Классификация текстов

7 декабря 2023

Вспомним про Word2Vec

Предобработка текста

Перед тем, как запускать извлечение признаков из текста, его нужно предварительно подготовить - сделать пригодным для обработки алгоритмами ML. Для этого необходимо выполнить над текстом следующие операции:

1. **Токенизация** – разбиение длинных участков текста на более мелкие (абзацы, предложения, слова). Токенизация – это самый первый этап обработки текста.
2. **Нормализация** – приведение текста к «рафинированному» виду (единый регистр слов, отсутствие знаков пунктуации, расшифрованные сокращения, словесное написание чисел и т.д.). Это необходимо для применения унифицированных методов обработки текста. Отметим, что в случае текста термин «нормализация» означает приведение слов к единообразному виду, а не преобразование абсолютных величин к единому диапазону.
3. **Стеммизация** – приведение слова к его корню путем устранения придатков (суффикса, приставки, окончания).
4. **Лемматизация** – приведение слова к смысловой канонической форме слова (инфинитив для глагола, именительный падеж единственного числа — для существительных и прилагательных). Например, «зарезервированный» — «резервировать», «грибами» — «гриб», «лучший» — «хороший».
5. **Чистка** – удаление стоп-слов, которые не несут смысловой нагрузки (артикли, междометья, союзы, предлоги).

Стемминг и лемматизация

Form	Stem	Lemma
Studies	Studi	Study
Studying	Study	Study
beautiful	beauti	beautiful
beautifully	beauti	beautifully

Что должно получиться

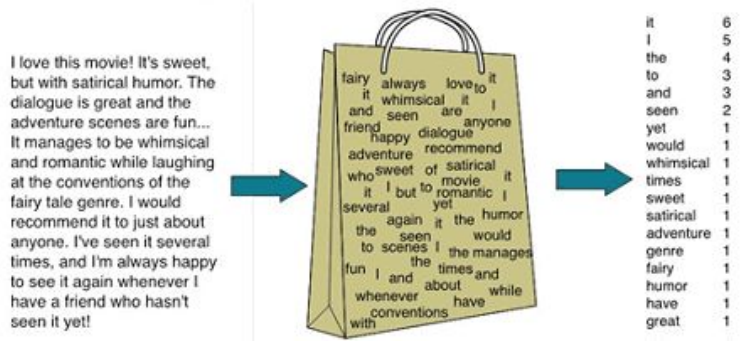
Raw Text	Pre-processed Text
Stuning even for the non-gamer: This sound track was beautiful! It paints the senery in your mind so well I would recomend it even to people who hate video game music! I have played the game Chrono Cross but out of all of the games I have ever played it has the best music! It backs away from crude keyboarding and takes a fresher step with grate guitars and soulful orchestras. It would impress anyone who cares to listen! ^_^	['stun', 'even', 'sound', 'track', 'beautiful', 'paint', 'senery', 'mind', 'well', 'would', 'recomend', 'even', 'people', 'hate', 'video', 'game', 'music', 'play', 'game', 'chrono', 'cross', 'game', 'ever', 'play', 'best', 'music', 'back', 'away', 'crude', 'keyboarding', 'take', 'fresh', 'step', 'grate', 'guitar', 'soulful', 'orchestra', 'would', 'impress', 'anyone', 'care', 'listen']

Векторизация: bag of words

«мешок слов» (bag of words) – детальная репрезентативная модель для упрощения обработки текстового содержания. Она не учитывает грамматику или порядок слов и нужна, главным образом, для определения количества вхождений отдельных слов в анализируемый текст.

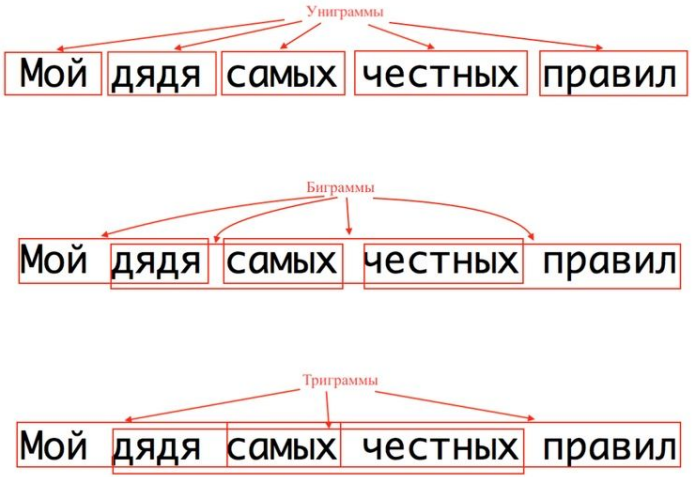
На практике bag of words реализуется следующим образом: создается вектор длиной в словарь, для каждого слова считается количество вхождений в текст и это число подставляется на соответствующую позицию в векторе. Однако, при этом теряется порядок слов в тексте, а значит, после векторизации предложения, к примеру, «i have no cats» и «no, i have cats» будут идентичны, но противоположны по смыслу.

Для решения этой проблемы при токенизации используются n-граммы.



Векторизация: n-gramms

n-граммы — комбинации из n последовательных терминов для упрощения распознавания текстового содержание. Эта модель определяет и сохраняет смежные последовательности слов в тексте. При этом можно генерировать n-граммы из букв, например, чтобы учесть сходство родственных слов или опечаток.



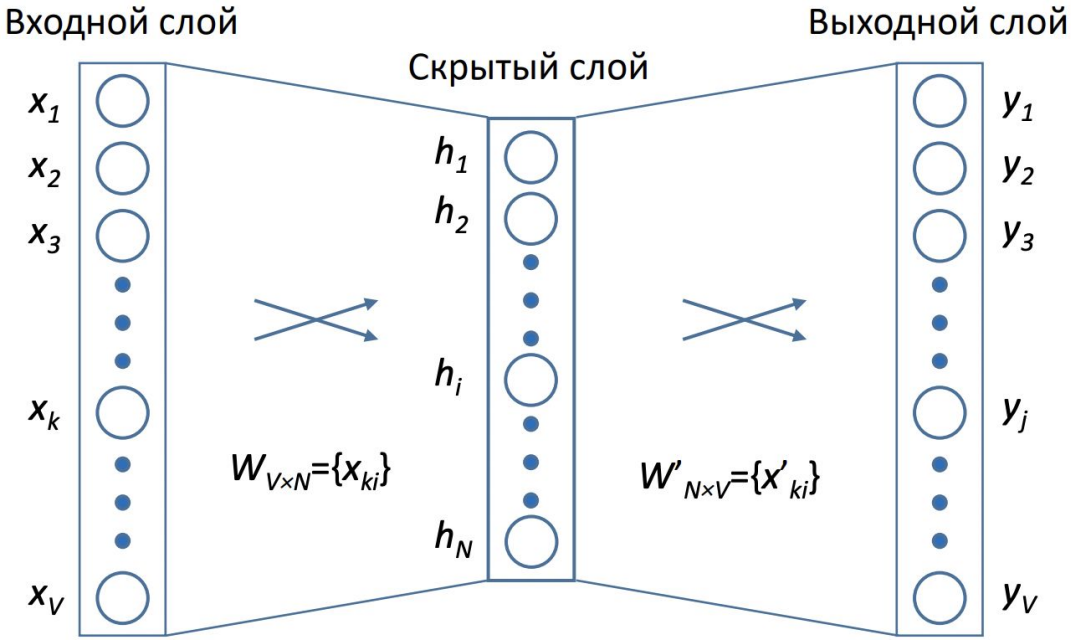
Векторизация: Word2Vec

Word2Vec — набор моделей для анализа естественных языков на основе дистрибутивной семантике и векторном представлении слов.

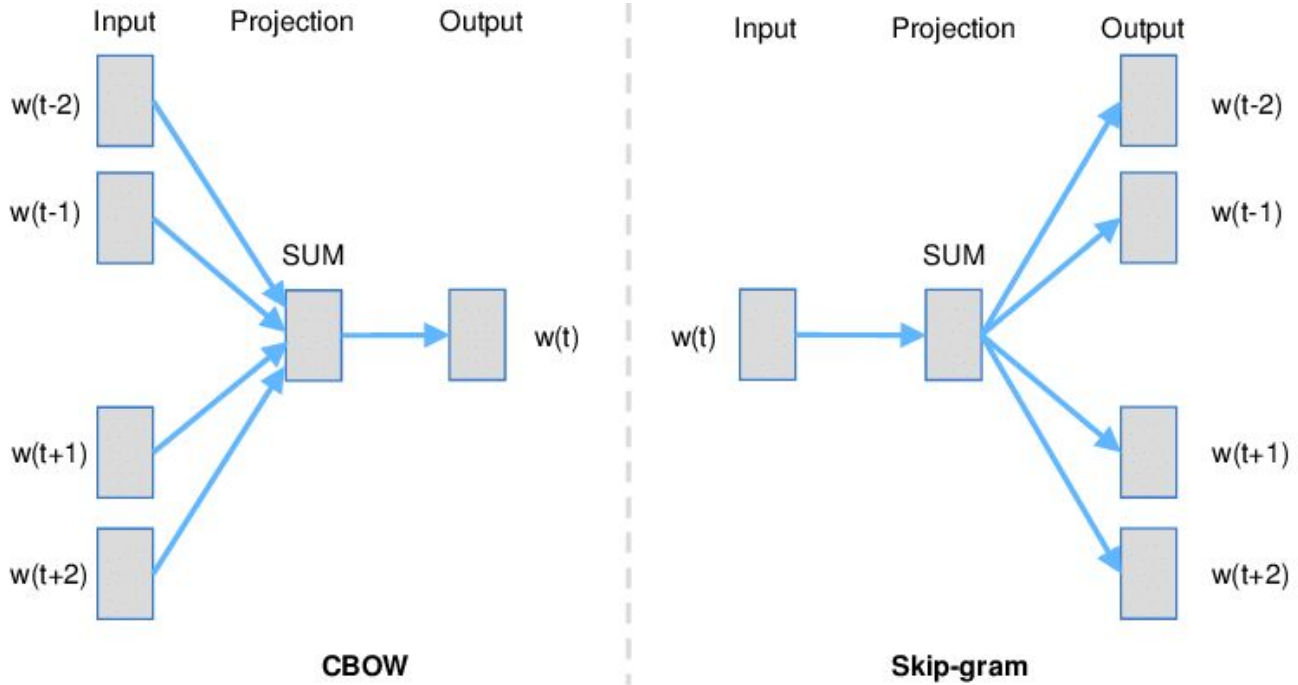
Этот метод разработан группой исследователей Google в 2013 году.

Сначала создается словарь, «обучаясь» на входных текстовых данных, а затем вычисляется векторное представление слов, основанное на контекстной близости. При этом слова, встречающиеся в тексте рядом, в векторном представлении будут иметь близкие числовые координаты. Полученные векторы-слова используются для обработки естественного языка и машинного обучения.

Обработка текста



Предсказываем следующее слово: как?



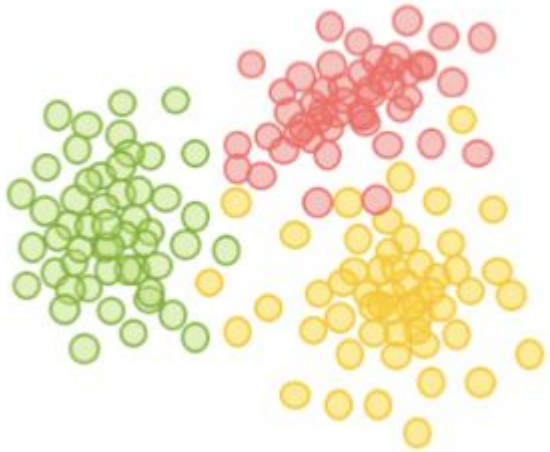
есть контекст, какое может быть слово?

есть слово, какой будет контекст?

Классификация

Что такое классификация?

- multi-class
- single-label



Примеры классификаторов

- новостная лента
- отзывы

Популярные датасеты

Dataset	Type	Number of labels	Size (train/test)	Avg. length (tokens)
SST	sentiment	5 or 2	8.5k / 1.1k	19
IMDb Review	sentiment	2	25k / 25k	271
Yelp Review	sentiment	5 or 2	650k / 50k	179
Amazon Review	sentiment	5 or 2	3m / 650k	79
TREC	question	6	5.5k / 0.5k	10
Yahoo! Answers	question	10	1.4m / 60k	131
AG's News	topic	4	120k / 7.6k	44
Sogou News	topic	6	54k / 6k	737
DBPedia	topic	14	560k / 70k	67

Популярные датасеты

https://ics.uci.edu/~smyth/courses/cs175/text_data_sets.html

Text Classification and Sentiment Analysis

Multiple text classification datasets from NLP-progress

Multiple sentiment analysis datasets from NLP-progress

Yelp Data Set Challenge (8 million reviews of businesses from over 1 million users across 10 cities)

Kaggle Data Sets with text content (Kaggle is a company that hosts machine learning competitions)

Labeled Twitter data sets from (1) the SemEval 2018 Competition and (2) Sentiment 140 project

Amazon Product Review Data from UCSD. This is a very large and rich data set with review text, ratings, votes, product metdata, etc. The full dataset is extremely large - some of the smaller subsets provided may be better for class projects.

IMDB Moview Review Data with 50,000 movie reviews and binary sentiment labels

Well-known Movie review data for sentiment analysis, from Pang and Lee, Cornell

Product review data from Johns Hopkins University (goal is to predict ratings on scale of 1 to 5)

Популярные датасеты

https://ics.uci.edu/~smyth/courses/cs175/text_data_sets.html

Dialog/Conversation/Chatbots

A repository of large datasets for models of conversational response
A survey paper on data sets available for building data-driven dialogue systems
Amazon Topical Chat Dataset with accompanying research paper and blog post from Amazon.
ConvAI2 Competition Dataset
Multiple labeled dialog/chatbot datasets from NLP-progress
Cornell Movie-Dialogs Corpus
Transcripts from the TV series "The Office" (formatted for the R language)

Language Models and Auto-complete Algorithms

Language modeling datasets from NLP-progress
Ngram data from Peter Norvig (Google), with an accompanying tutorial book chapter
Google ngrams, and Google syntactic ngrams over time, from Google books

Question-Answering Datasets

Multiple question-answering datasets from NLP-progress
WikiQA , a data set for "open-domain" question answering, from Microsoft Research
Question-Answering Data Sets from TREC (funding by the National Institute of Standards and Technology, NIST)
Question Answering Corpus from DeepMind
The Allen AI Science Challenge on Kaggle (competition ended in 2016)

Пример

Label: negative

Review
Hobgoblins .... Hobgoblins .... where do I begin?!?

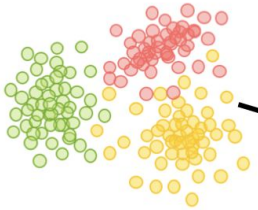
This film gives Manos - The Hands of Fate and Future War a run for their money as the worst film ever made . This one is fun to laugh at , where as Manos was just painful to watch . Hobgoblins will end up in a time capsule somewhere as the perfect movie to describe the term : " 80 's cheeze " . The acting (and I am using this term loosely) is atrocious , the Hobgoblins are some of the worst puppets you will ever see , and the garden tool fight has to be seen to be believed . The movie was the perfect vehicle for MST3 K , and that version is the only way to watch this mess . This movie gives Mike and the bots lots of ammunition to pull some of the funniest one - liners they have ever done . If you try to watch this without the help of Mike and the bots ..... God help you ! !

Классификация - два подхода

классические и нейронные

Классификация - два подхода

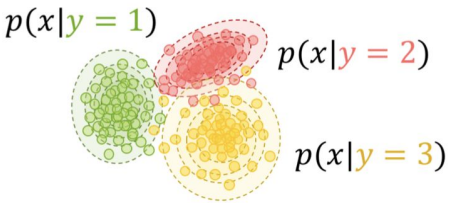
Генеративные модели



Дискриминативные модели

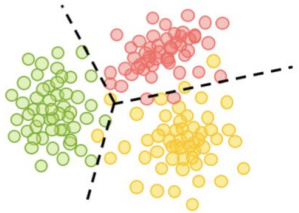
хотим найти класс, наиболее вероятный для нашего объекта

учим распределение объектов и классов



хотим найти вероятность класса, при условии заданного объекта

учим разделяющие границы



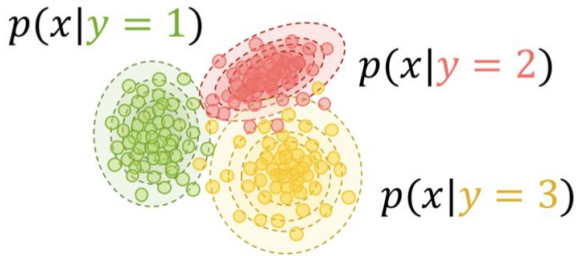
Генеративные модели

Учим распределение данных

$$p(x, y) = p(x|y) \cdot p(y)$$

Решаем задачу

$$y = \arg\max_k p(x, y) = \arg\max_k p(x|y) \cdot p(y)$$



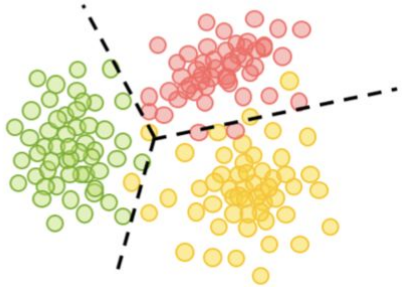
Дискриминативные модели

Учим распределение классов

$$p(y|x)$$

Ищем наиболее вероятный класс

$$y = \arg\max_k p(y|x)$$



Классические методы

Naive Bayes

Классика: наивный Байес

Вспомним формулу Байеса для условной вероятности

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

и задачу, которую мы хотим решить

$$y^* = \arg\max_k P(y = k|x)$$

Классика: наивный Байес

Bayes' rule (hence Naïve Bayes)

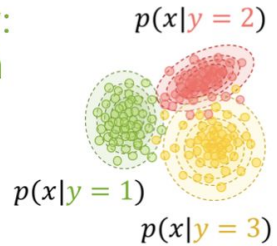
Ignore P(x) – it does not influence the argmax

$$y^* = \arg\max_k P(y = k|x) = \arg\max_k \frac{P(x|y = k) \cdot P(y = k)}{P(x)} = \arg\max_k P(x|y = k) \cdot P(y = k)$$

Классика: наивный Байес

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

posterior probability: after looking at data (i.e., we know x)



prior probability: before looking at data (i.e., we don't know x)

joint probability

Классика: наивный Байес

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

нужно определить

$$P(y = k) = \frac{N(y = k)}{\sum_{i=1}^{K} N(y = i)}$$

расчет вероятности конкретного класса очень прост: берем кол-во элементов рассматриваемого класса и делим на кол-во всех элементов

$$P(x|y = k) = P(x_1, x_2, \dots, x_n|y = k)$$

как нам рассчитать вероятность принадлежности элемента классу? дискуссионный вопрос. допустим мы разобьем текст на токены и посчитаем вероятность каждого токена в классе, но что для этого нужно?

Классика: наивный Байес

Вводим два предположения:

1) Bag of Words - порядок слов не важен.
2) Conditional Independence assumption - вероятность появления слова в классе не зависит от других слов в этом классе.

Решение:

$$P(x|y=k) = P(x_1, x_2, \dots, x_n|y=k)$$

$$= \prod_{i=1}^{n} P(x_i|y=k)$$

Классика: наивный Байес

Пример

$P(\text{Это классный фильм!} \mid y = +) =$

$$P(\text{Это} \mid y = +) \cdot P(\text{классный} \mid y = +) \cdot P(\text{фильм} \mid y = +) \cdot P(! \mid y = +)$$

Классика: наивный Байес

$$\prod_{i=1}^{n} P(x_i|y=k)$$ как это считать?

считаем вероятность слова, делим на количество всех слов в классе - отлично!

$$P(x_i|y=k) = \frac{N(x_i, y=k)}{\sum_{t=1}^{|V|} N(x_t, y=k)}$$

Но всегда есть “но”

$$P(x_i|y=k) = \frac{N(x_i, y=k)}{\sum_{t=1}^{|V|} N(x_t, y=k)}$$

что если слово не встречается в классе?
вероятность обнулится

$$\prod_{i=1}^{n} P(x_i|y=k)$$

но тогда обнулится произведение!

Уходим от обнуления

Сглаживание Лапласа (add-one)

$$P(x_i|y=k) = \frac{\delta + N(x_i, y=k)}{\sum_{t=1}^{|V|}(\delta + N(x_t, y=k))} = \frac{\delta + N(x_i, y=k)}{\delta \cdot |V| + \sum_{t=1}^{|V|} N(x_t, y=k)}$$

Почему это работает?

Data: $x =$ This film is awesome !
x_1 x_2 x_3 x_4 x_5

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

Positive class

$$P(x, y = +)$$
$$= P(y = +) \cdot P(x|y = +)$$
$$= P(y = +) \cdot$$
$$\cdot P(\text{This}|y = +)$$
$$\cdot P(\text{film}|y = +)$$
$$\cdot P(\text{is}|y = +)$$
$$\cdot P(\text{awesome}|y = +)$$
$$\cdot P(!|y = +)$$

Negative class

$$P(x, y = -)$$
$$= P(y = -) \cdot P(x|y = -)$$
$$= P(y = -) \cdot$$
$$\cdot P(\text{This}|y = -)$$
$$\cdot P(\text{film}|y = -)$$
$$\cdot P(\text{is}|y = -)$$
$$\cdot P(\text{awesome}|y = -)$$
$$\cdot P(!|y = -)$$

априорная вероятность 0.5

часто встречающиеся слова ничего не дают

слово-маркер! ура

$$P(\text{awesome}|y = +) \gg P(\text{awesome}|y = -)$$

Когда наивный Байес не будет работать?

Фильм не слишком плохой, хороший.

Фильм не слишком хороший, плохой.

Одинаковый мешок слов.

Logistic regression

Логистическая регрессия

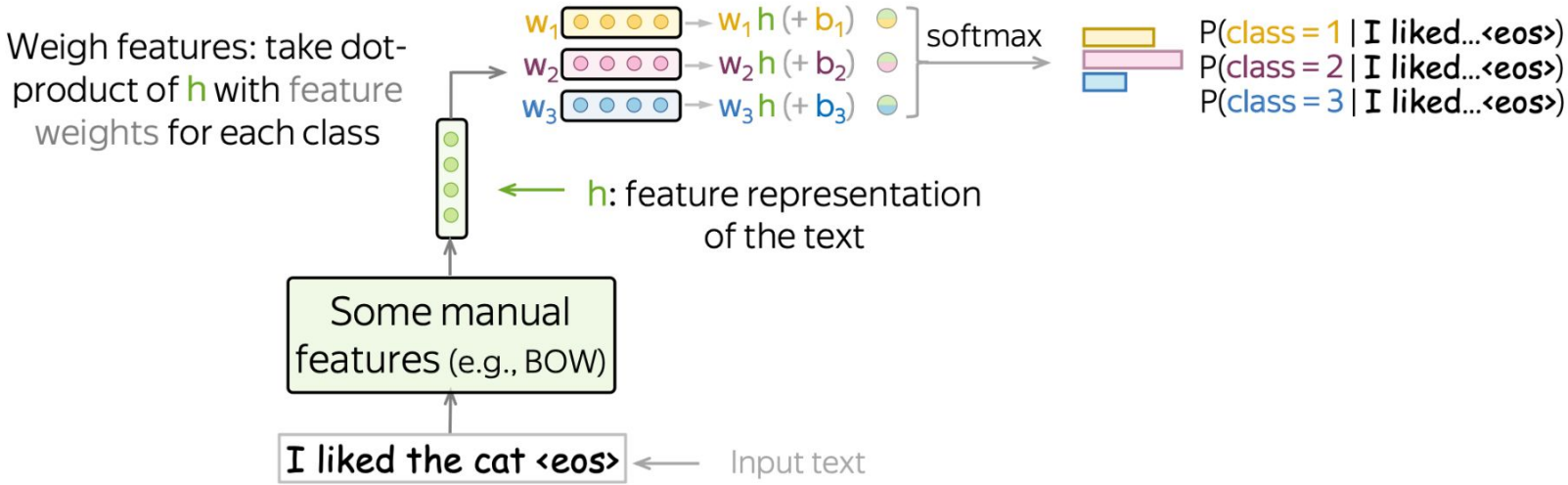
Пусть у нас есть набор фич f_i

есть веса фич для каждого класса w_i^k

$$w^{(k)} h = w_1^{(k)} \cdot f_1 + \cdots + w_n^{(k)} \cdot f_n, \qquad k = 1, \ldots, K$$

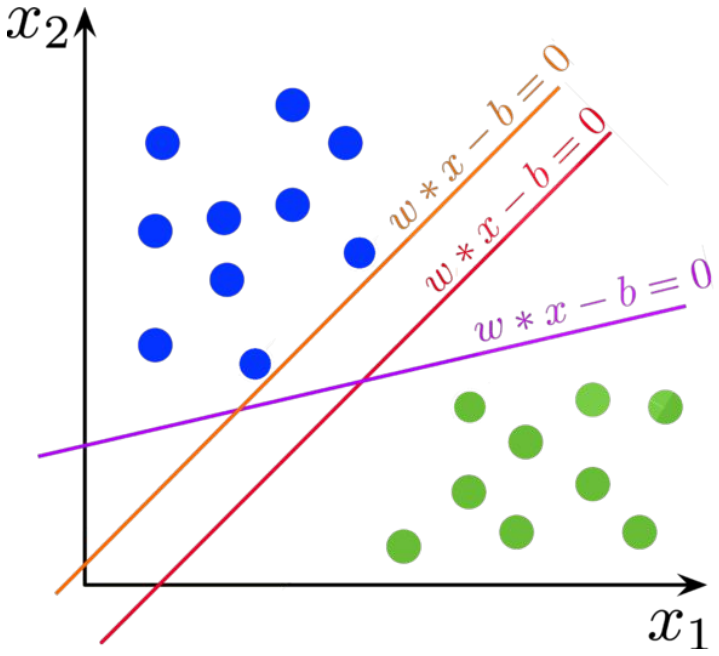
$$P(class = k|h) = \frac{\exp(w^{(k)} h)}{\sum_{i=1}^{K} \exp(w^{(i)} h)}$$

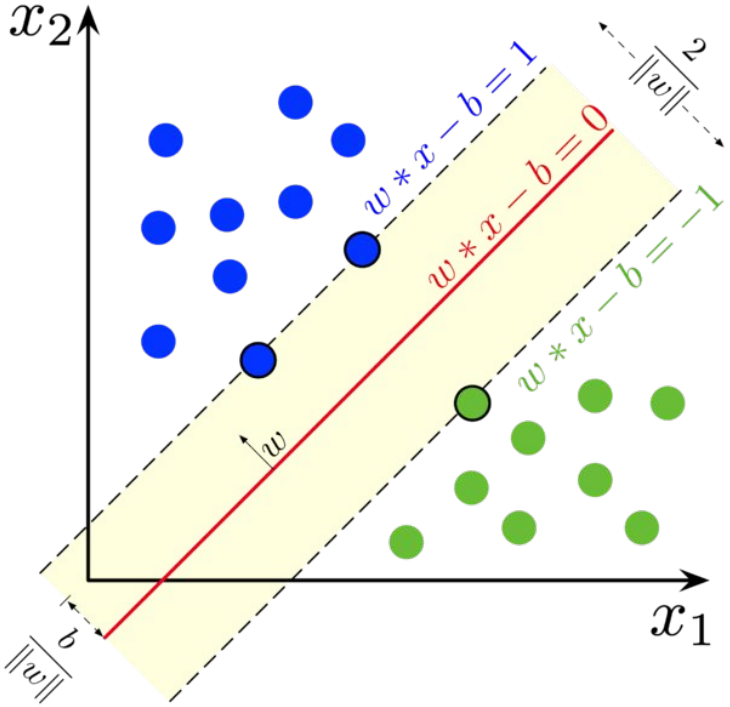
Логистическая регрессия



Support Vector Machine

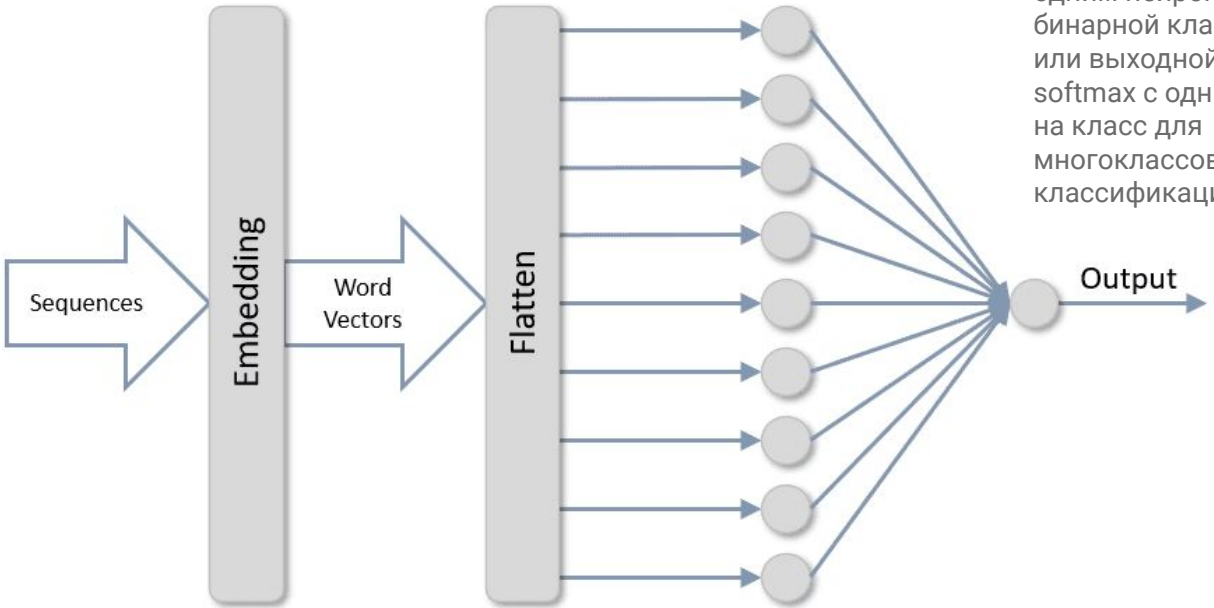
Метод опорных векторов SVM





Нейронные сети

Нейронные сети

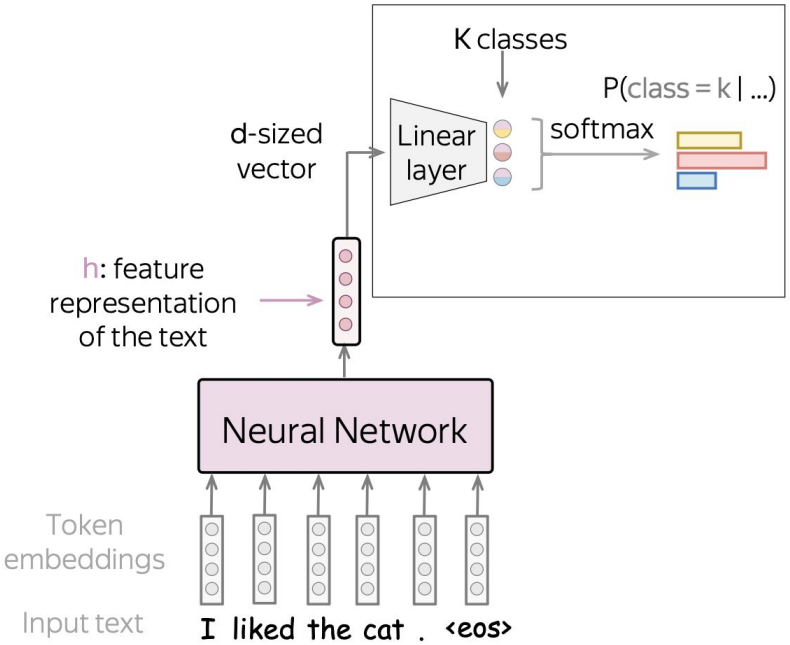


Выходной sigmoid-слой с одним нейроном для бинарной классификации или выходной слой softmax с одним нейроном на класс для многоклассовой классификации

Embedding *преобразует текст в массивы или последовательности скалярных значений, массивы векторов слов , которые кодируют информацию об отношениях между словами*

Flatten «сглаживает» 2D-массивы, выводимые слоем embedding, в 1D-массивы, которые можно вводить в плотный слой

Нейронные сети



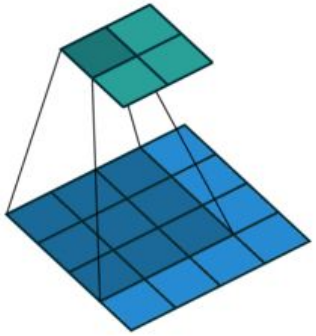
На самом деле, получаем то же самое, что и для логистической регрессии, единственное - выбор фич не руками. Нейронная сеть сама выбирает фичи, которые, как правило, не интерпретируемы.

CNN

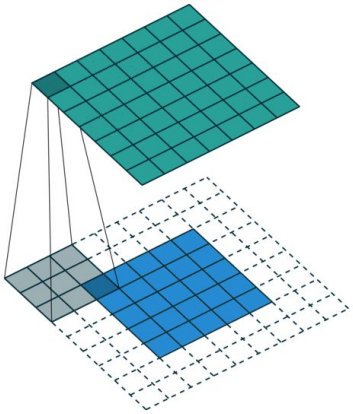
CNN: как работают свертки

Stride - определяет шаг фильтра по карте признаков (или изображению, если это первый слой). Если stride=1, то это значит, что фильтр будет помещен в каждое положение на карте признаков и из исходной карты признаков размером будет получена карта признаков того же размера.

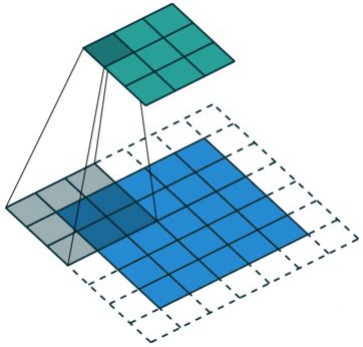
Padding - определяет насколько надо расширить карту признаков перед выполнением свертки. Как правило, это нужно, чтобы посчитать значение свертки с фильтром в положении на границе карты признаков. Есть несколько вариантов чем расширять карту признаков, самый частый вариант это zero padding, т.е. расширение карты признаков нулями.



no paddings,
no strides



full padding,
no strides



padding,
strides

CNN: pooling

Свёрнутое изображение					
1	0	4	2	125	67
8	2	5	4	34	12
20	13	25	15	240	2
76	8	6	6	100	76
34	66	134	223	201	3
255	123	89	55	32	2

Подвыборка
3x3
Размер шага - 3

Новое изображение	
25	

CNN: pooling

An absolutely great movie! I watched the premiere with my friends.

The movie about cats was absolutely great, and the cats were cute.

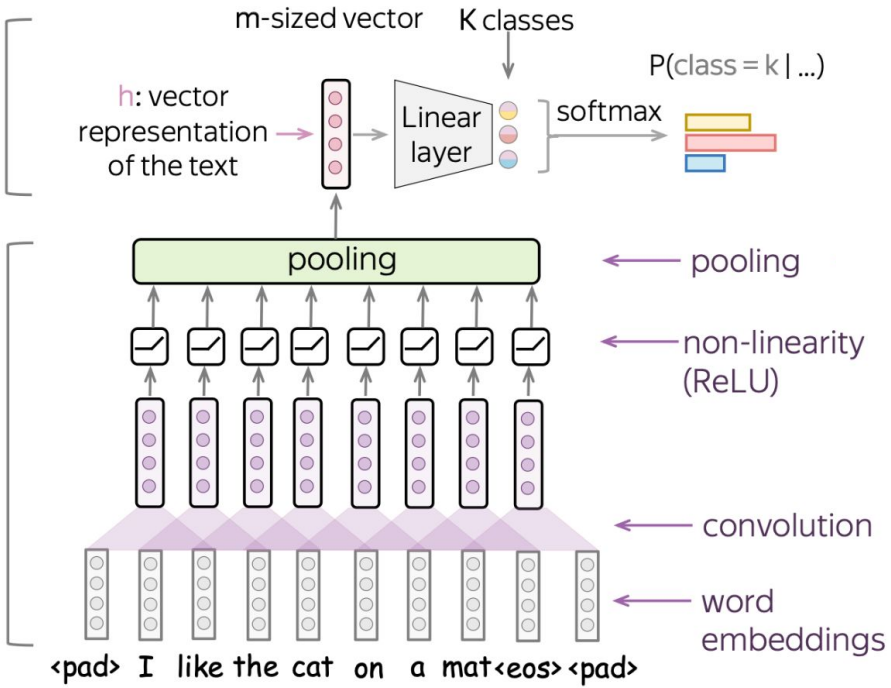
The movie is about cats running around, and it is absolutely great.

Не важно где слово, главное - оно есть.

CNN: резюме

стандартная нейронка

специфика свертки

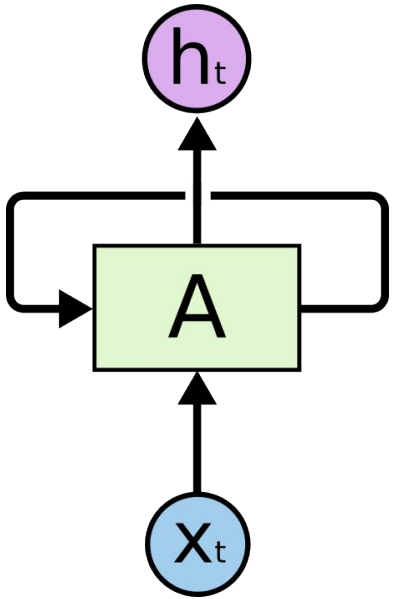


RNN

RNN: рекуррентные нейронные сети

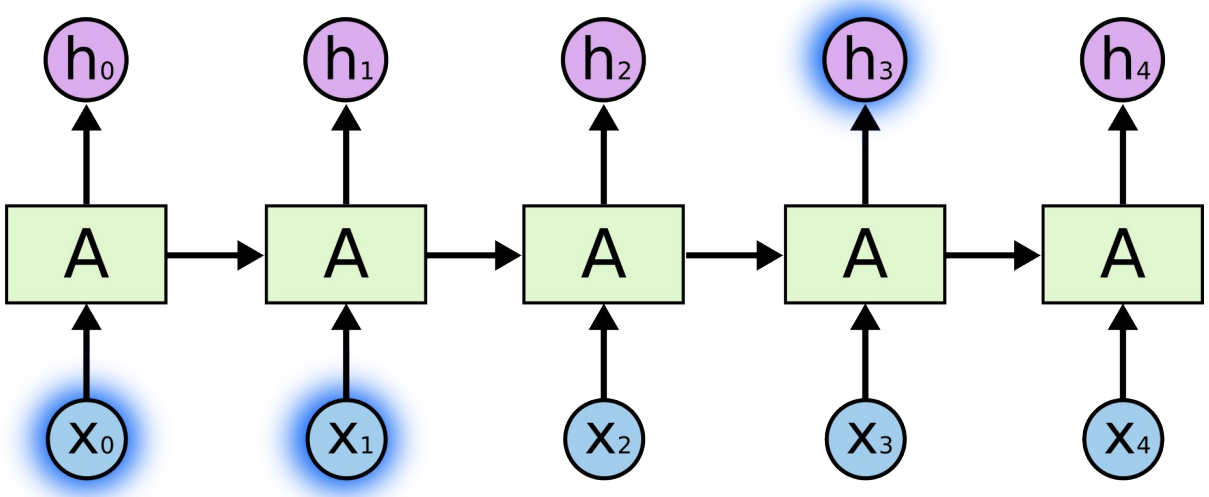
Хранят информацию с предыдущего слоя, из-за чего возникает эффект “запоминания”.

Логично предположить, что должны лучше подойти для текстов (мы помним контект).



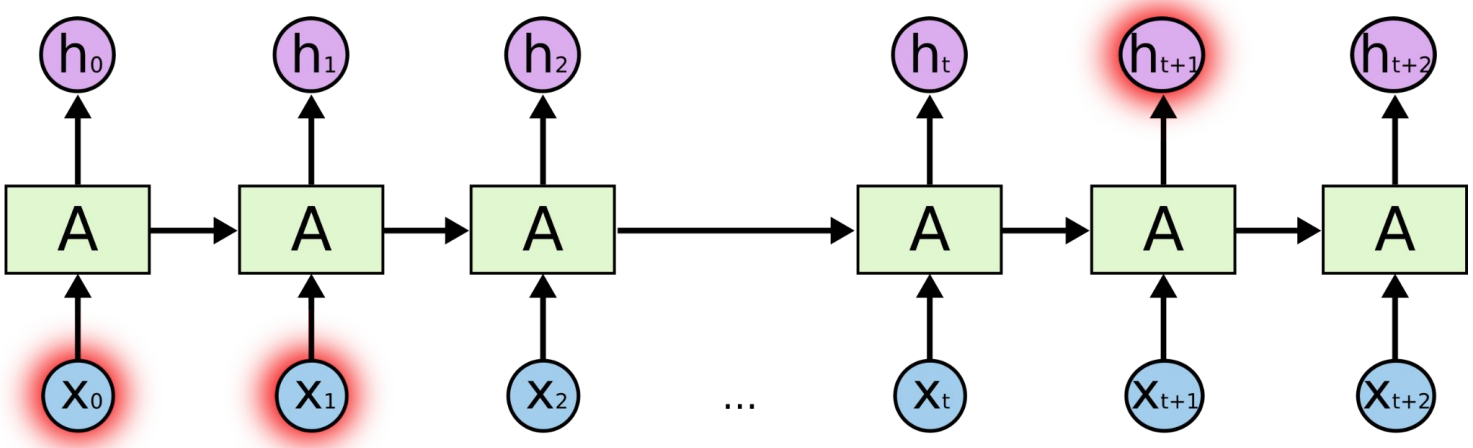
RNN: работает хорошо

Попробуем предсказать: **облака в** *небе*

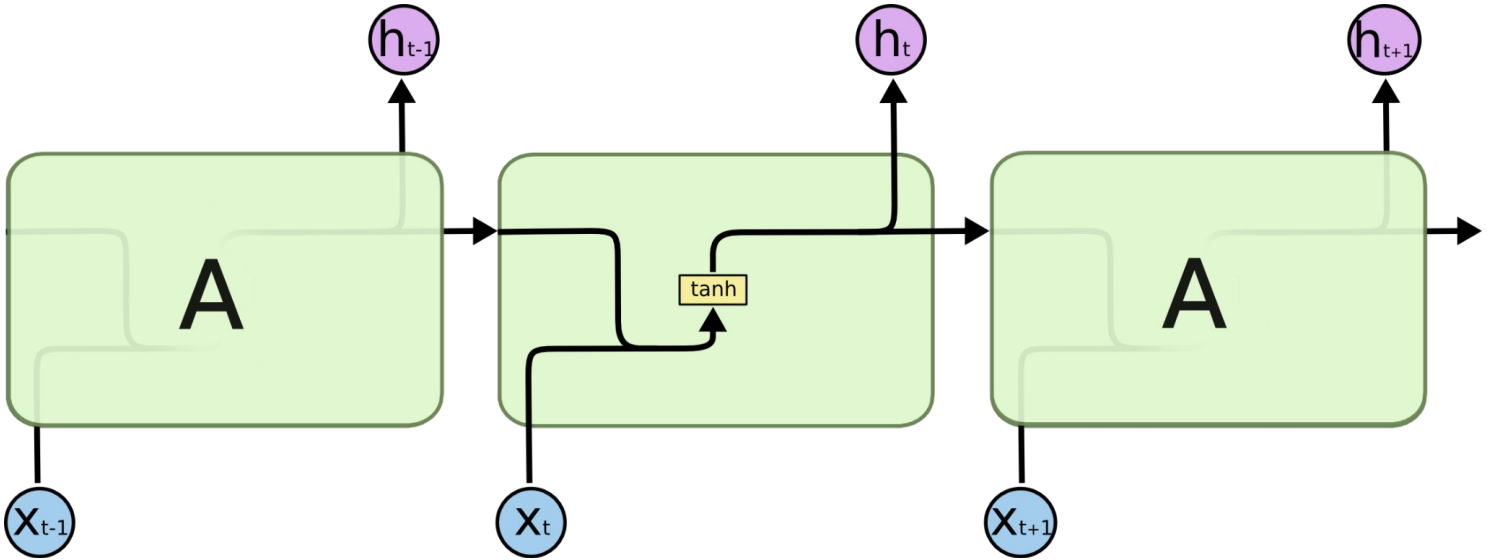


RNN: работает не очень

Попробуем предсказать: «**Я вырос во Франции... Я свободно говорю** *по-французски*»



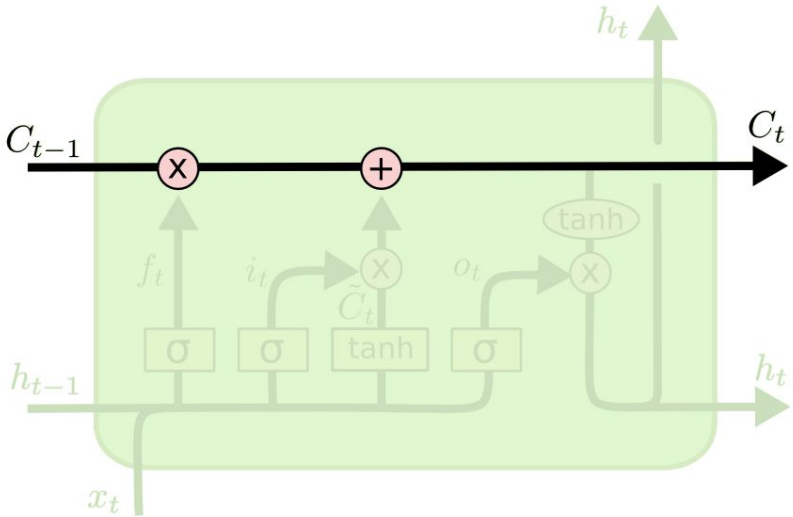
Решение: LSTM - долгосрочная краткосрочная память

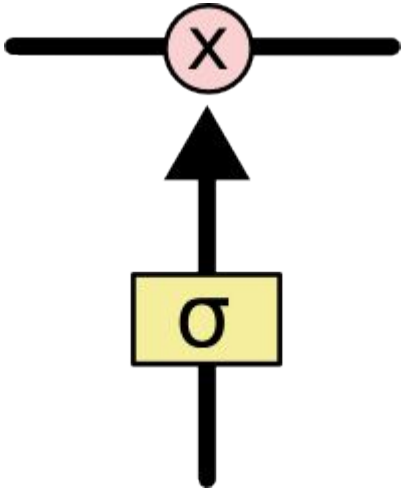


Как это работает?

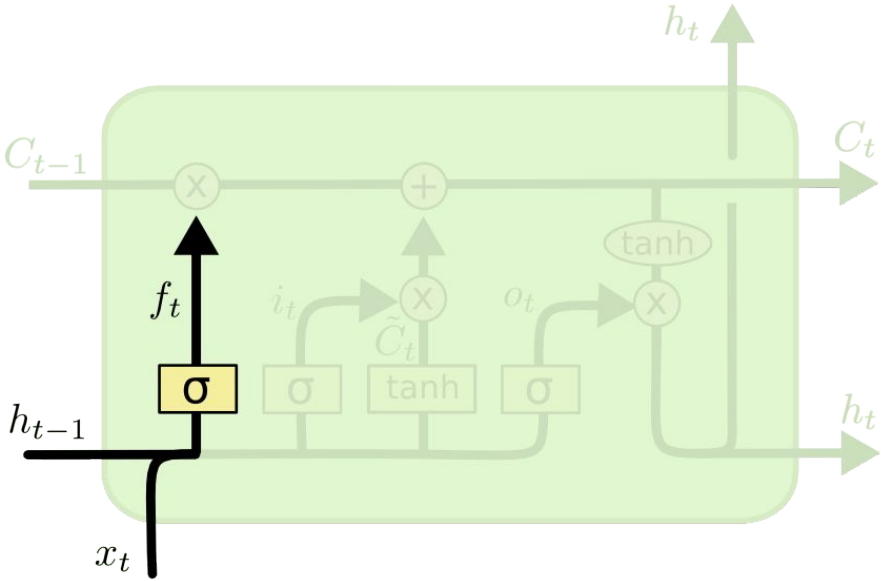
LSTM: основная идея

Конвейерная лента с контролируемыми выходами: решаем нужна ли нам информация.





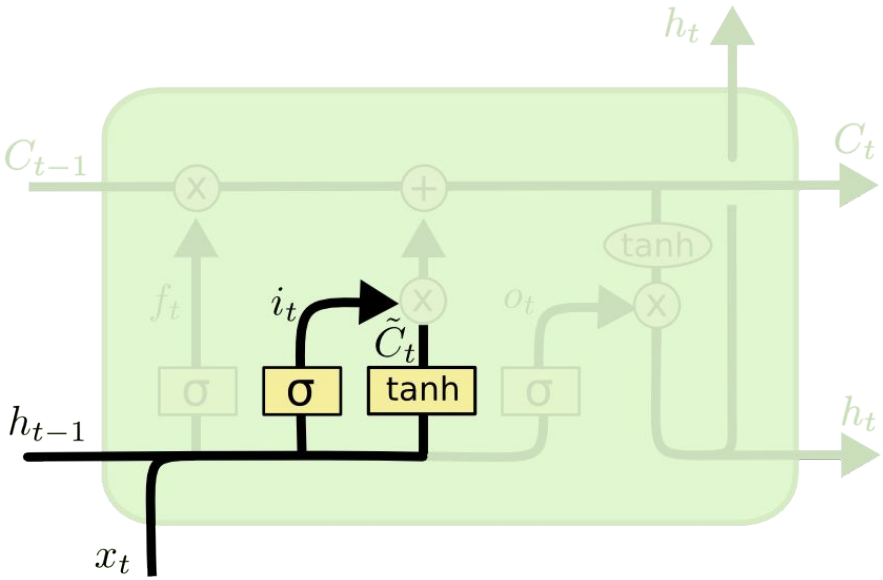
LSTM: первый шаг



“forget gate layer”

решаем какую информацию хотим забыть

LSTM: второй шаг

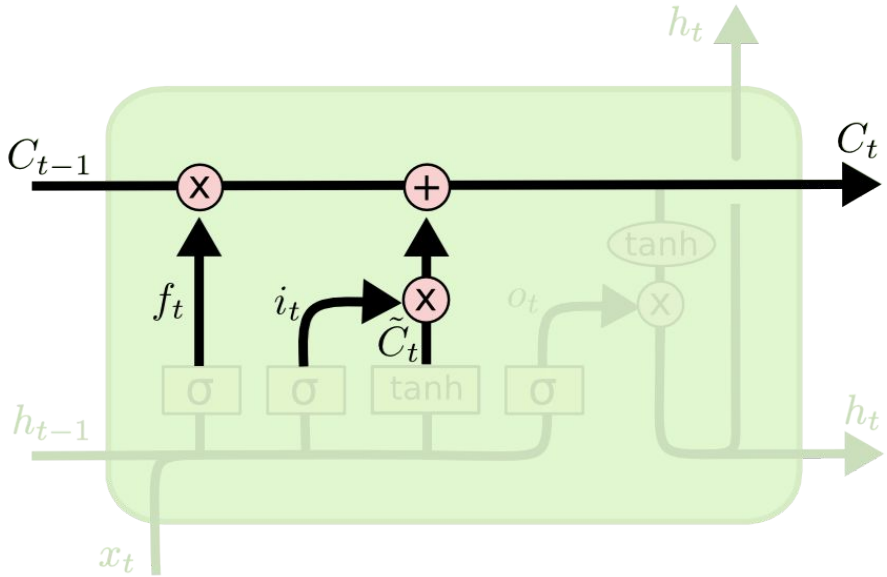


решаем, какую новую информацию мы будем хранить в состоянии ячейки

“input gate layer”
сигмоидный слой решает какие значения мы обновим

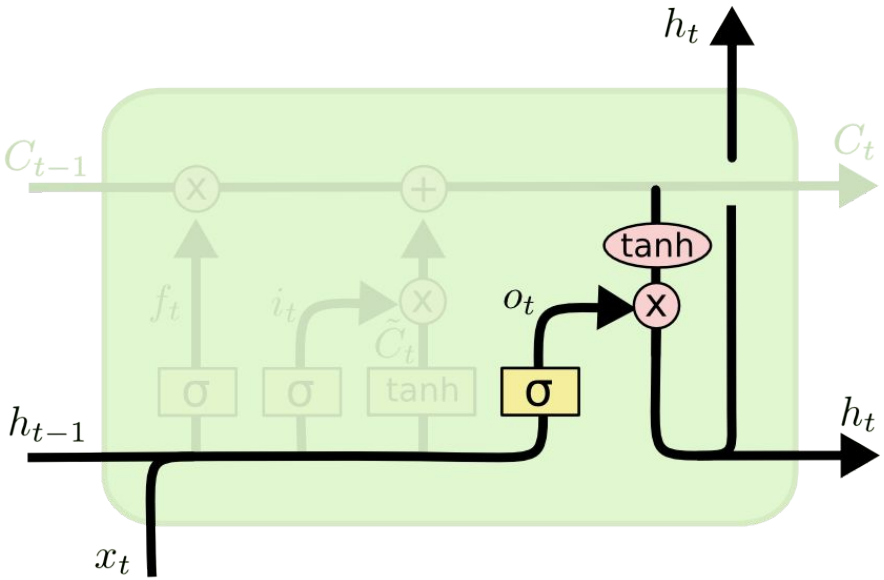
слой тангенса создает вектор новых значений-кандидатов

LSTM: считаем что получается



собираем два предыдущих действия вместе

LSTM: считаем что получается



- Сигмоидный слой решает, что нам нужно вывести.
- Тангенс нормализует к интервалу от -1 до 1.
- Значения умножаются и получаем выход.

Резюме

Два подхода: классические и нейронные.

Классика:

- наивный Байес
- логистическая регрессия
- SVM

Нейронные сети:

- CNN
- RNN
- LSTM

Спасибо за внимание!