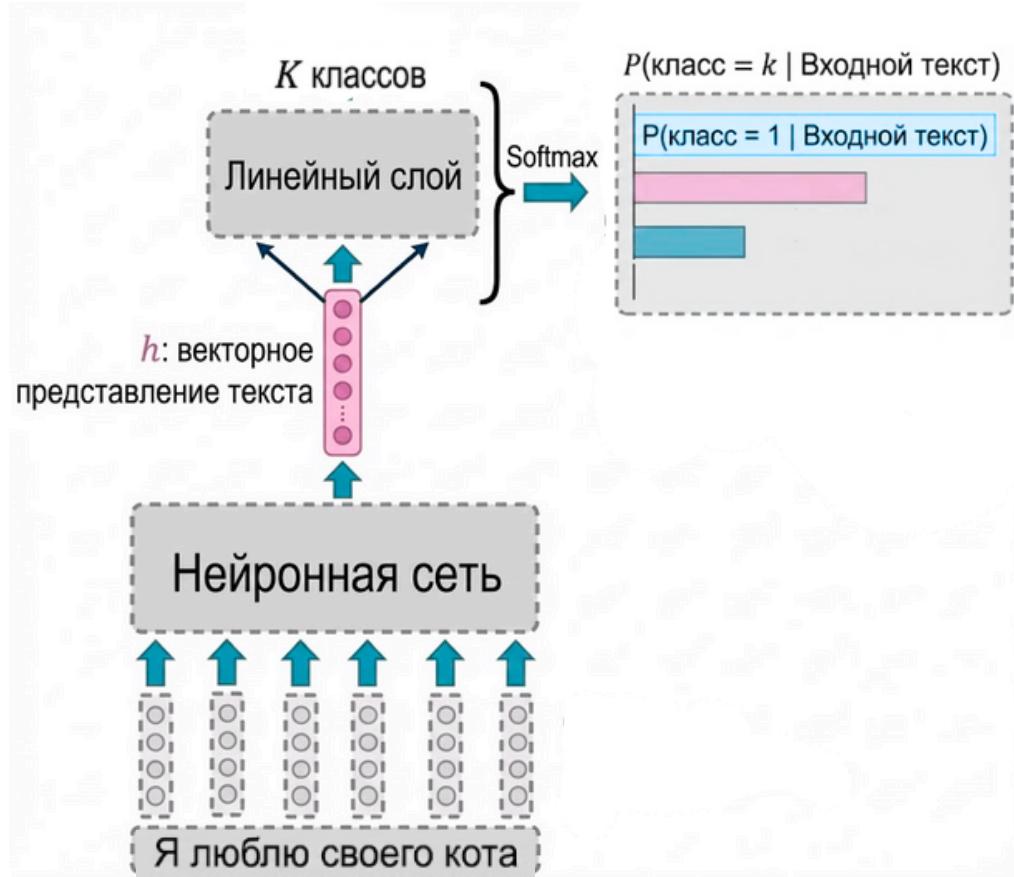
Интуиция последнего линейного слоя в классификаторе текста

- **Точки** - это векторные представления текстов h (после энкодера/извлечения признаков).
- **Векторы** - это столбцы матрицы последнего слоя W (то есть “векторы классов”).
- **Логит** для класса k считается как скалярное произведение.
- Чем сильнее h смотрит в сторону w_k (больше косинус/проекция), тем больше z_k и тем **вероятнее** выбран класс k .





Нейросетевые модели

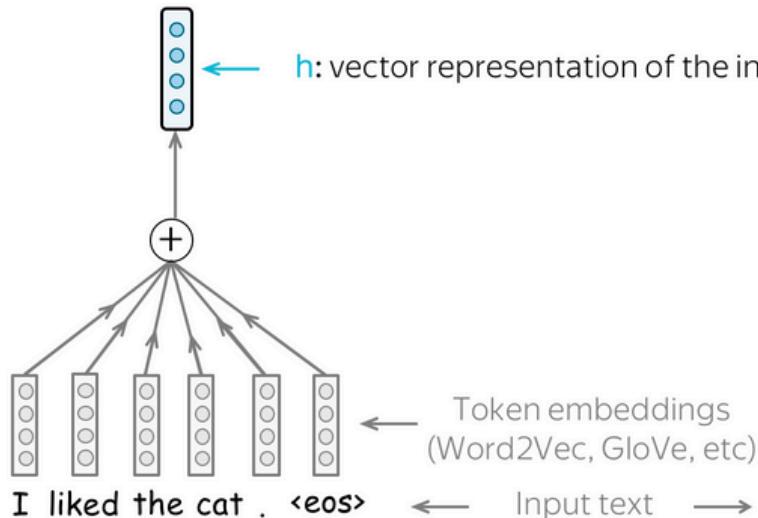




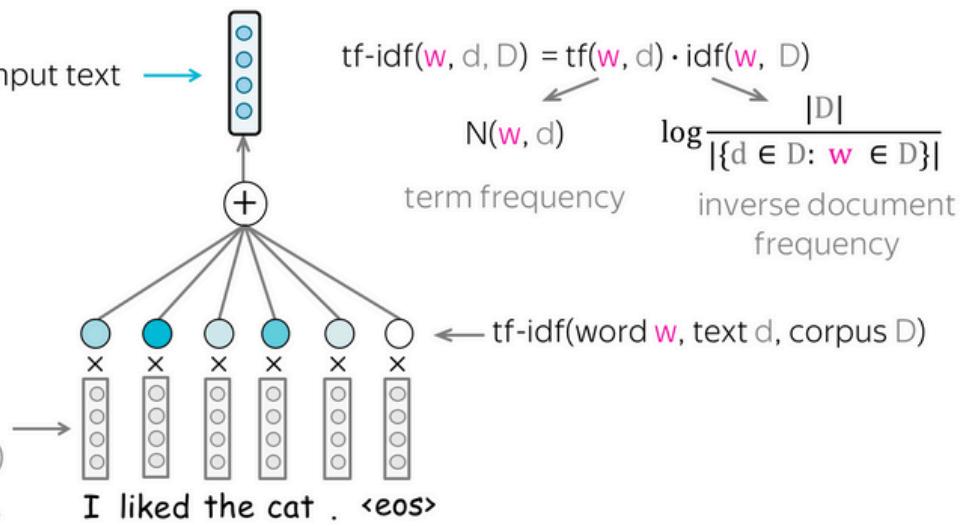
Bag of Embeddings (BOE) and Weighted BOE

18

Sum of embeddings
(Bag of Words, Bag of Embeddings)

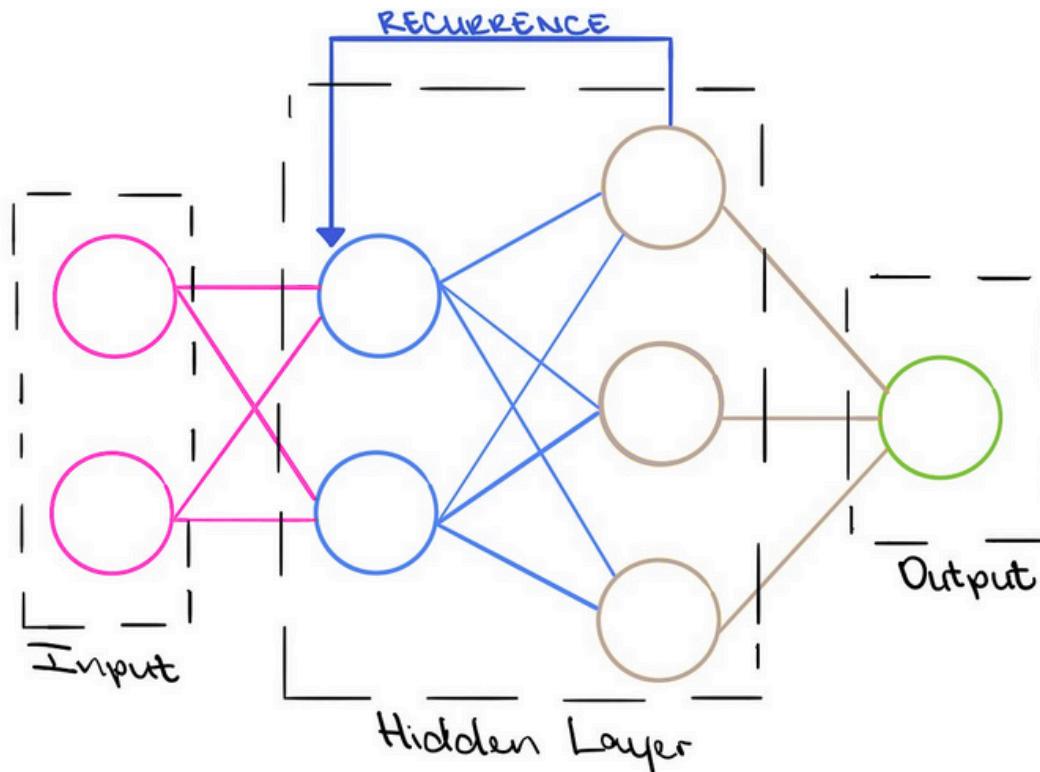


Weighted sum of embeddings
(e.g., using tf-idf weights)





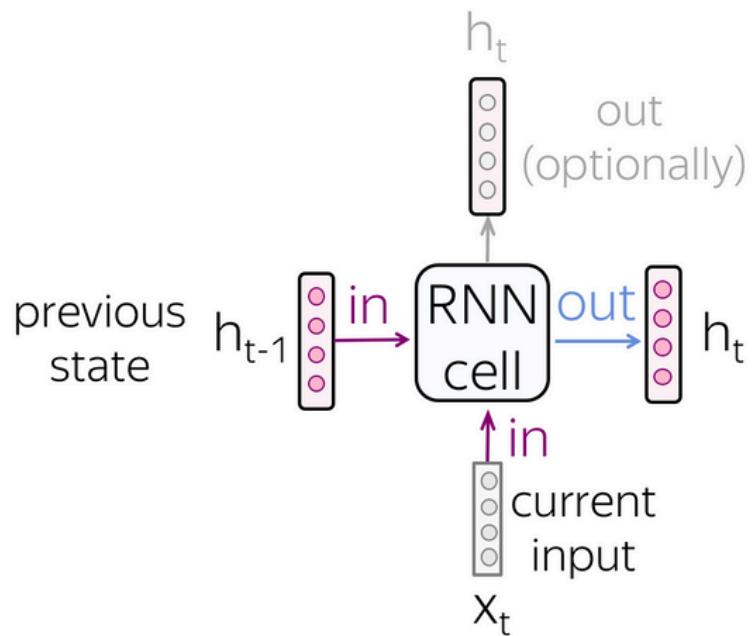
Recurrent Neural Networks





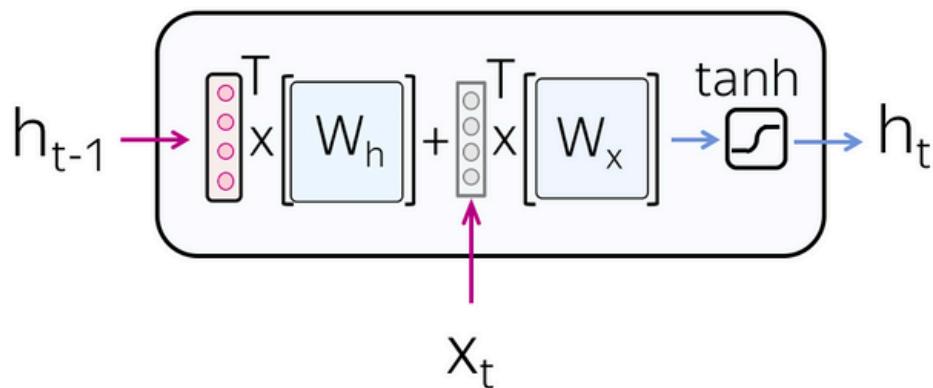
Recurrent Neural Networks

20 |



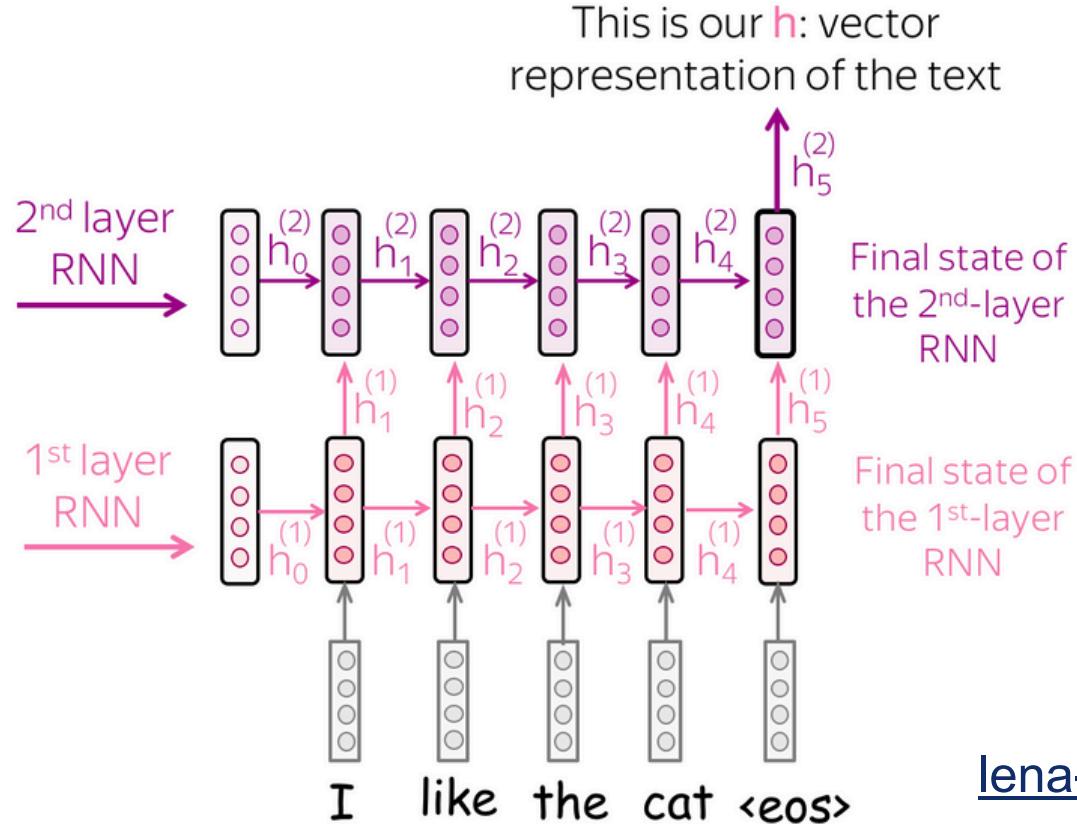
Vanilla RNN

$$h_t = \tanh(h_{t-1}W_h + x_tW_x)$$

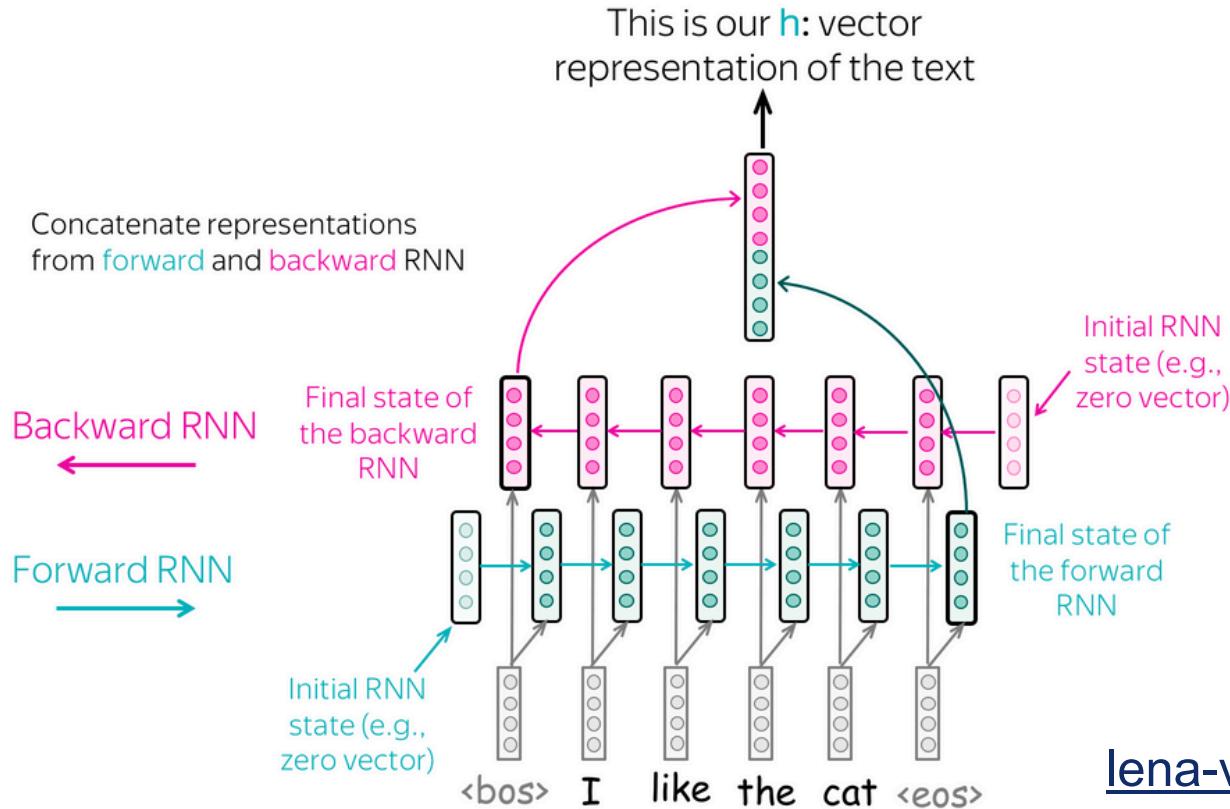


lena-voita.github.io

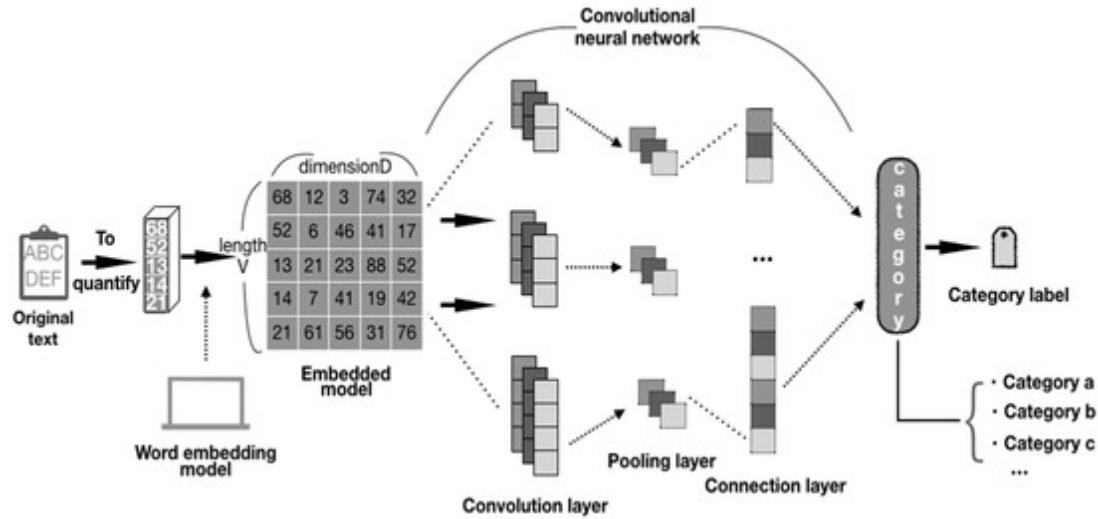
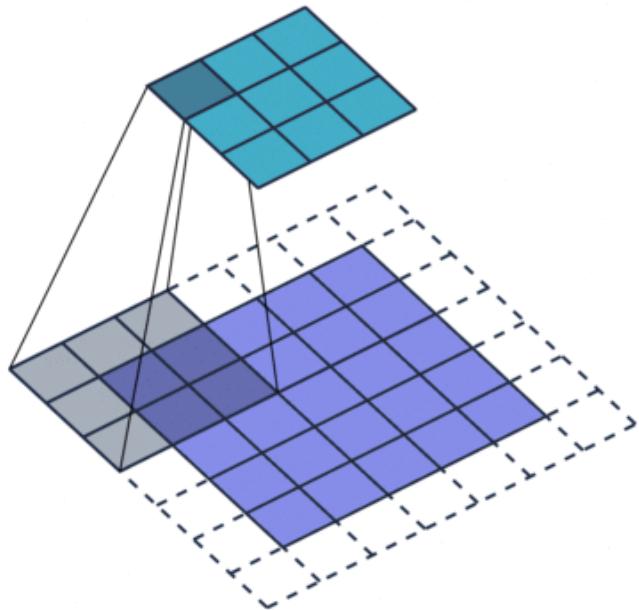
Recurrent Neural Networks



Recurrent Neural Networks



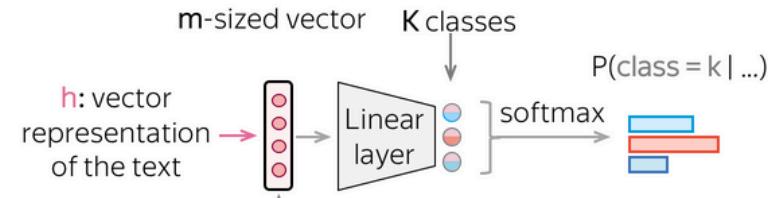
Convolutional - CNN



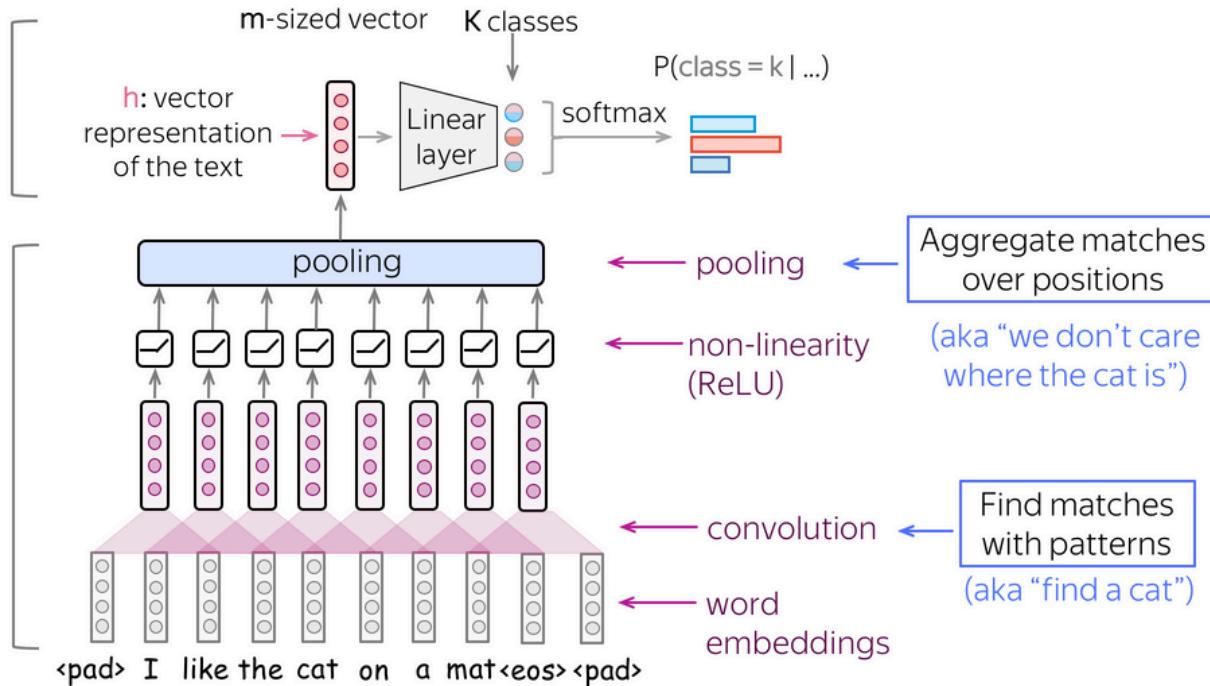
Convolution arithmetic

Convolutional - CNN

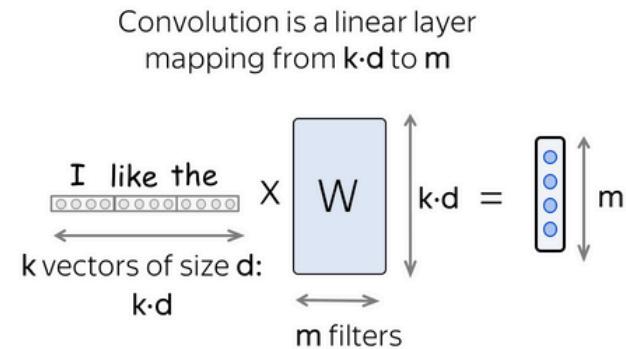
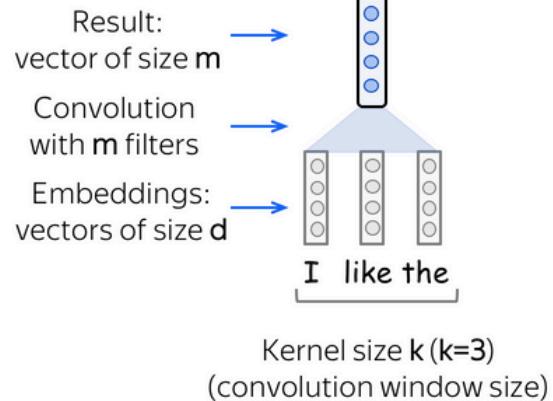
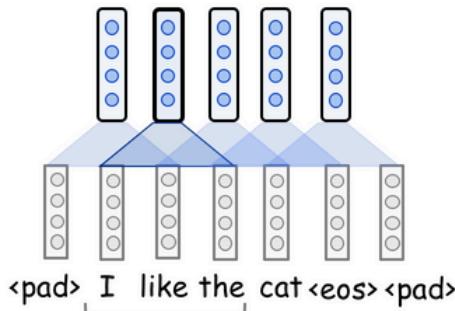
Standard part
(same for all NNs):
get probability
distribution



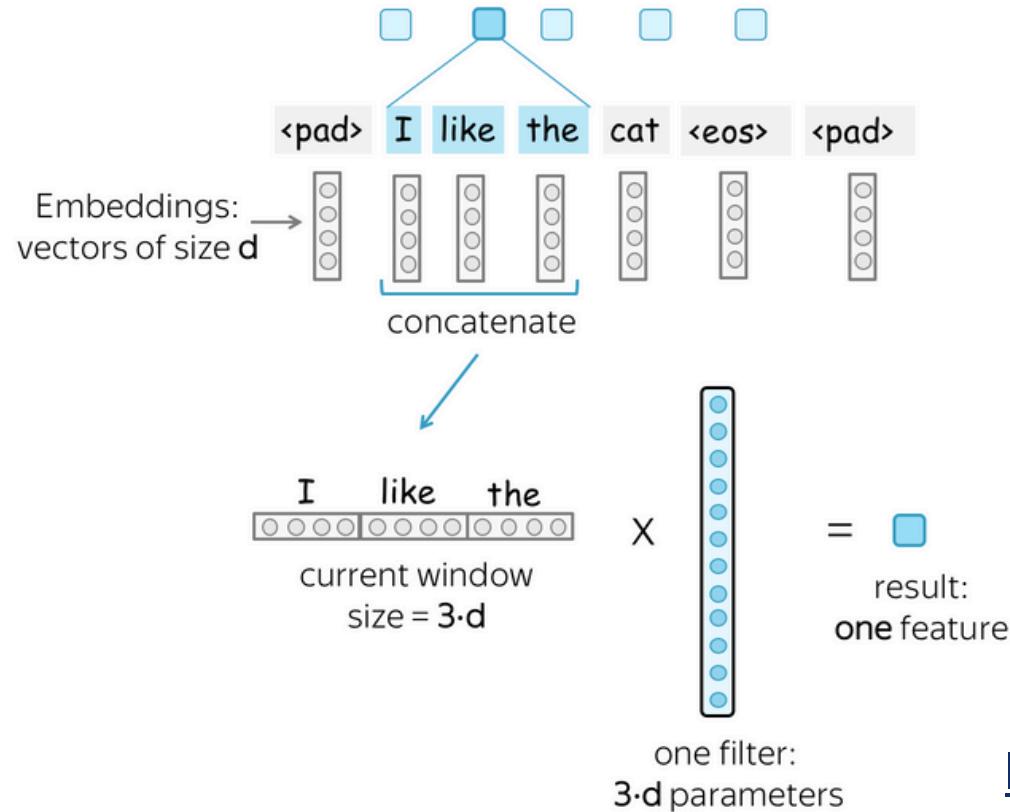
Specific to CNNs:
process text
(document)



Convolutional - CNN



Convolutional - CNN



Convolutional - CNN

