Глубокое обучение

Быстрое введение в Seq2Seq + Attention

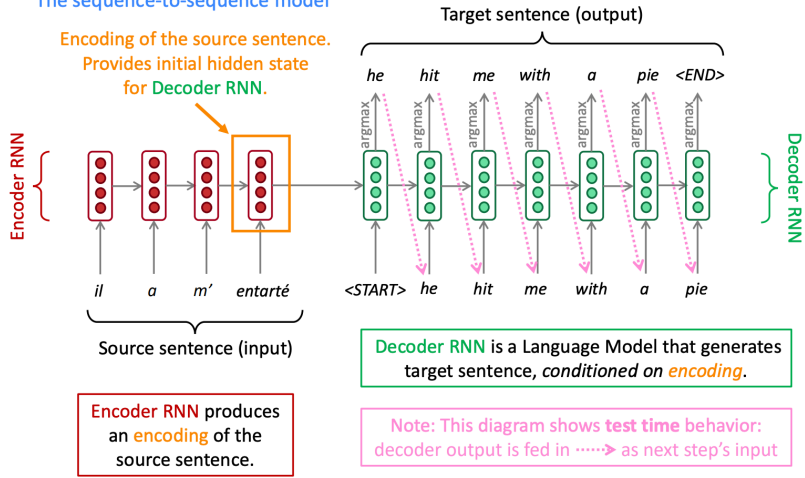
Neural Machine Translation: Seq2seq

Seq2seq

Seq2seq - это задача, в которой мы хотим предсказать по одной последовательности другую
Самая стандартная подобная задача - машинный перевод. Нейронные сети ворвались в эту сферу человеческого прогресса в 2014 году

Neural Machine Translation (NMT)

The sequence-to-sequence model



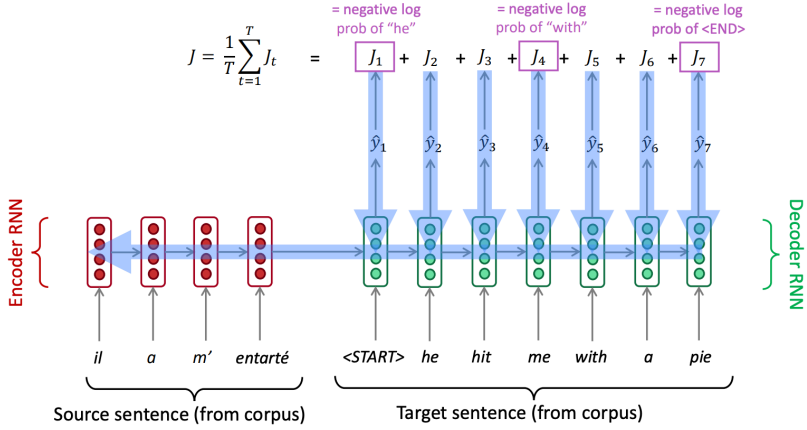
Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Sequence-to-sequence is versatile!

- Sequence-to-sequence is useful for *more than just MT*
- Many NLP tasks can be phrased as sequence-to-sequence:
  - Summarization (long text → short text)
  - Dialogue (previous utterances → next utterance)
  - Parsing (input text → output parse as sequence)
  - Code generation (natural language → Python code)

Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Training a Neural Machine Translation system

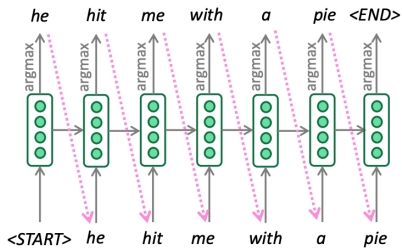


Seq2seq is optimized as a **single system.**
Backpropagation operates *"end-to-end"*.

Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Greedy decoding

- We saw how to generate (or “decode”) the target sentence by taking argmax on each step of the decoder



- This is greedy decoding (take most probable word on each step)
- **Problems with this method?**

Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Problems with greedy decoding

- Greedy decoding has no way to undo decisions!
  - Input: *il a m'entarté* *(he hit me with a pie)*
  - → *he* ____
  - → *he hit* ____
  - → *he hit a* ____ (whoops! no going back now…)

- How to fix this?

Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Beam search decoding

- Core idea: On each step of decoder, keep track of the *k* most probable partial translations (which we call *hypotheses*)
  - *k* is the beam size (in practice around 5 to 10)

- A hypothesis $y_1, \ldots, y_t$ has a score which is its log probability:

$$\text{score}(y_1, \ldots, y_t) = \log P_{\text{LM}}(y_1, \ldots, y_t | x) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$$

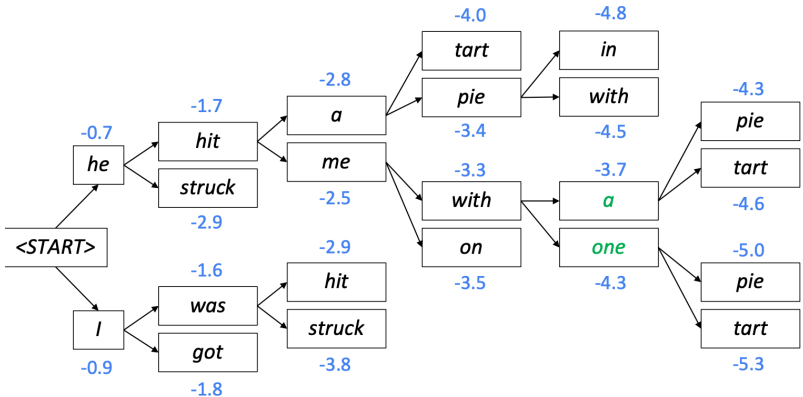
  - Scores are all negative, and higher score is better
  - We search for high-scoring hypotheses, tracking top *k* on each step

- Beam search is not guaranteed to find optimal solution
- But much more efficient than exhaustive search!

Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Beam search decoding: example

Beam size = k = 2. Blue numbers = $\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$



Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Beam search decoding: stopping criterion

- In greedy decoding, usually we decode until the model produces a <END> token
  - For example: *<START> he hit me with a pie <END>*
- In beam search decoding, different hypotheses may produce <END> tokens on different timesteps
  - When a hypothesis produces <END>, that hypothesis is complete.
  - Place it aside and continue exploring other hypotheses via beam search.
- Usually we continue beam search until:
  - We reach timestep *T* (where *T* is some pre-defined cutoff), or
  - We have at least *n* completed hypotheses (where *n* is pre-defined cutoff)

Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Beam search decoding: finishing up

- We have our list of completed hypotheses.
- How to select top one with highest score?

- Each hypothesis $y_1, \ldots, y_t$ on our list has a score

$$\text{score}(y_1, \ldots, y_t) = \log P_{\text{LM}}(y_1, \ldots, y_t | x) = \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$$

- Problem with this: longer hypotheses have lower scores

- Fix: Normalize by length. Use this to select top one instead:

$$\frac{1}{t} \sum_{i=1}^{t} \log P_{\text{LM}}(y_i | y_1, \ldots, y_{i-1}, x)$$

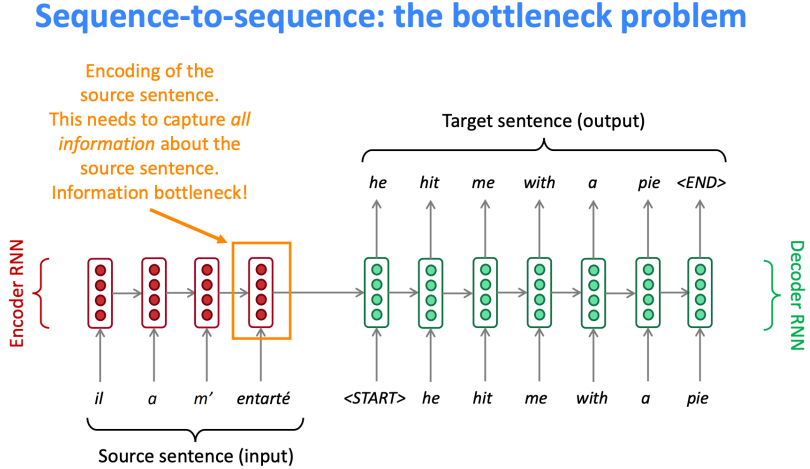
Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Attention!

Что можно улучшить в архитектуре Seq2seq?

Подсказка: Как связан decoder с encoder?

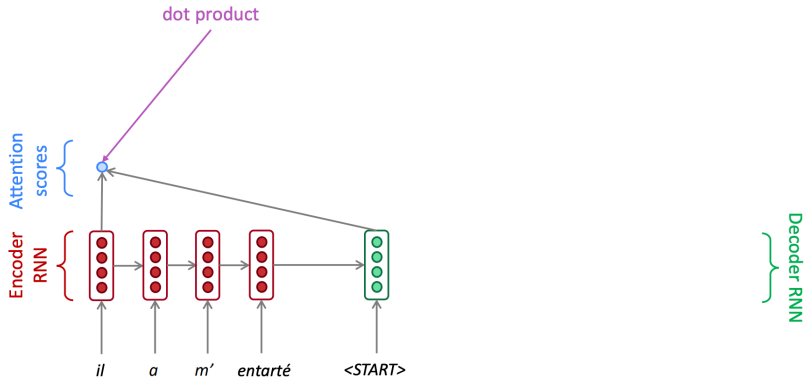
Attention



Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Attention

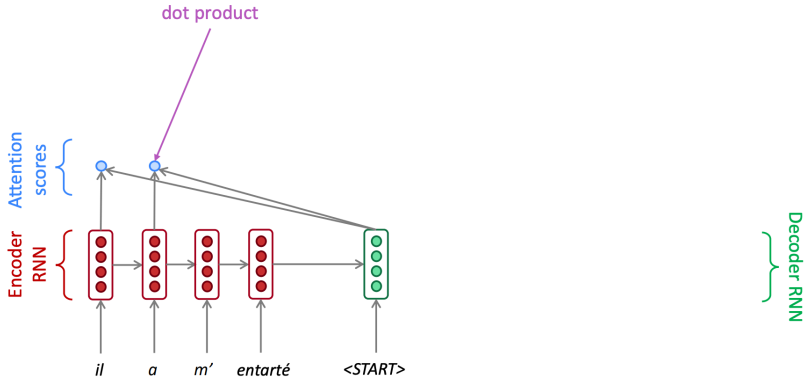
Sequence-to-sequence with attention



Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

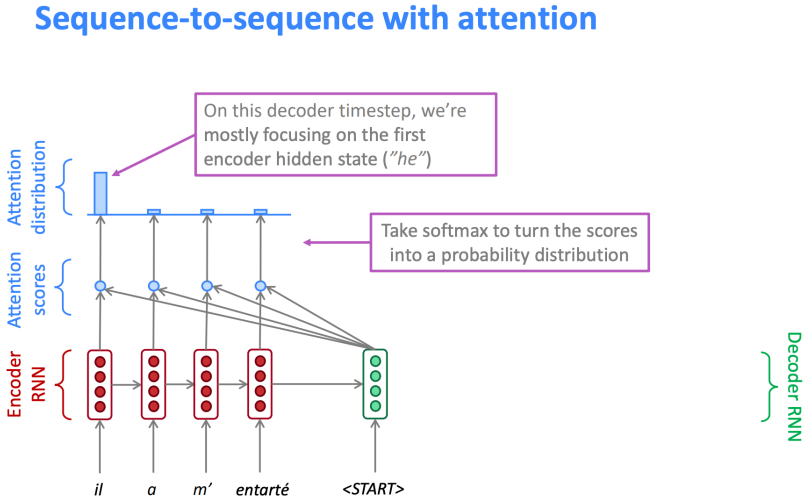
Attention

Sequence-to-sequence with attention



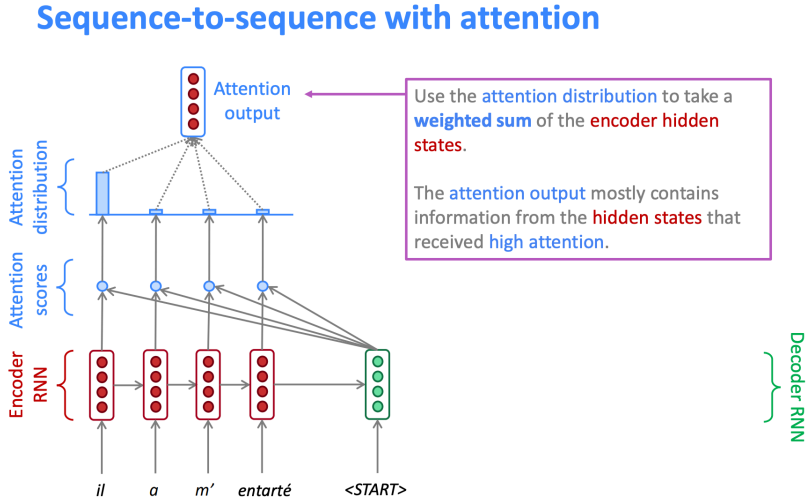
Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Attention



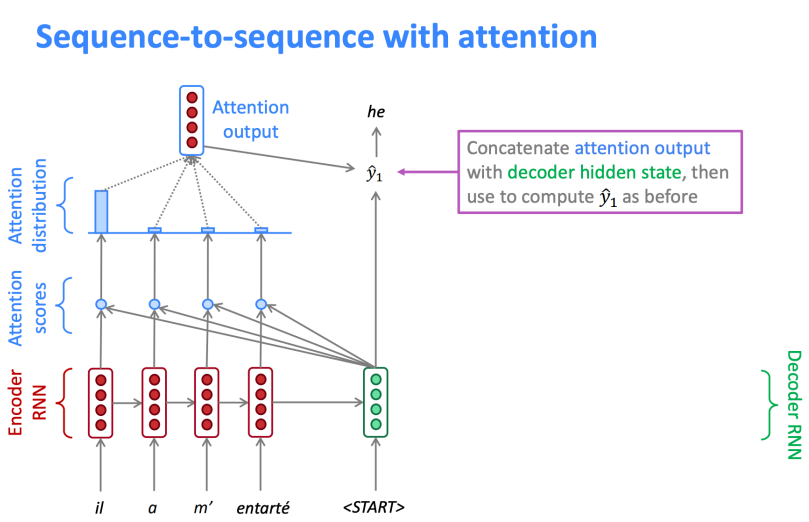
Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Attention



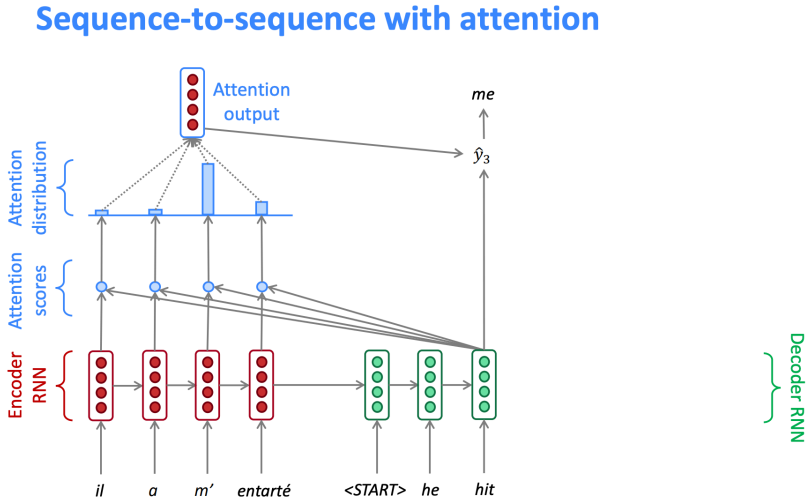
Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Attention



Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Attention



Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Интуиция за использованием Attention

- Взвешенная сумма - это выборочное выделение основной информации, хранящейся в словах исходного предложения. Выход с одной из ячеек декодера определяет на каких словах из исходного предложения нужно делать акцент.

- Attention - это способ получить представление о входных данных фиксированного размера из произвольного числа представлений в зависимости от рассматриваемого представления.

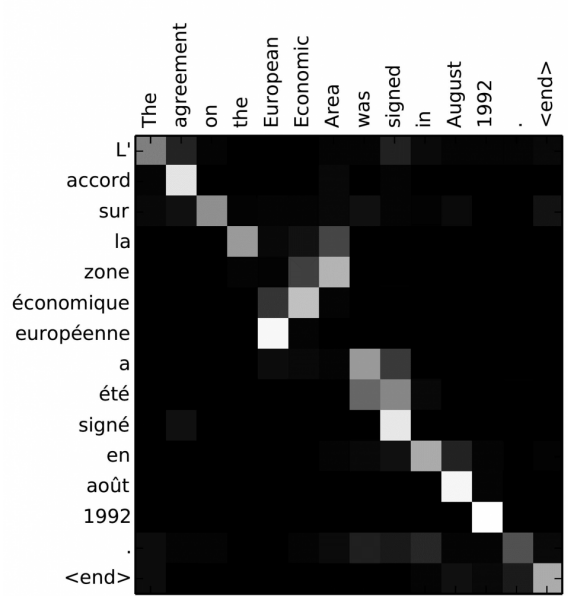
Варивции Attention

There are several ways you can compute $\boldsymbol{e} \in \mathbb{R}^N$ from $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and $\boldsymbol{s} \in \mathbb{R}^{d_2}$:

- Basic dot-product attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{h}_i \in \mathbb{R}$
  - Note: this assumes $d_1 = d_2$
  - This is the version we saw earlier
- Multiplicative attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{W} \boldsymbol{h}_i \in \mathbb{R}$
  - Where $\boldsymbol{W} \in \mathbb{R}^{d_2 \times d_1}$ is a weight matrix
- Additive attention: $\boldsymbol{e}_i = \boldsymbol{v}^T \tanh(\boldsymbol{W}_1 \boldsymbol{h}_i + \boldsymbol{W}_2 \boldsymbol{s}) \in \mathbb{R}$
  - Where $\boldsymbol{W}_1 \in \mathbb{R}^{d_3 \times d_1}, \boldsymbol{W}_2 \in \mathbb{R}^{d_3 \times d_2}$ are weight matrices and $\boldsymbol{v} \in \mathbb{R}^{d_3}$ is a weight vector.
  - d_3 (the attention dimensionality) is a hyperparameter

Источник: https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture08-nmt.pdf

Attention


The
agreement
on
the
European
Economic
Area
was
signed
in
August
1992
.
<end>
L'
accord
sur
la
zone
économique
européenne
a
été
signé
en
août
1992
.
<end>

Как оценивать качество Машинного перевода?

- Модели в машинном переводе сравнивают по BLEU score
- BLEU - это метрика сравнения полученного машиной перевода и человеческого. Насколько хорошо перевод совпадает с истинным предложением.
- Из проблем данной метрики: если машина перевела корректно по смыслу, но не так, как заложено в таргете, то BLEU будет низкий

BLEU

$$BLEU = \text{brevity penalty} \cdot \left(\prod_{i=1}^{n} \text{precision}_i \right)^{1/n} \cdot 100\%$$

$$\text{brevity penalty} = min \left(1, \frac{\text{output length}}{\text{reference length}} \right)$$

Источник: https://github.com/girafe-ai/ml-mipt/blob/master/week1_03_Machine_Translation_and_Attention/ml-mipt_f20_lect103_Machine_Tranlation.pdf

BLEU

BLEU (Bilingual Evaluation Understudy) compares the machine-written translation to human-written translation, and computes a similarity score based on:

- n-gram precision
- brevity penalty

Metric	System A	System B
precision (1gram)	3/6	6/6
precision (2gram)	1/5	4/5
precision (3gram)	0/4	2/4
precision (4gram)	0/3	1/3
brevity penalty	6/7	6/7
BLEU	0%	52%

$$BLEU = \text{brevity penalty} \cdot \left(\prod_{i=1}^{n} \text{precision}_i \right)^{1/n} \cdot 100\%$$

Источник: https://github.com/girafe-ai/ml-mipt/blob/master/week1_03_Machine_Translation_and_Attention/ml-mipt_f20_lect103_Machine_Tranlation.pdf

Краткий итог основной части лекции

1. Рассмотрели, что такое Machine translation
2. Познакомились с Seq2seq архитектурой
3. Поянли , что идея с вниманием является ключевой. С помощью нее можно тянуть информацию через всю последовательность токенов

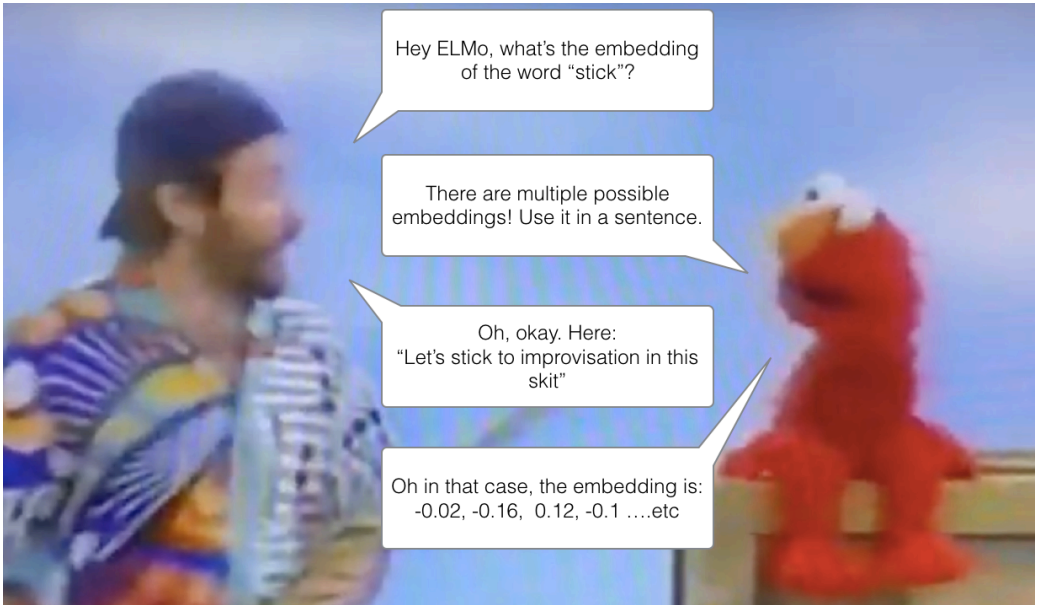
Маленькая мысль в конце: мы разобрали сложные архитектуры на составляющие запчасти и разобрались с тем, как их соединить воедино. Современный моделист в нейронных сетях - это скорее инженер, нежели аналитик

ELMO

Серия вопросов в зал

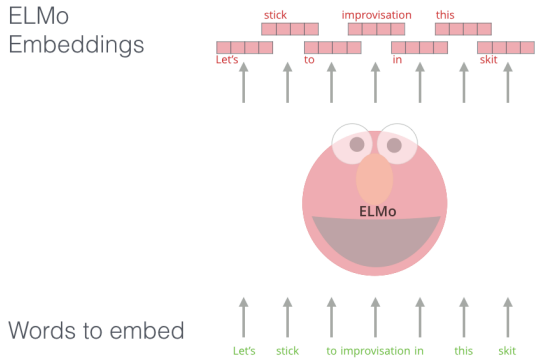
Как работают разные эмбединги?
В чем, по вашему мнению, их главная проблема?

Картиночка про решение проблемы



ELMO

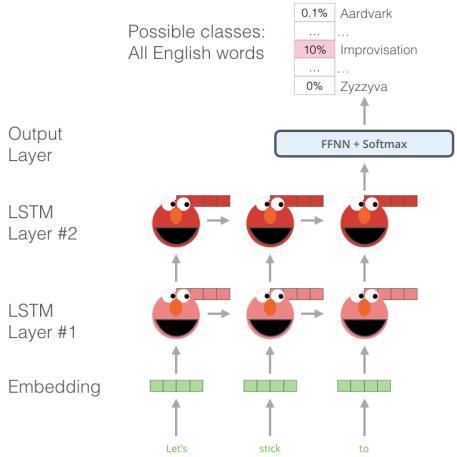
Захватываем контекст предложения через biderictional LSTM. Таким образом мы захватываем и контекст предложения (да, надо очень очень много данных, не обучайте это дома)



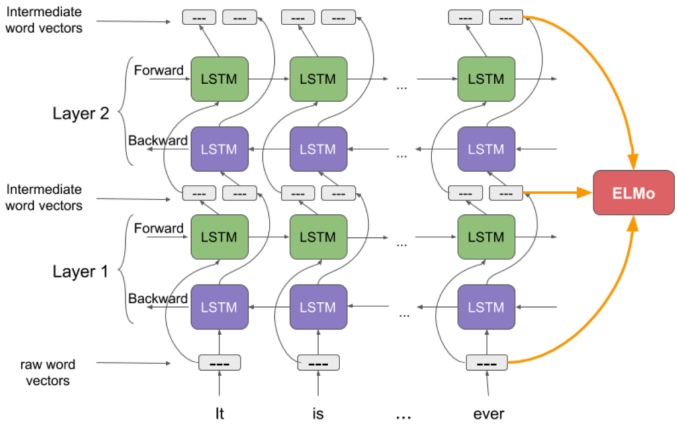
Рисунки на слайдах заимствованы отсюда

ELMO

Учится понимать язык ELMO следующим образом - оно берет большой дата сет и пытается предсказать следующее слово в предложении (Language Modeling task).



Архитектура ELMO



Source

ELMO

Применяем