OTUS

Рекомендательные системы

Рекомендательная система на основе текстовых данных

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Ставим “+”, если все хорошо
“-”, если есть проблемы

Тема вебинара

Рекомендательная система на основе текстовых данных

Дмитрий Гайнуллин

Преподаватель курсов Machine Learning и NLP
ML Engineer в компании AIC
Speech-to-text, LLMs, MLops
Занимаюсь NLP для малоресурсных языков

@gaydmi (TG)

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в учебной группе
#RecSys-2024-05

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

Пишем в чат

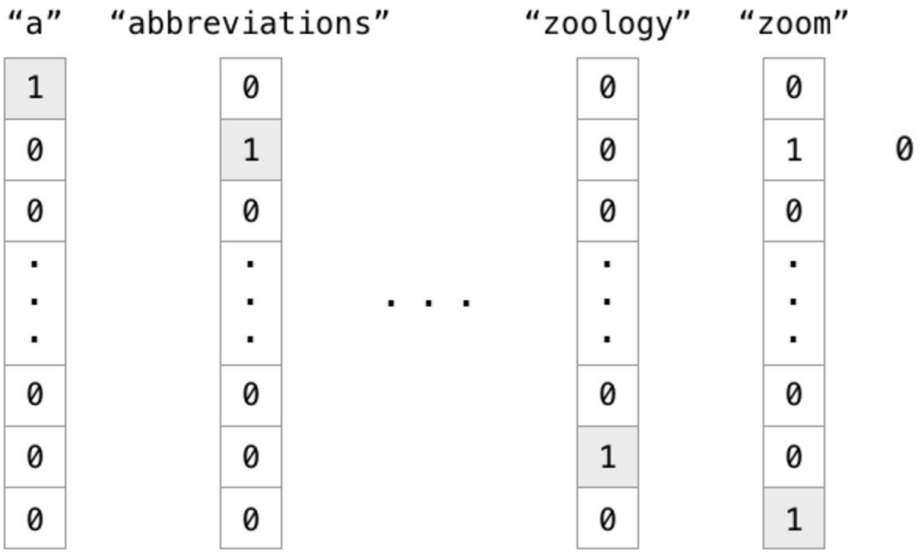
Говорим голосом

Документ

Ответьте себе или
задайте вопрос

One-hot encoding

Самый простой способ кодирования категориальных признаков:



Полученные векторы огромные и ортогональные

Вектора поумнее: TF-IDF

TF-IDF — статистическая мера, используемая для оценки важности слова в контексте документа, являющегося частью корпуса.
Вес некоторого слова пропорционален частоте употребления этого слова в документе и обратно пропорционален частоте употребления слова во всех документах коллекции.

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF

Term x within document y

$tf_{x,y}$ = frequency of x in y

df_x = number of documents containing x

N = total number of documents

Вектора поумнее: TF-IDF

	it	is	puppy	cat	pen	a	this
it is a puppy	1	1	1	0	0	1	0
it is a kitten	1	1	0	0	0	1	0
it is a cat	1	1	0	1	0	1	0
that is a dog and this is a pen	0	2	0	0	1	2	1
it is a matrix	1	1	0	0	0	1	0

SVD для получения эмбеддингов слов

По корпусу текстов D со словарём T строим **матрицу со-встречаемостей :**

$$X_{|T|\times|T|}$$

Элемент x_{ij} отражает со-встречаемость слов i и j в корпусе текстов.

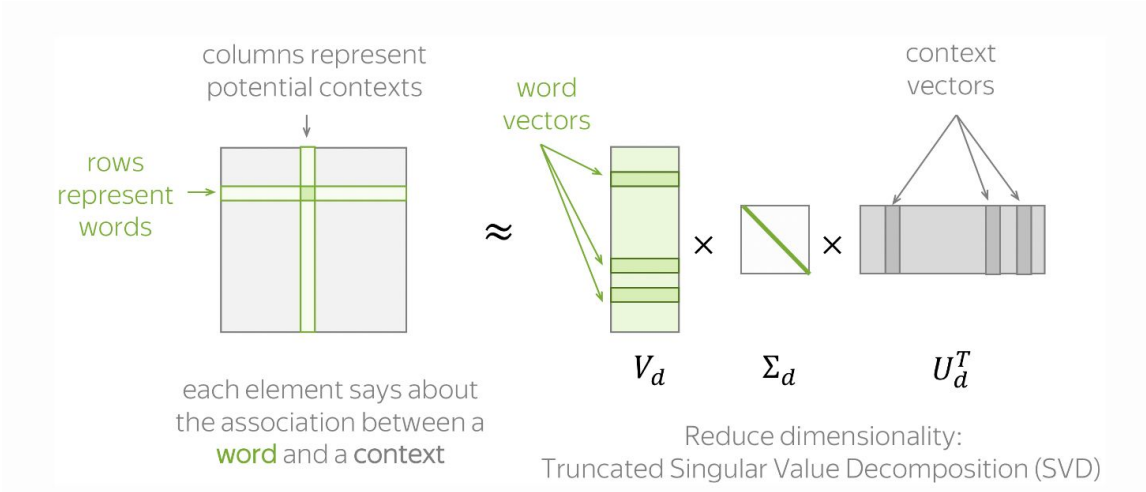
Возможны различные варианты учёта со-встречаемости слов:

- сумма по всей коллекции числа попаданий пары слов в окно фиксированного размера
- количество документов, хоть раз содержащих пару слов
- количество документов, хоть раз содержащих пару слов в окне

SVD для получения эмбеддингов слов

Понижаем размерность через SVD-разложение: $X = USV^T$

Из столбцов матрицы U выбираем первые K компонент



https://lena-voita.github.io/nlp_course/word_embeddings.html

Недостатки SVD

- Относительно низкое качество получаемых представлений
- Сложность работы с очень большой и разреженной матрицей
- Сложность добавления новых слов/документов (решается инкрементальными методами построения)

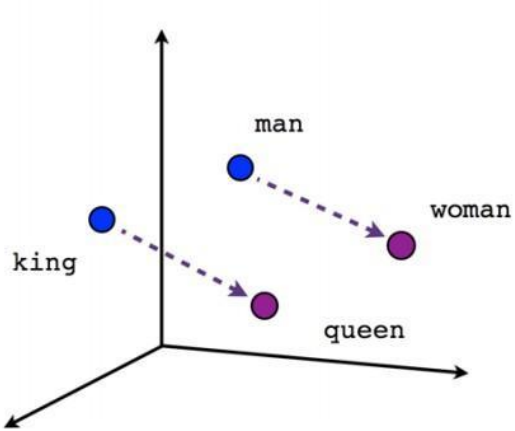
word2vec

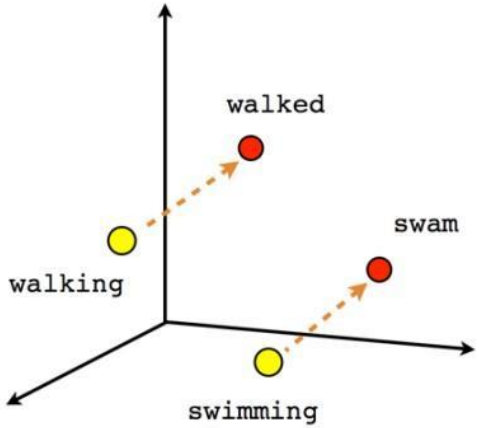
Mikolov, K. Chen, G. Corrado, J. Dean. Efficient Estimation of Word Representations in Vector Space (2013)

T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean. Distributed Representations of Words and Phrases and their Compositionality (2013).

word2vec

- **word2vec** — группа алгоритмов, предназначенных для получения векторных представлений слов
- **Идея:** «Слова со схожими значениями разделяют схожий контекст»
- Как правило, в векторном представлении семантически близкие слова оказываются рядом

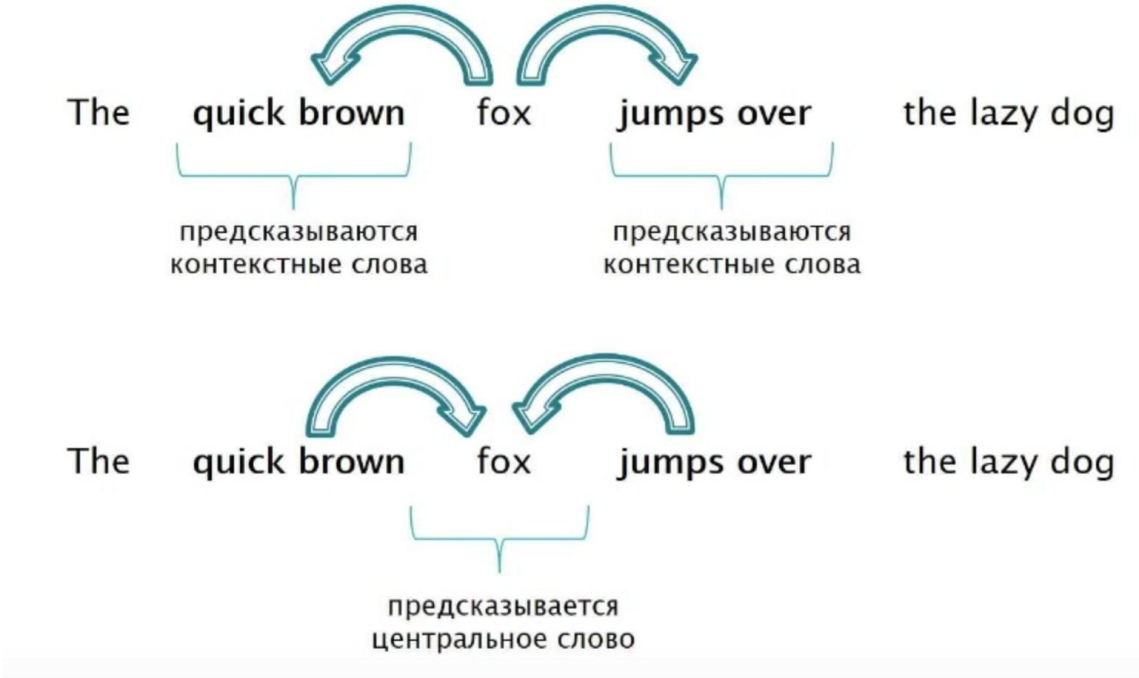




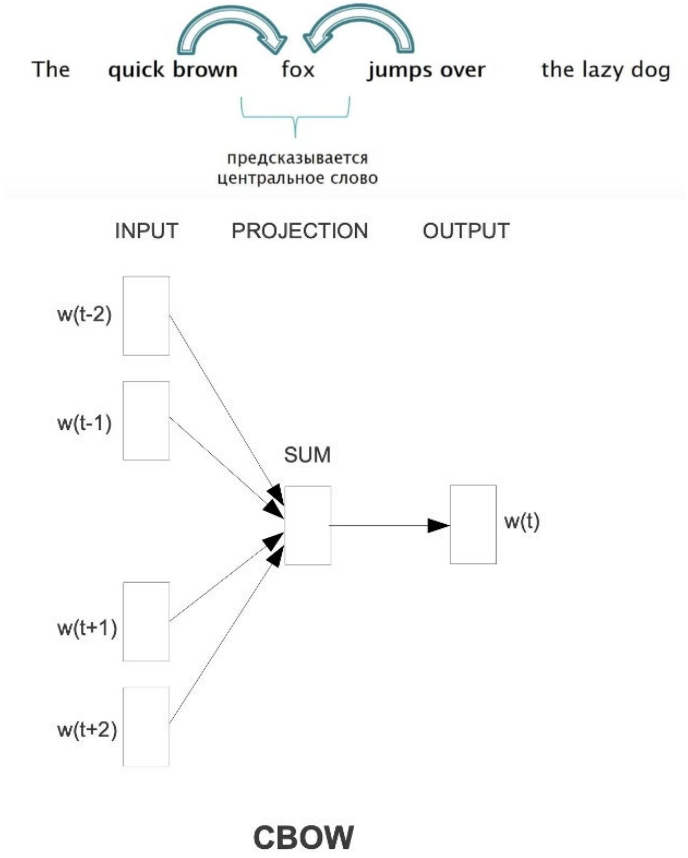
Две модели

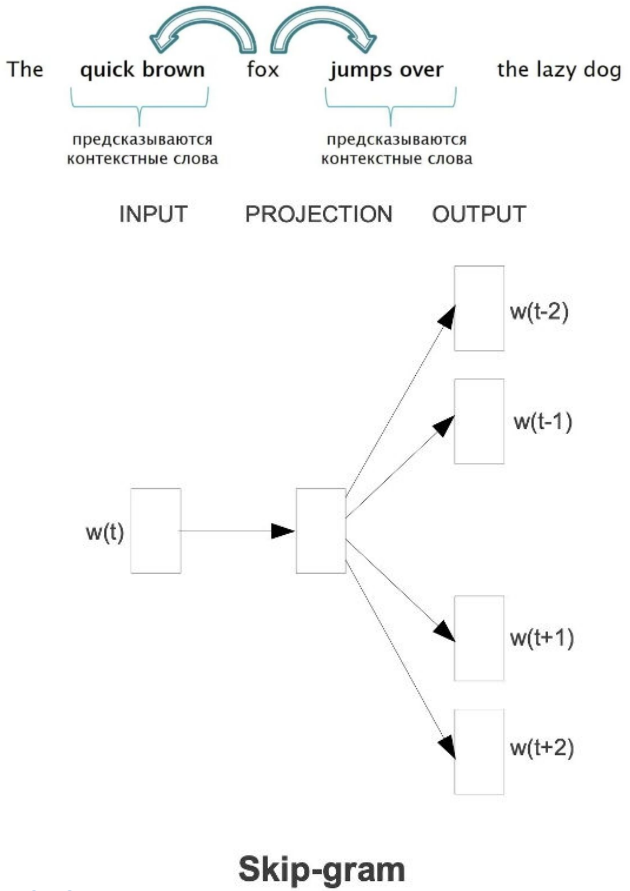
Skip-gram: по текущему слову предсказываем контекст

Continuous BOW: по контексту предсказываем текущее слово



Word2Vec





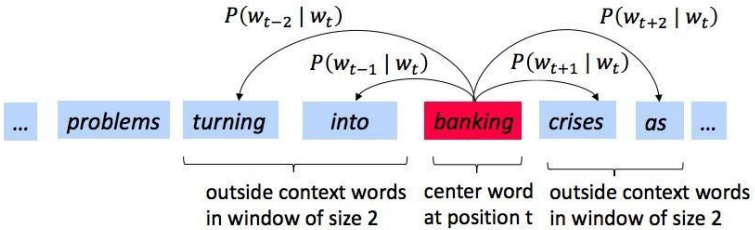
https://arxiv.org/abs/1301.3781

Skip-gram model

- Вероятность встретить слово w_O рядом со словом w_I:

$$p(w_O|w_I) = \frac{\exp\left(\left\langle v'_{w_O}, v_{w_I}\right\rangle\right)}{\sum_{w\in W}\exp\left(\left\langle v'_{w}, v_{w_I}\right\rangle\right)}$$

- W — словарь
- v_w — «центральное» представление слова
- v'_w — «контекстное» представление слова



https://trailhead.salesforce.com/

Skip-gram model

- Вероятность встретить слово w_O рядом со словом w_I:

$$p(w_O|w_I) = \frac{\exp\left(\left\langle v'_{w_O}, v_{w_I}\right\rangle\right)}{\sum_{w\in W}\exp\left(\left\langle v'_{w}, v_{w_I}\right\rangle\right)}$$

- Функционал для текста $T = (w_1 w_2 \dots w_n)$:

$$\sum_{i=1}^{n}\sum_{\substack{-c\le j\le c\\ j\neq 0}} \log p\left(w_{i+j}|w_i\right) \to \max$$

Skip-gram model

- Вероятность встретить слово w_O рядом со словом w_I:

$$p(w_O|w_I) = \frac{\exp(\langle v'_{w_O}, v_{w_I} \rangle)}{\sum_{w \in W} \exp(\langle v'_w, v_{w_I} \rangle)}$$

- Считать знаменатель ОЧЕНЬ затратно
- Значит, и производные считать тоже долго

Negative Sampling

$$p(w_O|w_I) = \log \sigma\left(\langle v'_{w_O}, v_{w_I}\rangle\right) + \sum_{i=1}^{k} \log \sigma\left(-\langle v'_{w_i}, v_{w_I}\rangle\right)$$

- w_i — случайно выбранные слова
- Слово w генерируется с вероятностью $P(w)$ — шумовое распределение
- $P(w) = \frac{U(w)^{\frac{3}{4}}}{\sum_{v \in W} U(v)^{\frac{3}{4}}}$, $U(v)$ — частота слова v в корпусе текстов

word2vec: особенности обучения

$$p(w_O|w_I) = \log \sigma\left(\langle v'_{w_O}, v_{w_I} \rangle\right) + \sum_{i=1}^{k} \log \sigma\left(-\langle v'_{w_i}, v_{w_I} \rangle\right)$$

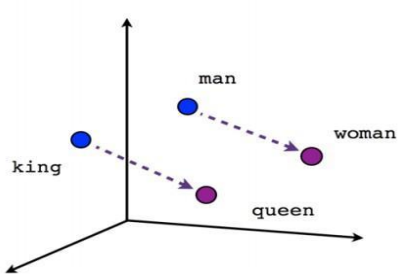
- w_i — случайно выбранные слова

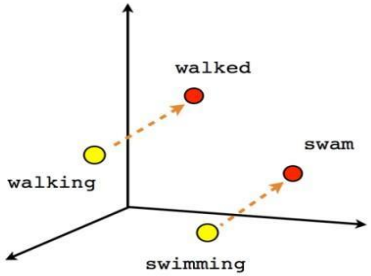
- Положительные примеры — слова, стоящие рядом
- Отрицательные примеры: подбираем к слову «шум», то есть другое слово, которое не находится рядом
- Важно семплировать в SGD слова с учётом их популярности — иначе будем обучаться только на самые частые слова

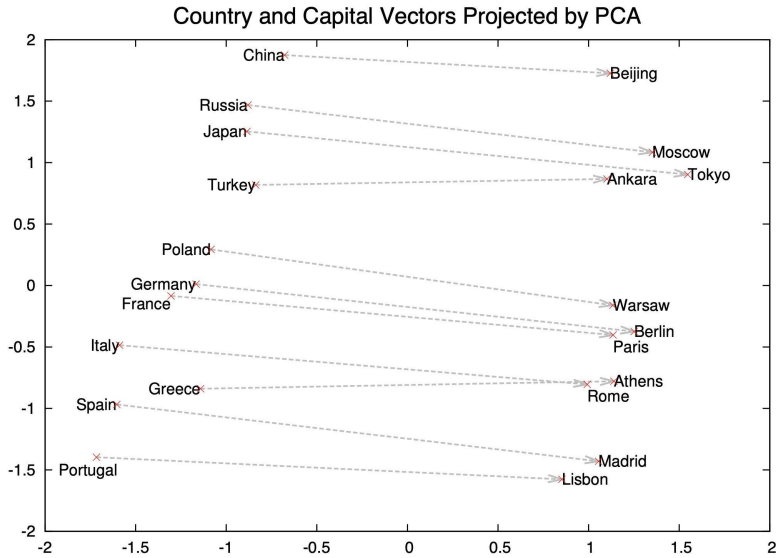
Как это использовать?

- Можно искать похожие слова
- Можно менять формы слов
- Можно искать определённые отношения
- Можно использовать как признаки для моделей

word2vec


man
woman
king
queen


walked
swam
walking
swimming


Country and Capital Vectors Projected by PCA
China
Beijing
Russia
Japan
Moscow
Tokyo
Ankara
Turkey
Poland
Germany
France
Warsaw
Berlin
Paris
Italy
Athens
Rome
Greece
Spain
Madrid
Portugal
Lisbon
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

Проблемы word2vec

- Не учитываем структуру слов
- Не закладываем никакой априорной информации о разных формах одного слова
- Не умеем обрабатывать опечатки

Вопросы?

FastText

- Заменим каждое слово на «мешок»
- «руслан» -> (<руслан>, <ру, рус, усл, сла, лан, ан>)
- Слово w заменяется на набор токенов $t_1, \dots, t_n$
- Мы обучаем векторы токенов: $v_{t_1}, \dots, v_{t_n}$ (на самом деле есть «центральные» и «контекстные» версии всех векторов)
- $z_w = \sum_{i=1}^{n} v_{t_i}$ — вектор слова
- Все остальные детали — как в word2vec

https://arxiv.org/abs/1607.04606

Что бывает ещё?

- GloVe
- ELMo
- Трансформерные модели (BERT и т. п.)

Работа с текстом

- Векторные представления строятся для слов
- Можно просто усреднить по всем словам — получим признаки для текста
- Можно усреднять с весами
- Популярный вариант: *Word2Vec/FastText с весами tf-idf*
- Можно ли умнее?

Оценка качества

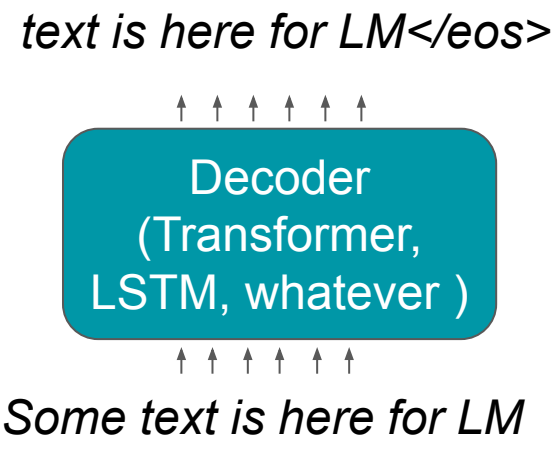
Внешний критерий: метрика исходной задачи

Внутренний: качество поиска размеченных аналогий

Вопросы?

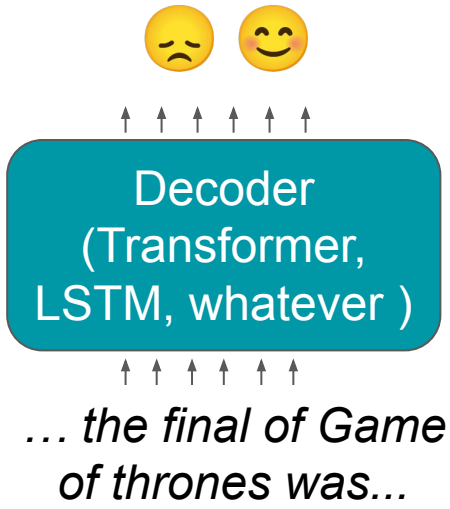
Pretraining

- Предобучить модель на задачу языкового моделирования
- Обучить модель языку на огромном корпусе неразмеченных текстов
- Сохранить параметры модели



Fine-tuning

- Взять предобученную модель
- Дообучить под конкретную задачу на небольшом объеме данных
- Получить высокий результат благодаря знаниям модели, полученным на этапе предобучения



BERT

BERT = **B**idirectional ***Encoder*** **R**epresentations from **T**ransformers

Новый подход дообучения предобученных моделей (pre-training + fine-tuning) для решения задач самых разных NLP задач

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

BERT в топе

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

MultiNLI

Premise: Hills and mountains are especially sanctified in Jainism.
Hypothesis: Jainism hates nature.
Label: Contradiction

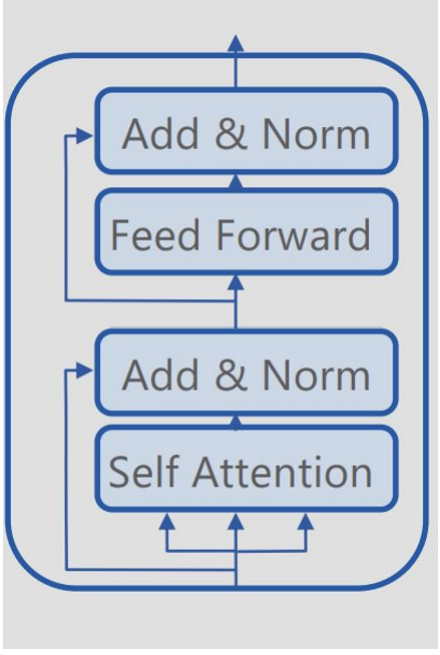
CoLa

Sentence: The wagon rumbled down the road.
Label: Acceptable

Sentence: The car honked down the road.
Label: Unacceptable

Архитектура BERT

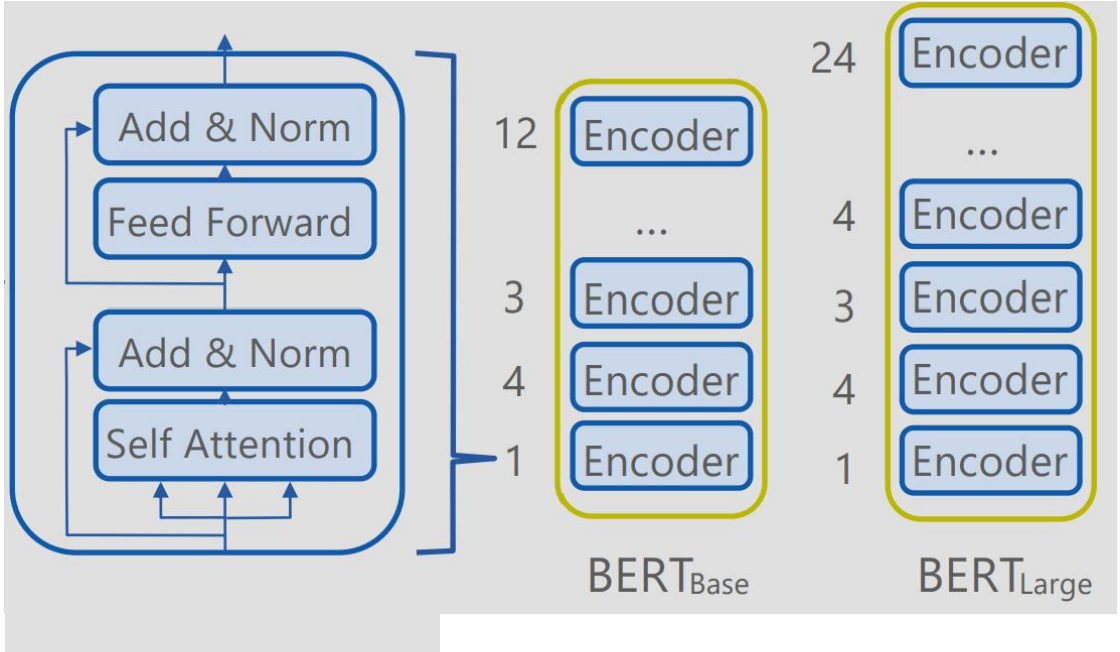
Модель основана на энкодерной части трансформера



Архитектура BERT

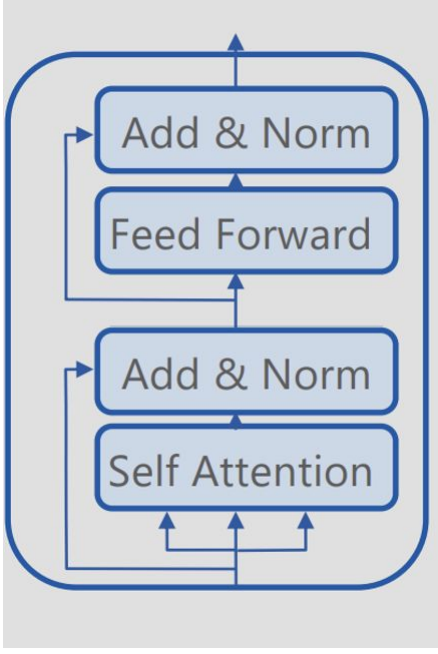
Модель основана на энкодерной части трансформера

Две конфигурации: *base & large*



Архитектура BERT

Вопрос: какое основное отличие между энкодером и декодером?

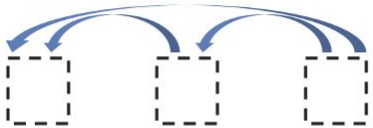


Архитектура BERT

Вопрос: какое основное отличие между энкодером и декодером?

В энкодере self-attention смотрит на весь контекст, а декодер использует masked self-attention, запрещая токенам смотреть в будущее.

Encoder Self-Attention



MaskedDecoder Self-Attention

Архитектура BERT

Вопрос: какое основное отличие между энкодером и декодером?

В энкодере self-attention смотрит на весь контекст, а декодер использует masked self-attention, запрещая токенам смотреть в будущее.

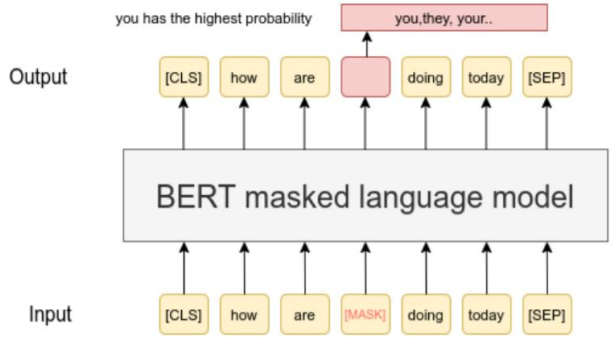
⇩

Как энкодер, BERT смотрит на весь контекст (токены справа и слева).

BERT: предобучение

Две задачи:

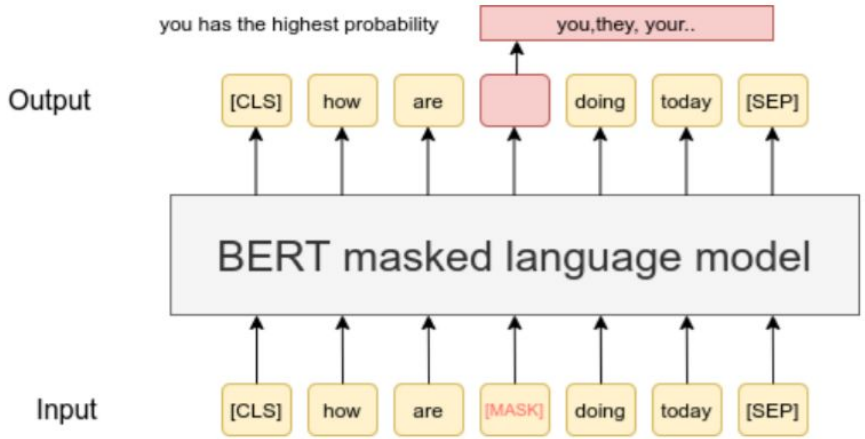
- Masked Language Modeling Task (**MLM**)
- Next Sentence Prediction (**NSP**)



Masked Language Modeling Task (MLM)

Маскируем *k%* слов (токенов) в текстах, затем учим модель их предсказывать (обычно *k=15%*).

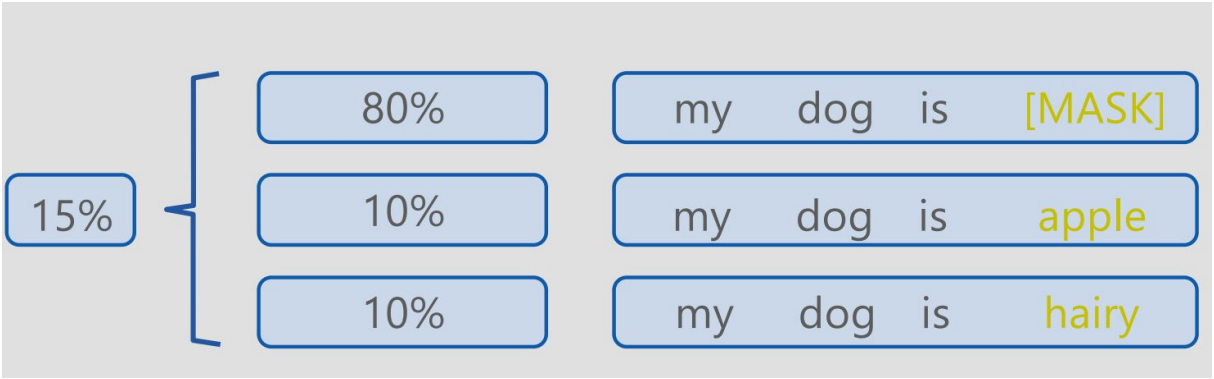
- *Too little masking: слишком дорого тренировать*
- *Too much masking: недостаточно текста для обучения*



Masked Language Modeling Task (MLM)

Случайным образом выбираем *15%* токенов, при этом:

- *80 % заменяем на [MASK]*
- *10% заменяем на случайные слова*
- *10% оставляем без изменения*



Next Sentence Prediction (NSP) objective

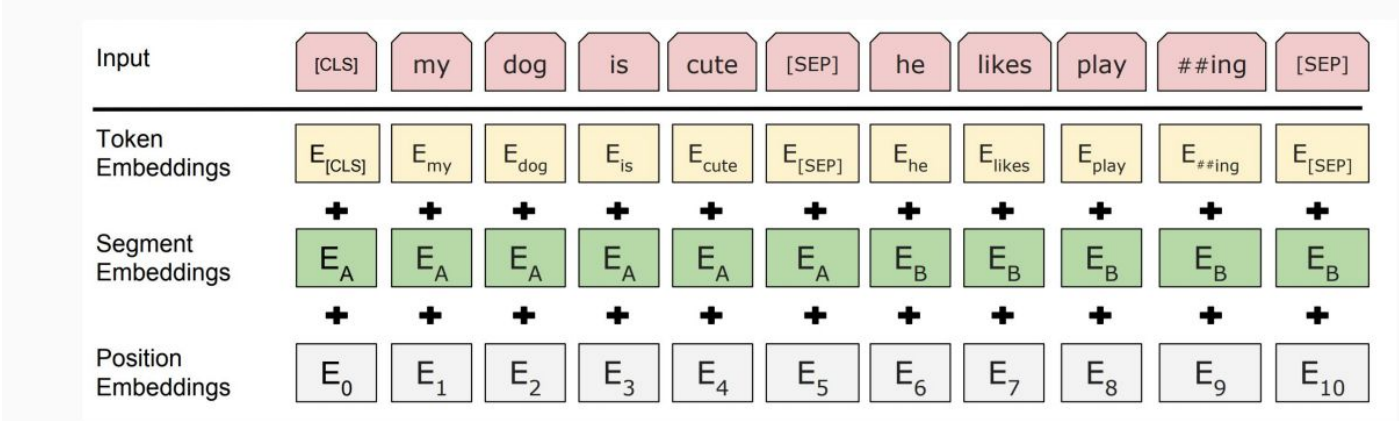
- Два предложения: A,B;
- Задача: предсказать, является ли B продолжением A
  - *Сэмплируем пары A,B (с p=0.5) текстов*
  - *Учимся извлекать взаимосвязь между 2 предложениями*

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.
Slides. https://nlp.stanford.edu/seminar/details/jdevlin.pdf

Как получить эмбеддинг предложения?

- [CLS] – эмбеддинг CLS token’a с последнего слоя
- [MEAN] – усреднение векторов-слов (токенов) с последнего слоя
- [MAX] – покомпонентный максимум (maxpooling) векторов-слов (токенов) с последнего слоя

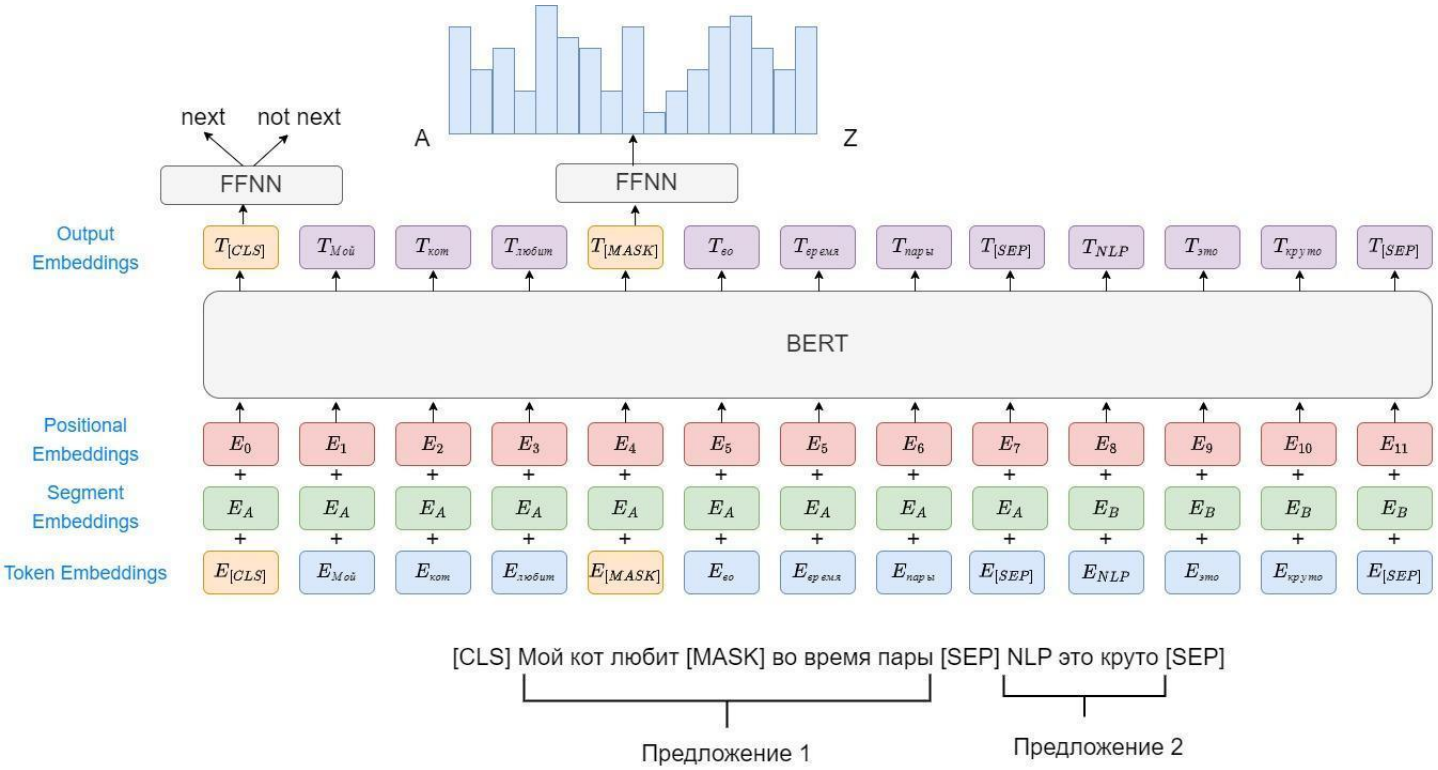
Input



- Use 30,000 WordPiece vocabulary on input.
- Each token is sum of three embeddings
- Single sequence is much more efficient.

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.
Slides. https://nlp.stanford.edu/seminar/details/jdevlin.pdf

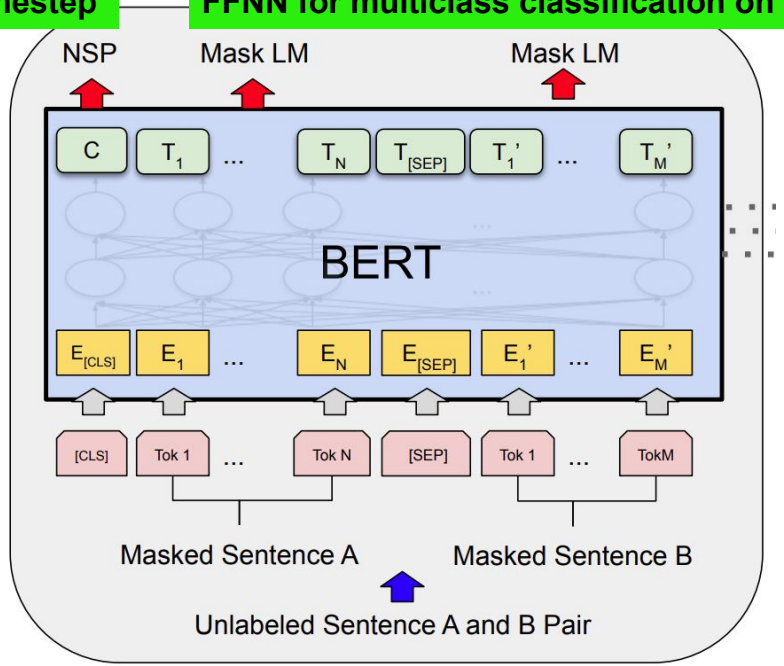
MLM и NSP вместе



BERT предобучение

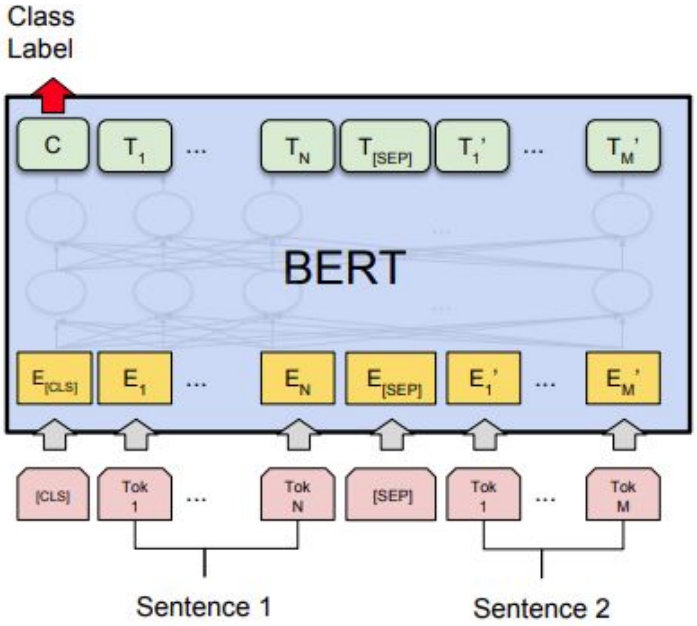
FFNN for binary classification on 1st timestep

FFNN for multiclass classification on each timestep

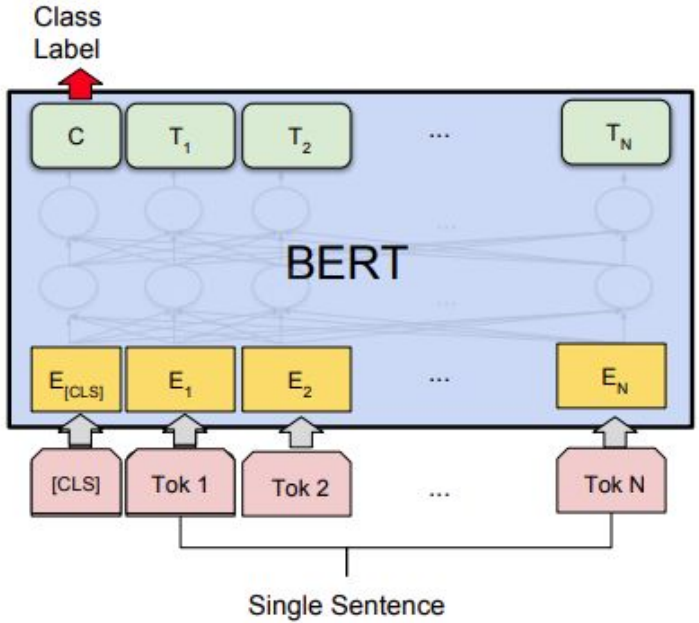


Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.
Slides. https://nlp.stanford.edu/seminar/details/jdevlin.pdf

BERT Fine-tuning



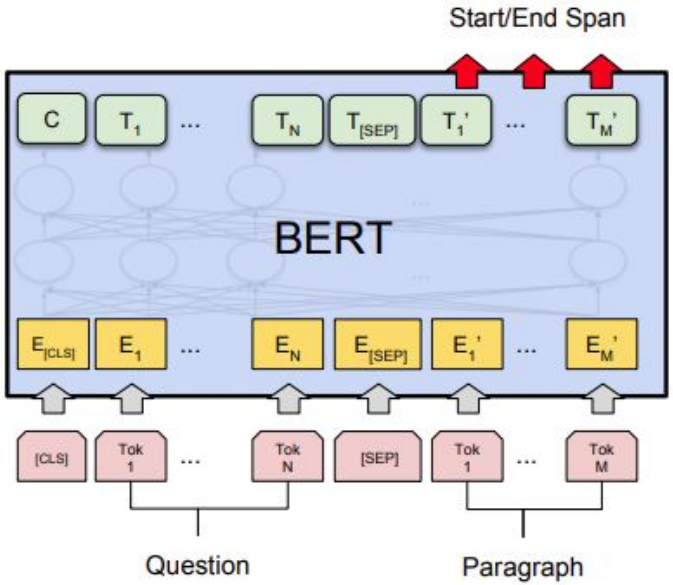
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



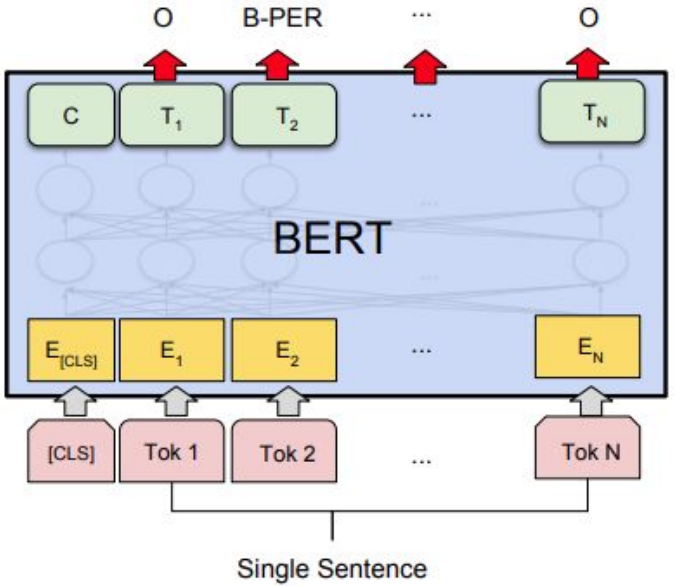
(b) Single Sentence Classification Tasks: SST-2, CoLA

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT Fine-tuning



(c) Question Answering Tasks:
SQuAD v1.1



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Результаты

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

MultiNLI

Premise: Hills and mountains are especially sanctified in Jainism.
Hypothesis: Jainism hates nature.
Label: Contradiction

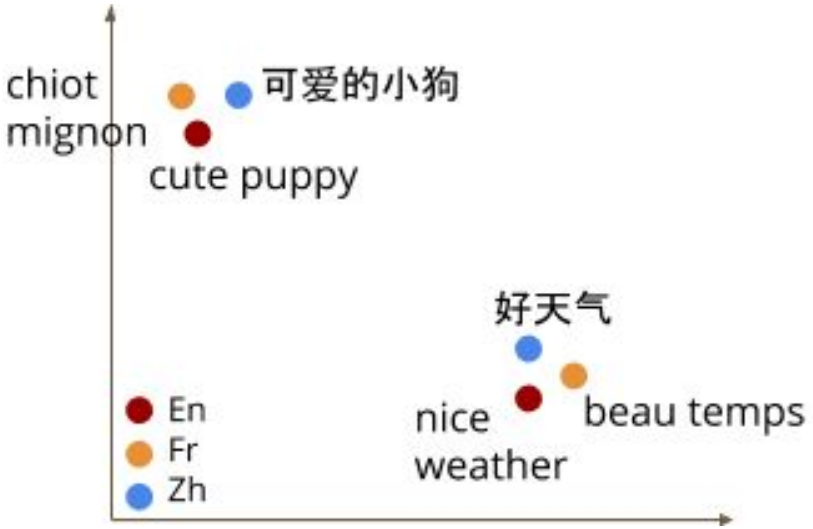
CoLa

Sentence: The wagon rumbled down the road.
Label: Acceptable

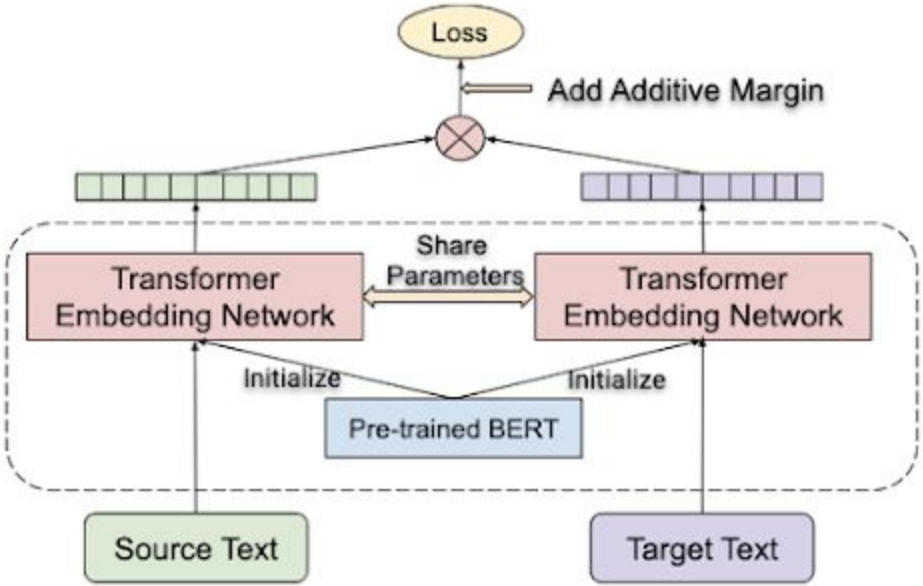
Sentence: The car honked down the road.
Label: Unacceptable

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.
Slides. https://nlp.stanford.edu/seminar/details/jdevlin.pdf

Language-Agnostic BERT Sentence Embedding



Language-Agnostic BERT Sentence Embedding



Вопросы?

LIVE

Заполните, пожалуйста,
опрос о занятии
по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары

Дмитрий Гайнуллин

ML Engineer, AIC

@gaydmi + чат группы https://t.me/+PTsYJeYOKsBmYzg6