Глубинное обучение
Лекция 7: Нейросети в задачах обработки текстов

Лектор: Антон Осокин
Лаборатория компании Яндекс, ФКН ВШЭ
Yandex Research

Научные стажировки: https://cs.hse.ru/big-data/yandexlab/internship

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

ФКН ВШЭ, 2021

Виды задач для нейросетей

- Очень сильно упрощаем: текст – последовательность токенов
- Классификация последовательностей
  - Примеры: определение темы, sentiment analysis
  - Учитывать последовательность или нет?
  - Bag-of-words, RNN, transformer

Виды задач для нейросетей

- Очень сильно упрощаем: текст – последовательность токенов
- Классификация последовательностей
  - Примеры: определение темы, sentiment analysis
  - Учитывать последовательность или нет?
  - Bag-of-words, RNN, transformer
- Разметка последовательности
  - Примеры: определение частей речи, chunking
  - Локальный классификатор, RNN, CRF, RNN-CRF (BiLSTM-CRF)

Виды задач для нейросетей

- Очень сильно упрощаем: текст – последовательность токенов
- Классификация последовательностей
  - Примеры: определение темы, sentiment analysis
  - Учитывать последовательность или нет?
  - Bag-of-words, RNN, transformer
- Разметка последовательности
  - Примеры: определение частей речи, chunking
  - Локальный классификатор, RNN, CRF, RNN-CRF (BiLSTM-CRF), transformer
- Последовательности в последовательность (разной длины!)
  - Примеры: машинный перевод, аннотация (summarization), диалоги
  - Авторегрессионные модели: seq2seq (+attention), ByteNet, transformer и т.д.
  - Неавторегрессионные модели

Виды задач для нейросетей

- Очень сильно упрощаем: текст – последовательность токенов
- Классификация последовательностей
  - Примеры: определение темы, sentiment analysis
  - Учитывать последовательность или нет?
  - Bag-of-words, RNN, transformer
- Разметка последовательности
  - Примеры: определение частей речи, chunking
  - Локальный классификатор, RNN, CRF, RNN-CRF (BiLSTM-CRF)
- Последовательности в последовательность (разной длины!)
  - Примеры: машинный перевод, аннотация (summarization), диалоги
  - Авторегрессионные модели: seq2seq (+attention), ByteNet, transformer и т.д.
  - Неавторегрессионные модели
- Генерация последовательностей
  - Примеры: описание изображения (captioning)
  - Авторегрессионные и неавторегрессионные модели

План лекции

- Непрерывные представления слов (embedding)
  - word2vec, FastText
- Обработка последовательностей
  - Seq2seq
  - Seq2seq + attention
  - Transformer
- Контекстно-зависимые представления (предобучение)
  - ELMo, BERT и его друзья

Как вставить текст в нейросеть?

Непрерывные представления слов (word embeddings)

- Позволяют строить непрерывные представления дискретных объектов
- Непрерывные представления – это способ поместить текст в нейросеть
- Представление – вектор по индексу (токена: слова, символы, n-граммы)
- Представления могут обучаться совместно с моделью

Непрерывные представления слов (word embeddings)

- Позволяют строить непрерывные представления дискретных объектов
- Непрерывные представления – это способ поместить текст в нейросеть
- Представление – вектор по индексу (токена: слова, символы, n-граммы)
- Представления могут обучаться совместно с моделью

- Предобученные представления
  - Обучены на больших корпусах текстов
  - Обучены без ручной разметки (self-supervision)
  - Freeze, fine-tune, train from scratch?

Представления word2vec (skipgram)

[Mikolov et al., 2013]

- Обучение на предсказании контекста по слову
  - Обучение на корпусе текстов без разметки (self supervision)
- Вспомогательная задача:
  - Предсказываем каждое слово из контекста отдельно

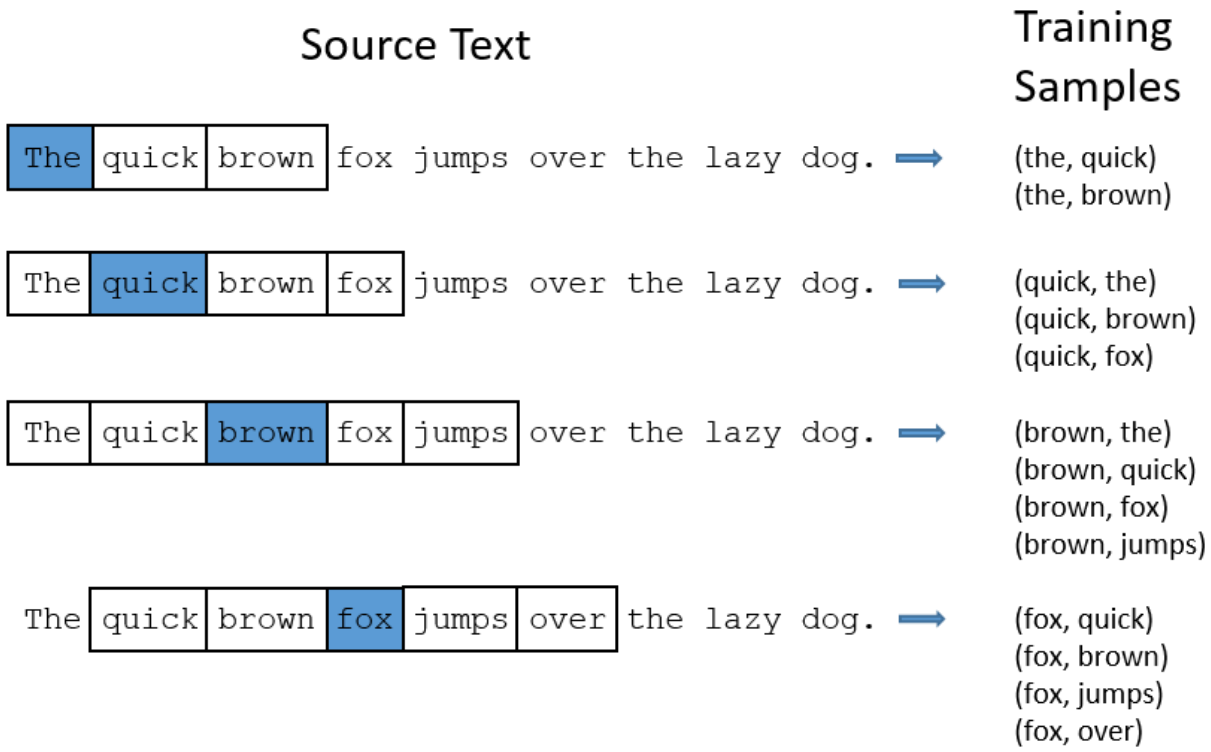


image credit:
Chris McCormick

Представления word2vec (skipgram)

[Mikolov et al., 2013]

- Обучение на предсказании контекста по слову
  - Обучение на корпусе текстов без разметки
- Предсказываем каждое слово из контекста отдельно
  - Текущее слово w; слово из контекста v
  - Для каждого слова – 2 представления (*in*, *out*)
  - Совместимость – скалярное произведение $\text{in}_w^T \text{out}_v$
  - Полезные – представления *in*
  - Модель с softmax $P(v \mid w, \theta) = \frac{\exp(\text{in}_w^T \text{out}_v)}{\sum_{v'} \exp(\text{in}_w^T \text{out}_{v'})}$
    - Медленная нормировка
  - Обычное решение – Noise Contrastive Estimation (NCE)

$$\text{loss}(w, v) = \log(1 + \exp(-\text{in}_w^T \text{out}_v)) + \sum_{\text{random } v'} \log(1 + \exp(\text{in}_w^T \text{out}_{v'}))$$

Представления word2vec (skipgram)

[Mikolov et al., 2013]

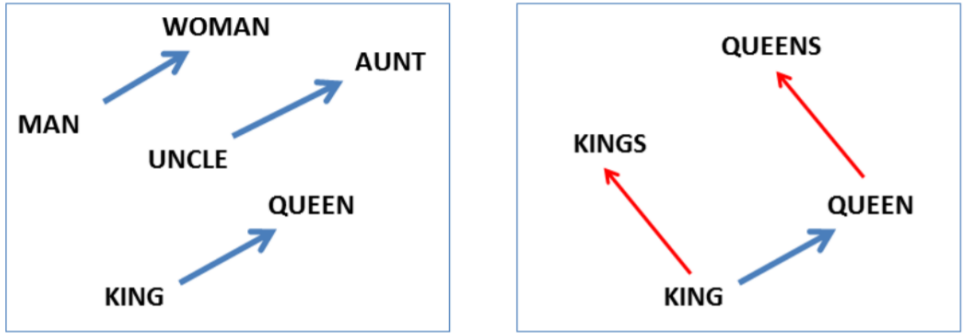
- Обучение на предсказании контекста по слову
  - Обучение на корпусе текстов без разметки
- Предсказываем каждое слово из контекста отдельно
  - Используются представления *in*
- Достоинства
  - Ближайшие соседи (cosine distance = норм. скал. произв., корпус GoogleNews)
    - university: student, teacher, teaching, students, schools
    - Putin: Medvedev, Vladimir_Putin, President_Vladimir_Putin, Prime_Minister_Vladimir_Putin, Kremlin
    - putin: lol, mr, don't, obama, Hahaha
    - obama: dems, americans, washington, america, libs

Source: http://bionlp-www.utu.fi/wv_demo/

Представления word2vec (skipgram)

[Mikolov et al., 2013]

- Обучение на предсказании контекста по слову
  - Обучение на корпусе текстов без разметки
- Предсказываем каждое слово из контекста отдельно
  - Используются представления *in*
- Достоинства
  - Ближайшие соседи (cosine distance = норм. скал. произв., корпус GoogleNews)
  - Арифметика над представлениями
    - king – man + woman = queen



Source: [Mikolov et al., 2013]

- Популярные представления – GloVe [Pennington et al., 2014]

Представления fastText (skipgram)

[Bojanowski et al., 2017]

Как в word2vec:

- Обучение на предсказании контекста по слову
  - Обучение на корпусе текстов без разметки
- Предсказываем каждое слово из контекста отдельно

Новая идея:

- Добавить информацию о символах слова (через n-граммы)

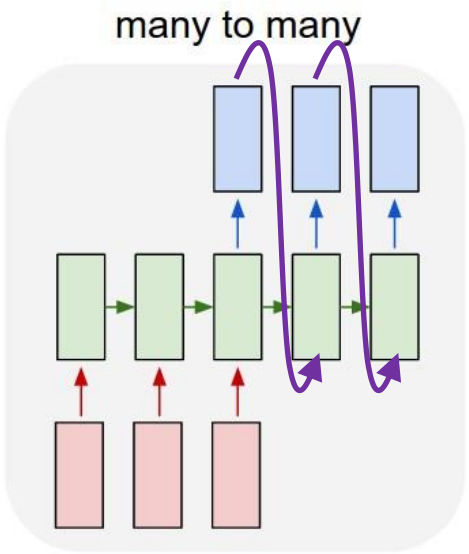
$$\text{in}_w = \text{word}_w + \sum_{p \in \text{n-grams(w)}} \text{part}_p \qquad \text{in}_w^T \text{out}_v \Rightarrow \text{word}_w^T \text{out}_v + \sum_{p \in \text{n-grams(w)}} \text{part}_p^T \text{out}_v$$

  - “where” = “<where>”, “<wh”, “whe”, “her”, “ere”, “re>”
- Важно использовать длинные n-граммы (n ≤ 6)
- Достоинства:
  - Близость по написанию
  - Слова вне словаря, опечатки и т.д.

Код и данные на fasttext.cc

Модель seq2seq [Sutskever et al. , 2014]

- Модели для предсказания последовательностей разной длины



Входы, память, выходы

Входы – представления входов

Память – слои RNN

Выходы – шансы слов из словаря (logits, идут в logsoftmax)

Авторегрессионные связи (→) передают решение о текущем слове

На входы → подаются представления выходного алфавита

image credit: Andrej Karpathy

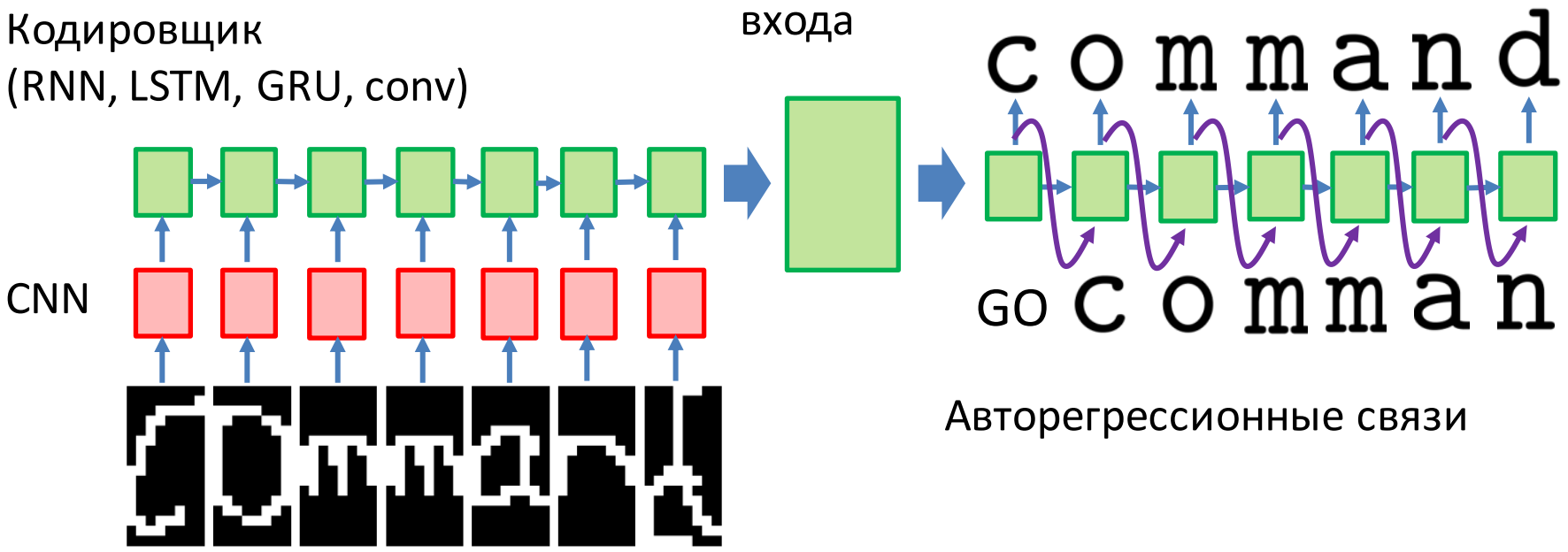
Последовательное предсказание

- Пример:  command

Представление входа

Выход (предсказание)

Кодировщик
(RNN, LSTM, GRU, conv)



Авторегрессионные связи

Если не фиксирована длина выхода, то используют символ EOS (с барьером)

Обучение авторегрессионных моделей

- Обычный способ – метод максимального правдоподобия

$$P(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}) = P(y_1 \mid \boldsymbol{x}, \boldsymbol{\theta}) P(y_2 \mid y_1, \boldsymbol{x}, \boldsymbol{\theta}) P(y_3 \mid y_2, y_1, \boldsymbol{x}, \boldsymbol{\theta}) \dots$$

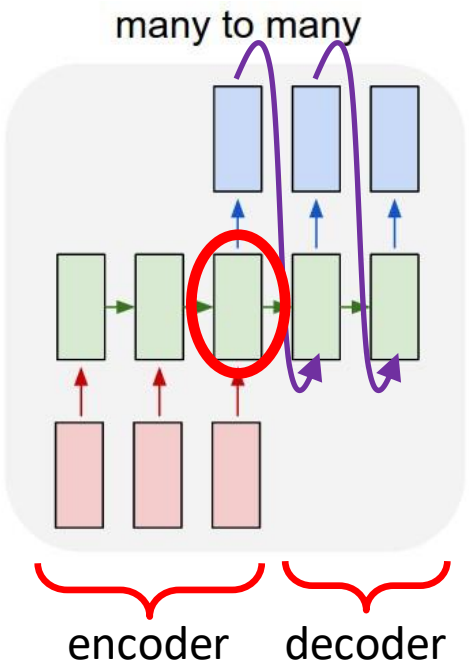
- На каждом шаге декодировщика – softmax и log-loss

- Teacher forcing – на вход декодировщику подаются правильные ответы

- Проблема:
  - Модель видит только правильные траектории
  - Не знает, что делать при ошибке
  - Как исследовать траектории (exploration)?
  - Связь с обучением с подкреплением (RL)

Модель seq2seq [Sutskever et al. , 2014]

- Модели для предсказания последовательностей разной длины



Модель encoder-decoder

○ – представление всего входа

Модель плохо работает для длинных последовательностей

Причина: представление входа – вектор фиксированной размерности

(не может представить весь язык)

Решение: механизм внимания

(attention)

image credit: Andrej Karpathy

Модель seq2seq с вниманием

[Bahdanau et al. , 2015]

- Модели для предсказания последовательностей разной длины

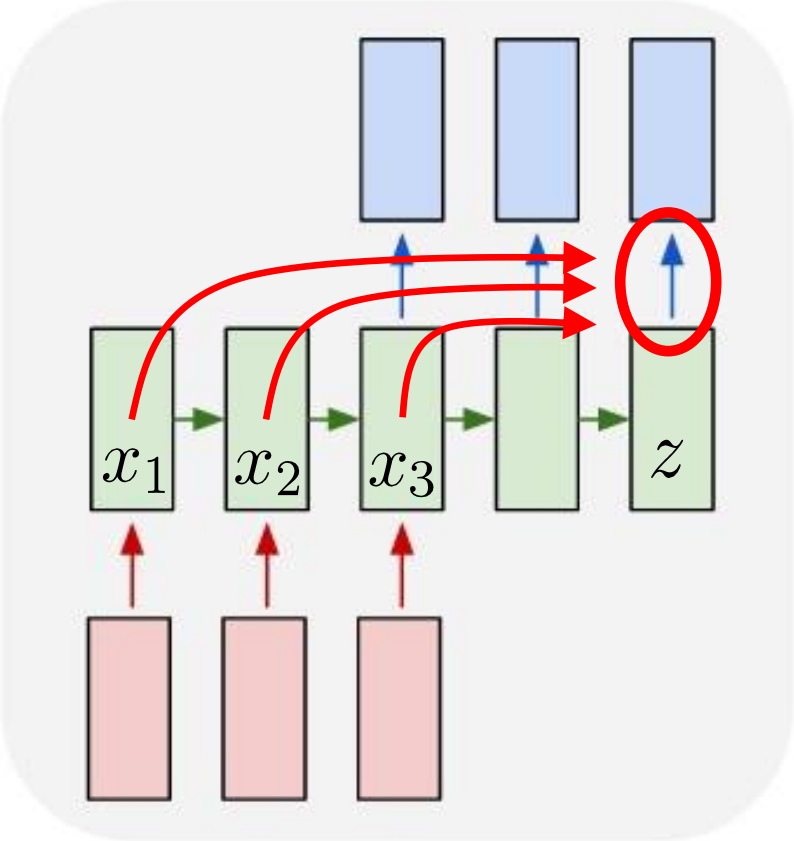


image credit: Andrej Karpathy

Внимание «выбирает релевантные элементы памяти»

Модель внимания:

- релевантность

$$s_i := \text{score}(x_i, z) = \begin{cases} x_i^T z \\ W[x_i; z] \end{cases}$$

W – параметры модели

- вероятности

$$a_1, a_2, \dots := \text{softmax}(s_1, s_2, \dots)$$

- контекст

$$c := \sum_i a_i x_i$$

- новые признаки $\tilde{z} := [c; z]$
- soft-argmax

Transformer [Vaswani et al., 2017]

- Статья – Attention Is All You Need!
- Архитектура:
  - Multi-head self attention

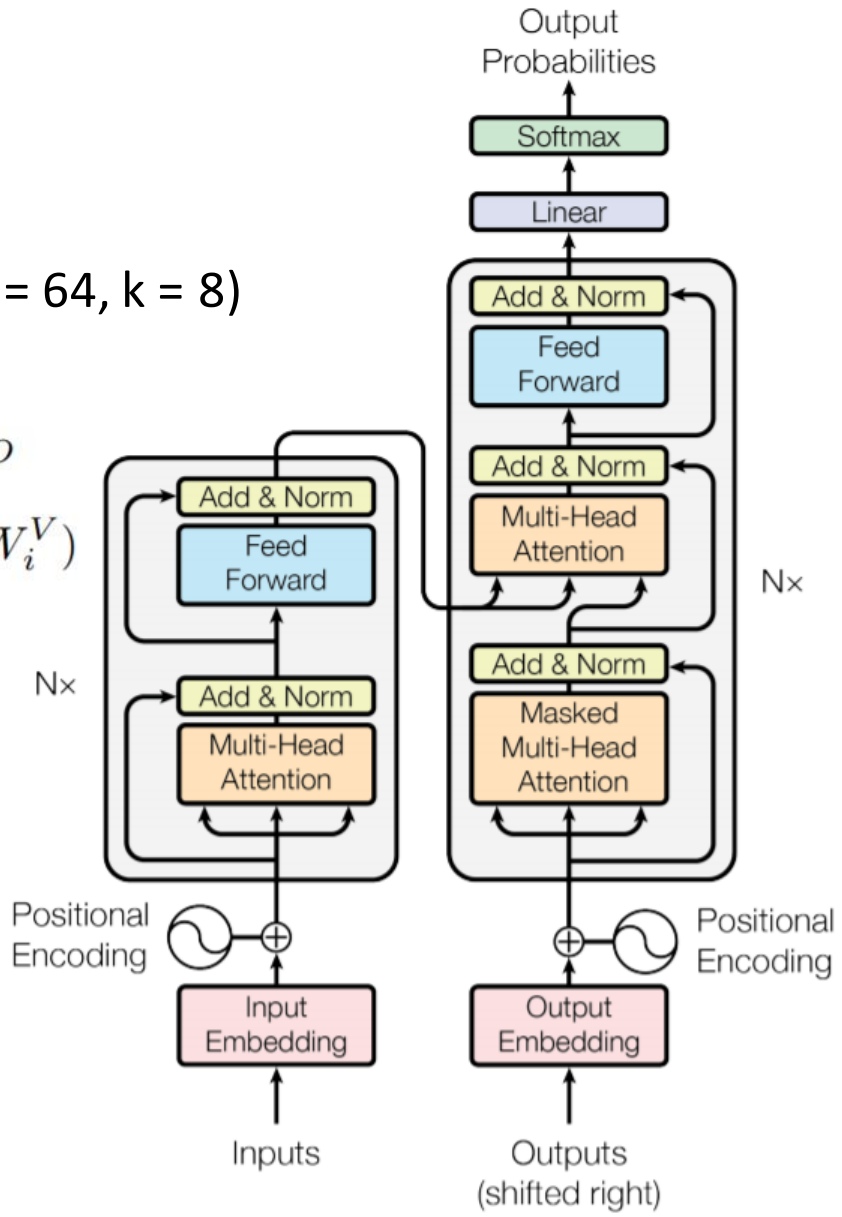
(Q = query, K = key, V = value, d = dimension = 64, k = 8)

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$

$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

  - 2-layer NN with ReLu
  - Encoding: token and positional
    - Positional – sin и cos от длины
  - В декодере –маска будущего!
- SOTA в переводе и др.!
- Поиск архитектуры [So et al., 2019]
- Ускорение: Reformer, Linformer, etc.



Словарь из Byte Pair Encoding (BPE)

[Sennrich et al., 2015]

- Размер словаря – важный параметр!
  - Большой => мало слов на редкие позиции, медленно, память
  - Маленький => много слов вне словаря
- Построение словаря BPE - итеративно
  - Инициализация – из токенов символов (unicode - осторожно)
  - Итеративное склеивание самых частых пар
  - Пересечение границ слов?
  - Символы типа знаков препинания?
- Позволяет делать представления любого слова
- Размер словаря – контролируемый параметр

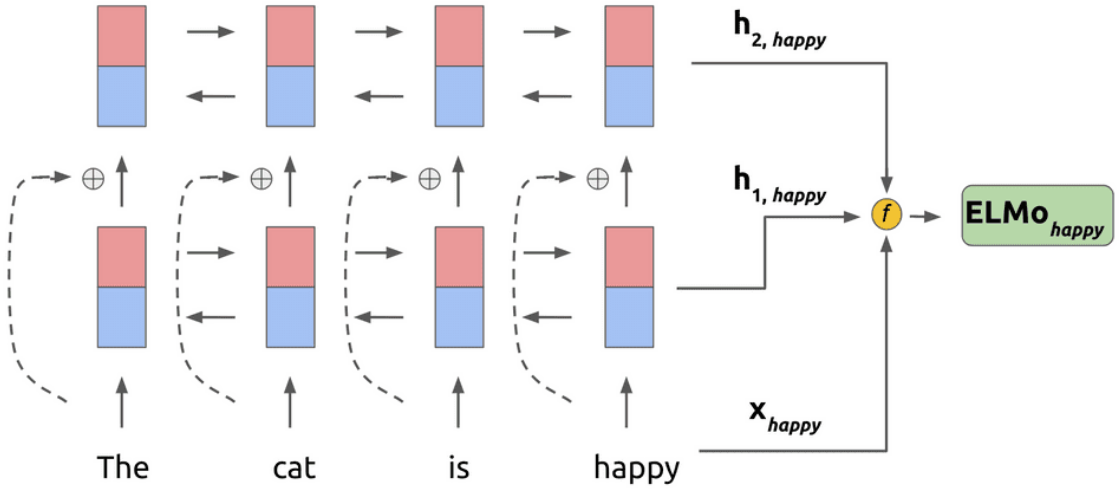
Предобученые представления слов – из языковых моделей!

ELMo

[Peters et al., 2018; AllenNLP]
https://github.com/allenai/allennlp

- ELMo = Embeddings from Language Models
- Контекстно зависимые представления!
- Архитектура:
  - Представления слов = свертки поверх представлений символов
    - Легко обрабатывать слова вне словаря
  - 2 модели поверх – forward и backward (2-layer LSTM + skip con.)
  - Итоговое представление – линейная комбинация представлений слоев!

- Обучение:
  - Forward – след. слово
  - Backward – пред. слово



BERT

[Devlin et al., 2018; Google]

- BERT = Bidirectional Encoder Representations from Transformers
- Очень большой трансформер:
  - L – глубина, H – размерность внутри, A – число голов multi-head attention
  - $BERT_{BASE}$: L=12, H=768, A=12, Total Parameters=110M
  - $BERT_{LARGE}$: L=24, H=1024, A=16, Total Parameters=340M
- Обучение:
  - Masked LM: 15% tokens selected randomly
    - 80% - [MASK]
    - 10% - исходное слово
    - 10% - случайное слово
  - Next sentence

Input = [CLS] the man went to [MASK] store [SEP]
he bought a gallon [MASK] milk [SEP]
Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]
penguin [MASK] are flight ##less birds [SEP]
Label = NotNext

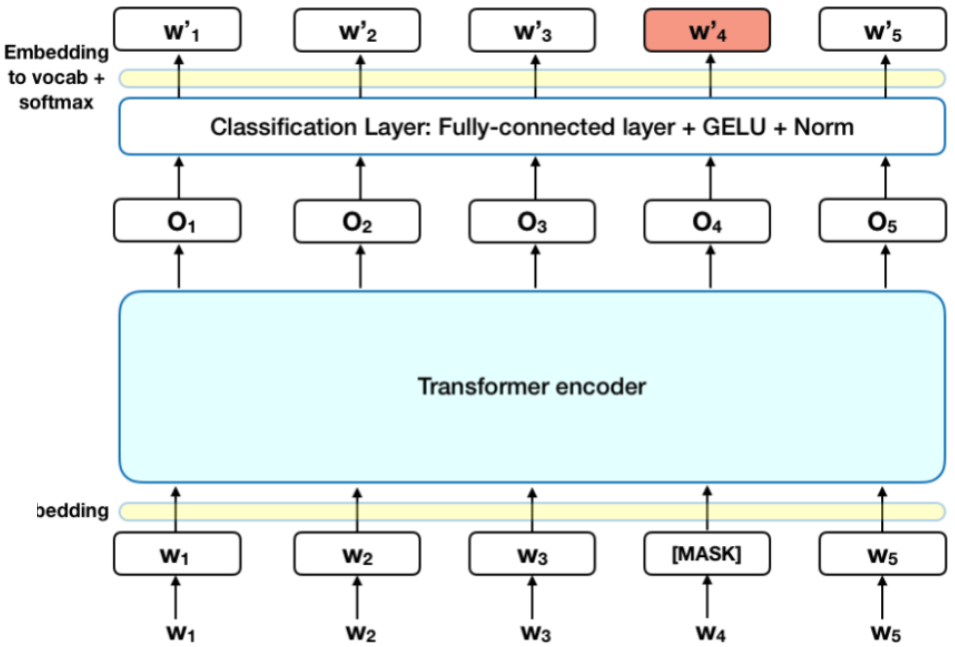


image credit: Rani Horev

BERT friends: RoBERTa, ALBERT, T5, GPT-3

[Liu et al., 2019; Facebook]
[Lan et al., 2019; Google]
[Raffel et al., 2019; Google]
[Brown et al., 2020; OpenAI]

- Тренд: увеличение моделей и данных
- Обучение очень сложное и дорогое
- Высокая чувствительность к гиперпараметрам
- RoBERTa: BERT был существенно недообучен
  - Можно стать SOTA изменив параметры и обучая дольше
  - Большой батч, byte-level (а не character-level, unicode!) BPE, динамические маски для предложений, etc.
- Можно (часто нужно!) использовать готовые модели!
- GPT-3 – 175B параметров, обучение стоит миллионы $
  - Может работать для предсказания во few-shot режиме без fine-tuning
  - но не доступна
- Отличная библиотека:
  - https://github.com/huggingface/transformers

Заключение

- Обработка языка активно использует нейросети
- Очень большая область – много задач
  - Есть успехи!
- Представления, Seq2seq, внимание, transformer, BERT, etc.
- Понимание смысла – очень сложная задача!