Современные нейросетевые подходы в области обработки естественного языка

Мирас Амир

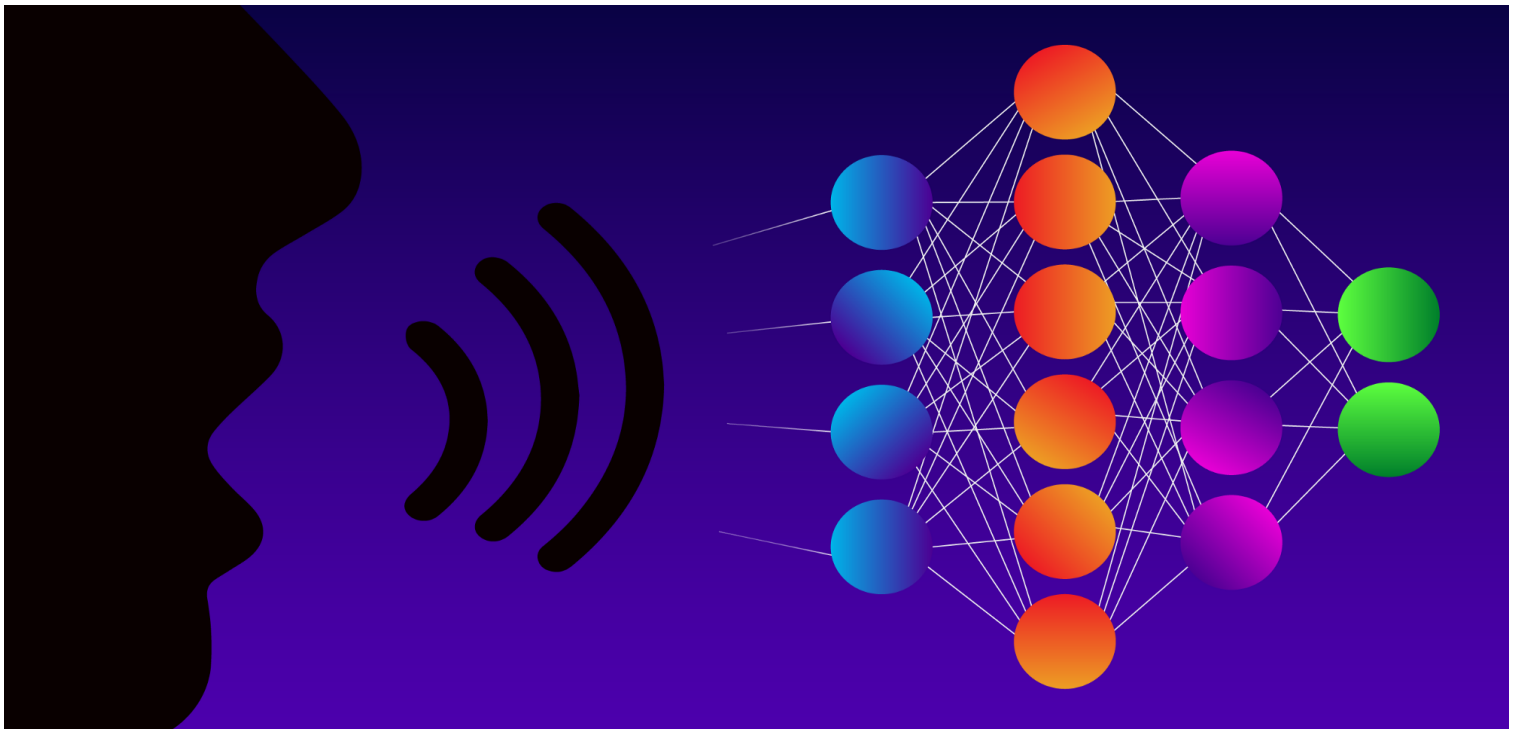
МГУ имени М.В. Ломоносова

15 октября 2018 г.

Содержание

Что такое NLP?

- NLP – это способ построения вычислительных алгоритмов для анализа, понимания и извлечения смысла с человеческого языка.



Почему это сложно?

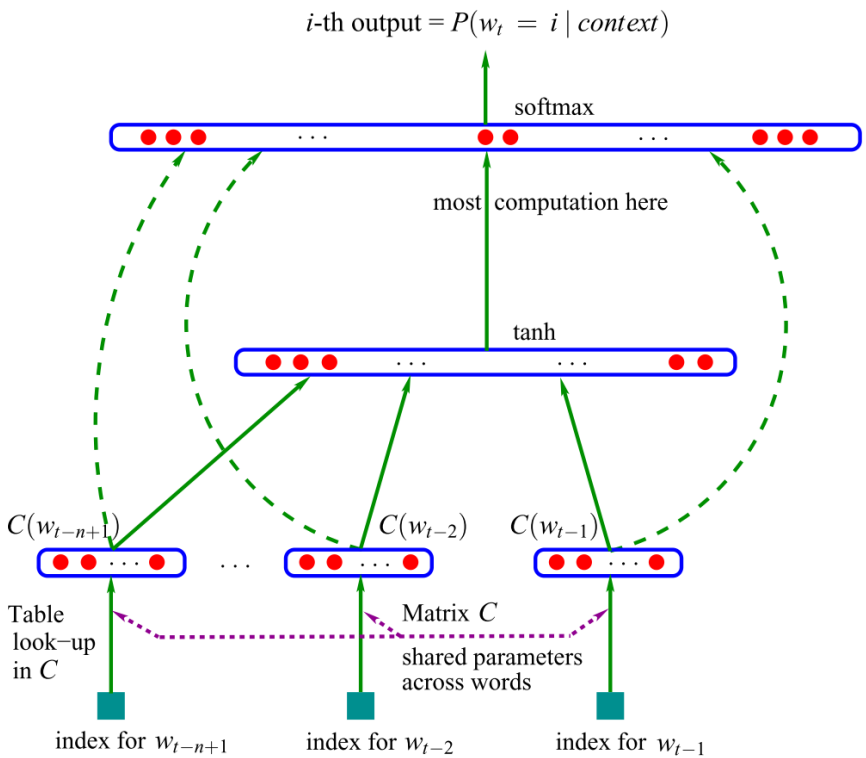
- Для понимания естественного языка нужно не просто правильным образом «распарсить» текст как последовательность букв или слов.
- Например, задача разрешение анафоры:
  - «Мама вымыла раму, и теперь она блестит».
  - «Мама вымыла раму, и теперь она устала».
  - К чему относится местоимение «она» в каждой из этих фраз?
- Нужно иметь «здравый смысл», представление об окружающем мире.

Примеры задач

- Задачи:
  - классификация текстов;
  - тематическое моделирование;
  - машинный перевод;
  - автоматическое реферирование;
  - диалоговые модели;
  - ответы на вопросы;
  - ...
- Сосредоточимся не на конкретных задачах, а на революции нейросетевых методов их решения.

2001 – Нейросетевое языковое моделирование

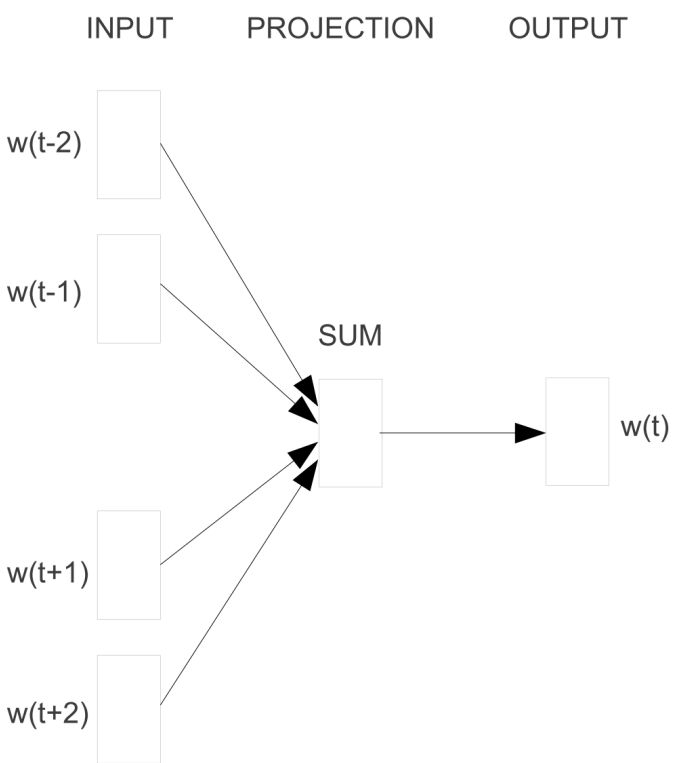
- Языковое моделирование: предсказывание следующего слова по предыдущим словам.
- Классический подход: n-граммы слов со сглаживанием.



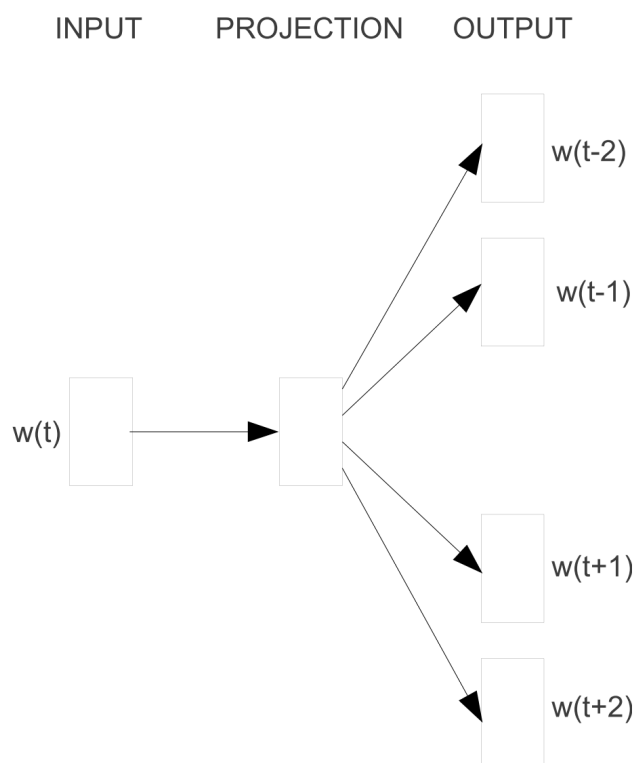
[Bengio et al., NIPS '01]

2013 – Word2vec

- Word2vec – эффективное обучение векторных представлений слов.
- Word2vec предложен в двух вариантах: skip-gram и CBOW.



CBOW



Skip-gram

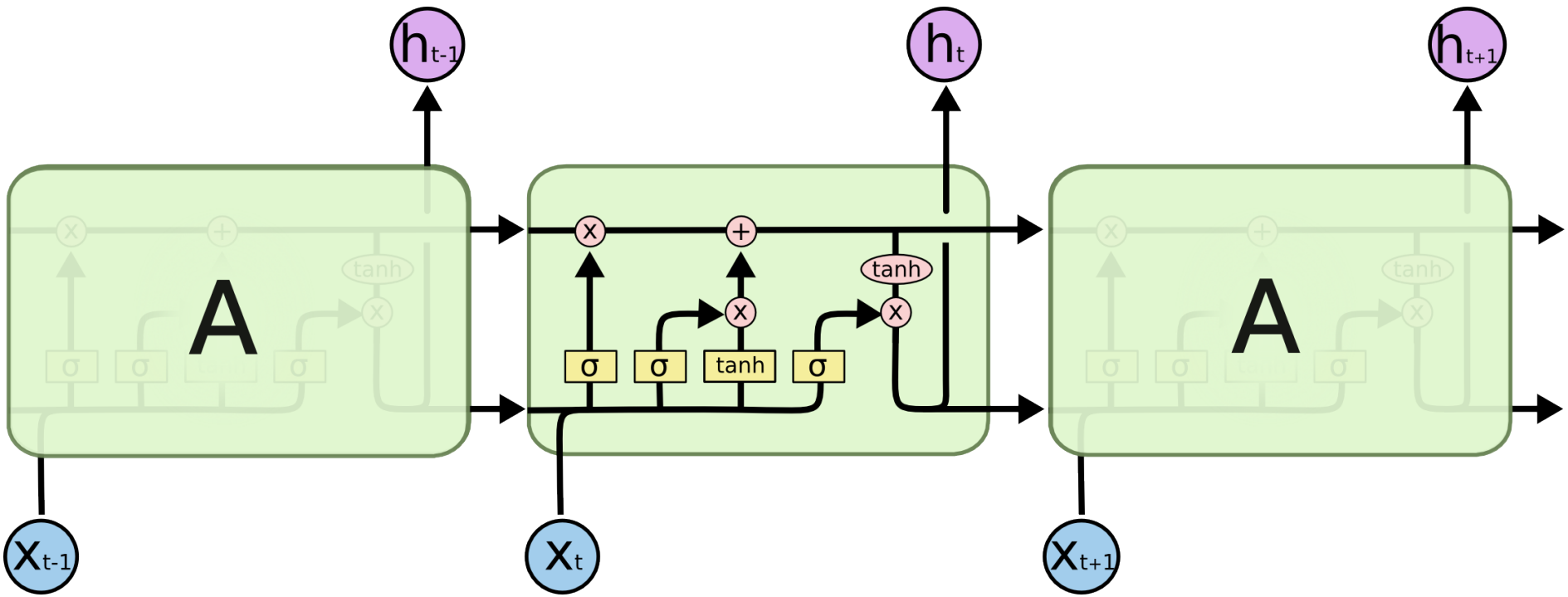
[Mikolov et al., ICLR '13; Mikolov et al., NIPS '13]

2013 – Нейронные сети

- Основная проблема для нейронных сетей – работа с динамическими входными последовательностями.
- Основные типы:
  - Рекуррентные нейронные сети.
  - Сверточные нейронные сети.

2013 – Рекуррентные нейронные сети

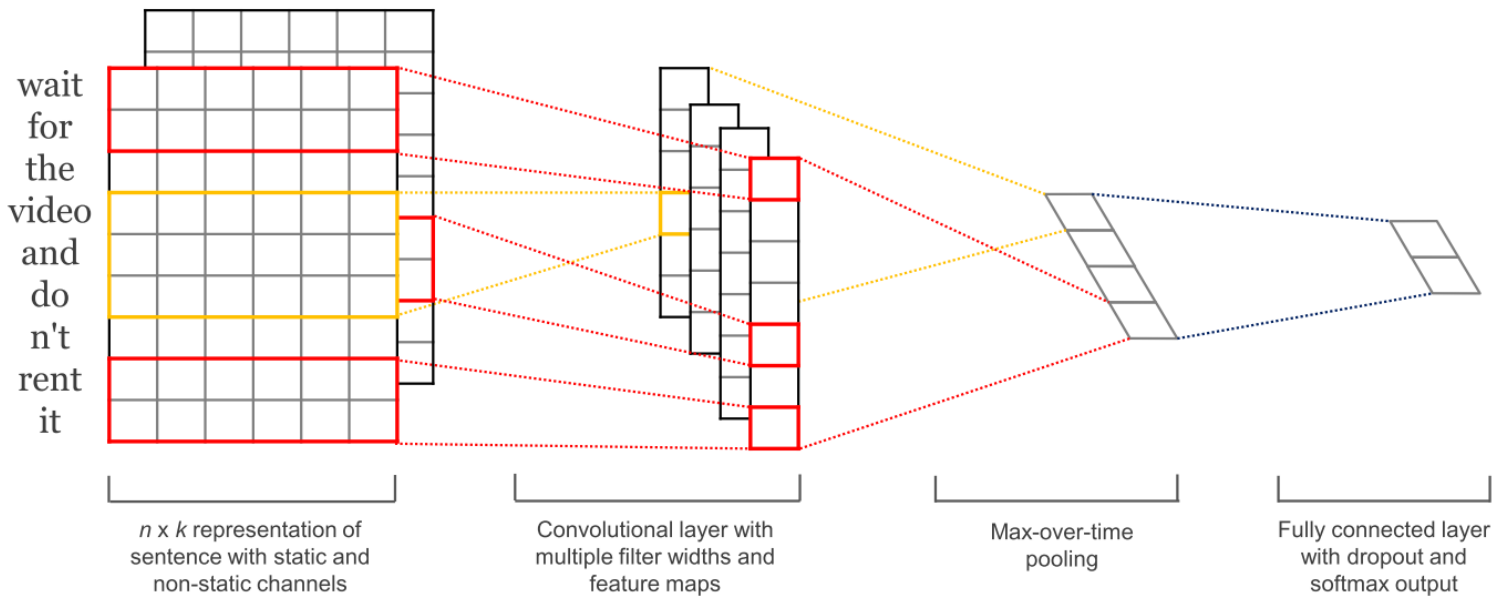
- Обычный RNN не используются, поскольку градиенты исчезают или взрываются для длинных входов.
- Решение: LSTM.



[Olah, '15]

2014 – Сверточные нейронные сети

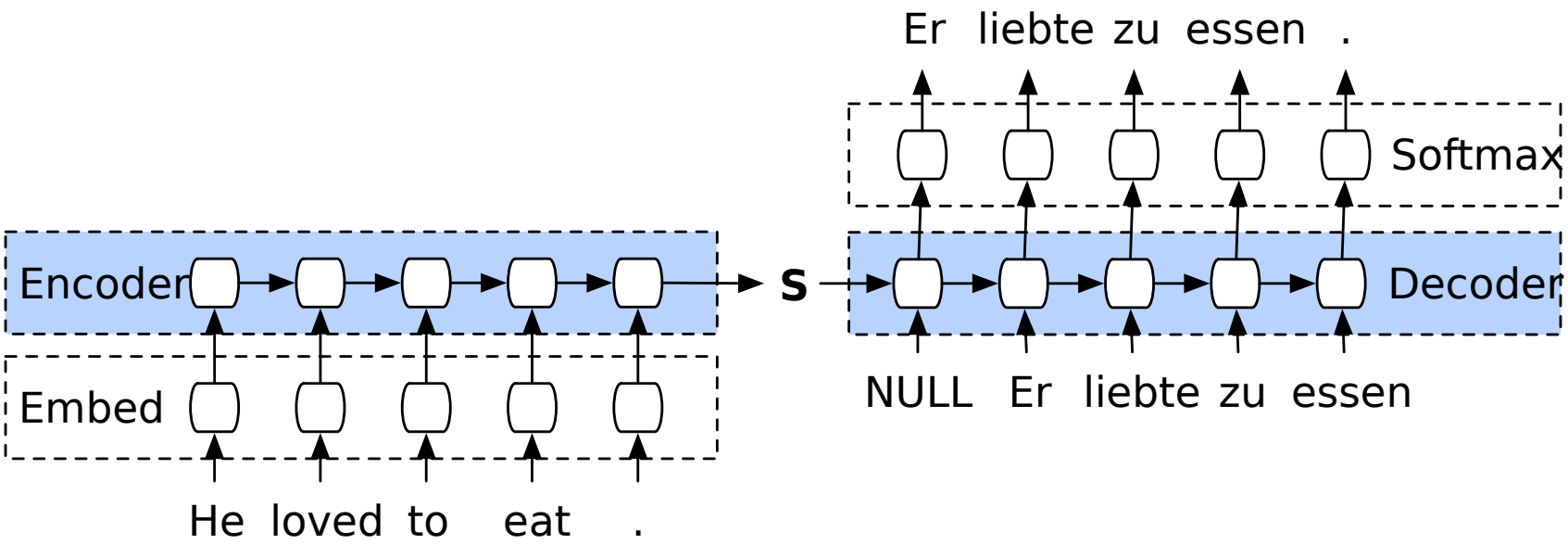
- Одномерная свертка, которая ходит только по ширине.
- Max-over-time pooling: max-pooling, примененный ко всей последовательности сразу.
- Легко распараллеливается по сравнению с RNN.
- Проблема: CNN могут работать только со входом фиксированного размера.



[Kim, EMNLP '14]

2014 – seq2seq

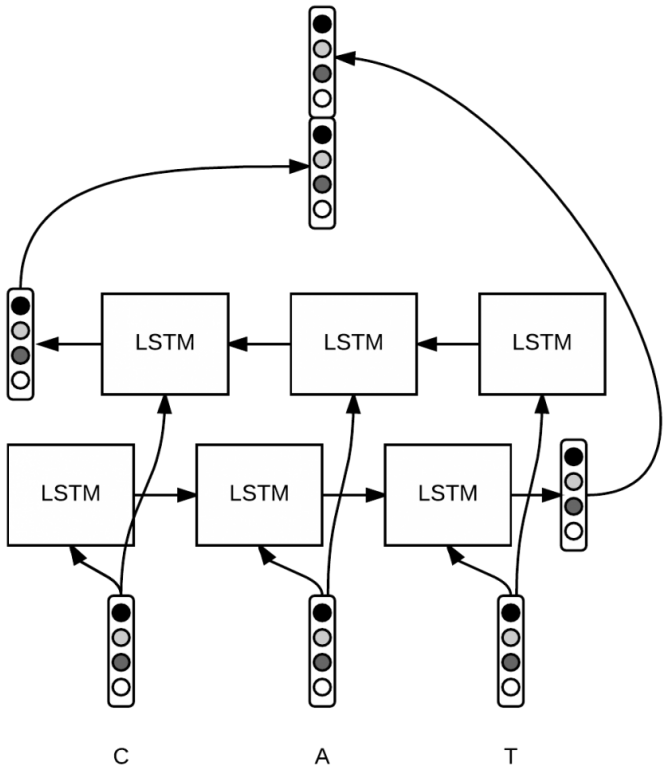
- Кодировщик обрабатывает вход слово за словом и сжимает его в векторное представление; затем декодировщик предсказывает выход слово за словом на основе состояния кодировщика.
- Основное приложение: машинный перевод.



[Sutskever et al., NIPS '14]

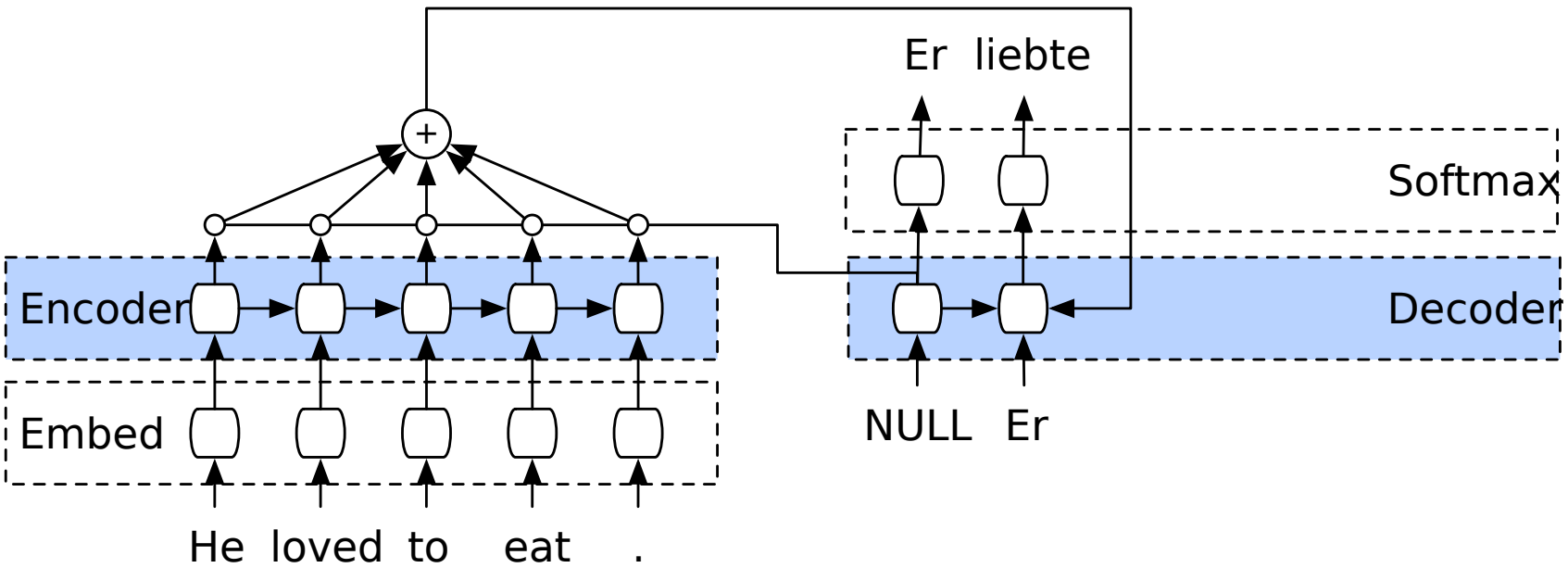
2014 – seq2seq

- Для сохранения контекста слова как слева, так и справа используется двунаправленный LSTM.
- Проблема: всё предложение целиком сворачивается в вектор фиксированной размерности.



2015 – Механизм внимания

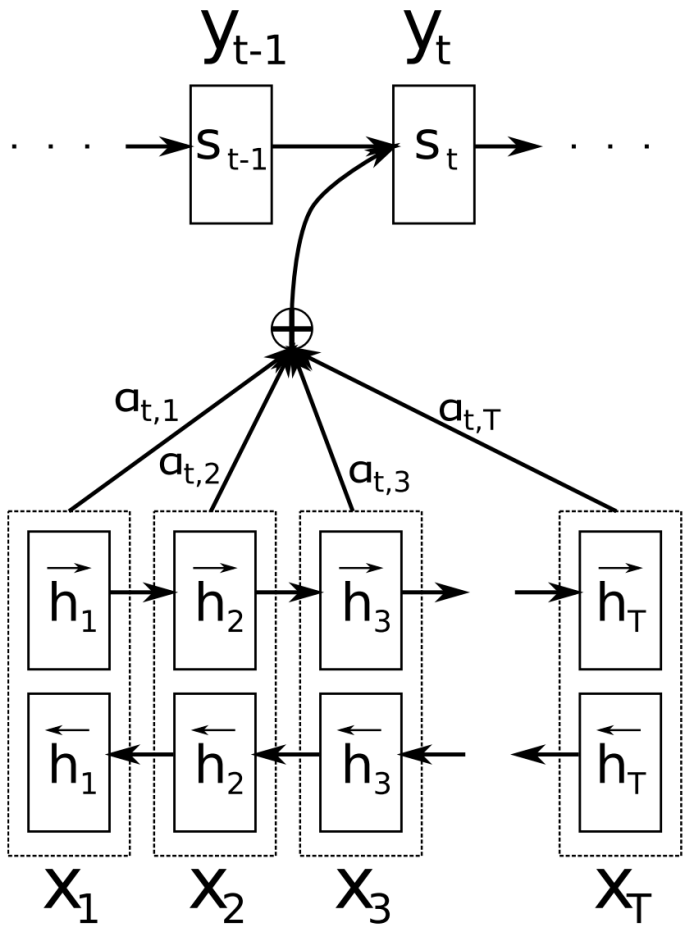
- Вместо того, чтобы создавать один вектор контекста из последнего скрытого состояния кодировщика, будем использовать взвешенную комбинацию всех входных состояний.



[Bahdanau et al., ICLR '15]

2015 – Механизм внимания

- Пусть входная последовательность $x = [x_1, x_2, \cdots, x_T]$, а выходная $y = [y_1, y_2, \cdots, y_M]$.
- $h_i = \left[\overrightarrow{h_i}, \overleftarrow{h_i}\right]$.
- Вектор контекста для выхода y_t: $c_t = \sum_{i=1}^{T} \alpha_{ti} h_i$.
- Насколько хорошо выровнены y_t и x_i: $\alpha_{ti} = \frac{score(s_{t-1}, h_i)}{\sum_{j=1}^{T} score\left(s_{t-1}, h_j\right)}$.
- $score\left(s_{t-1}, h_i\right) = v_a^T \tanh(W_a\left[s_t, h_i\right])$.



[Bahdanau et al., ICLR '15]

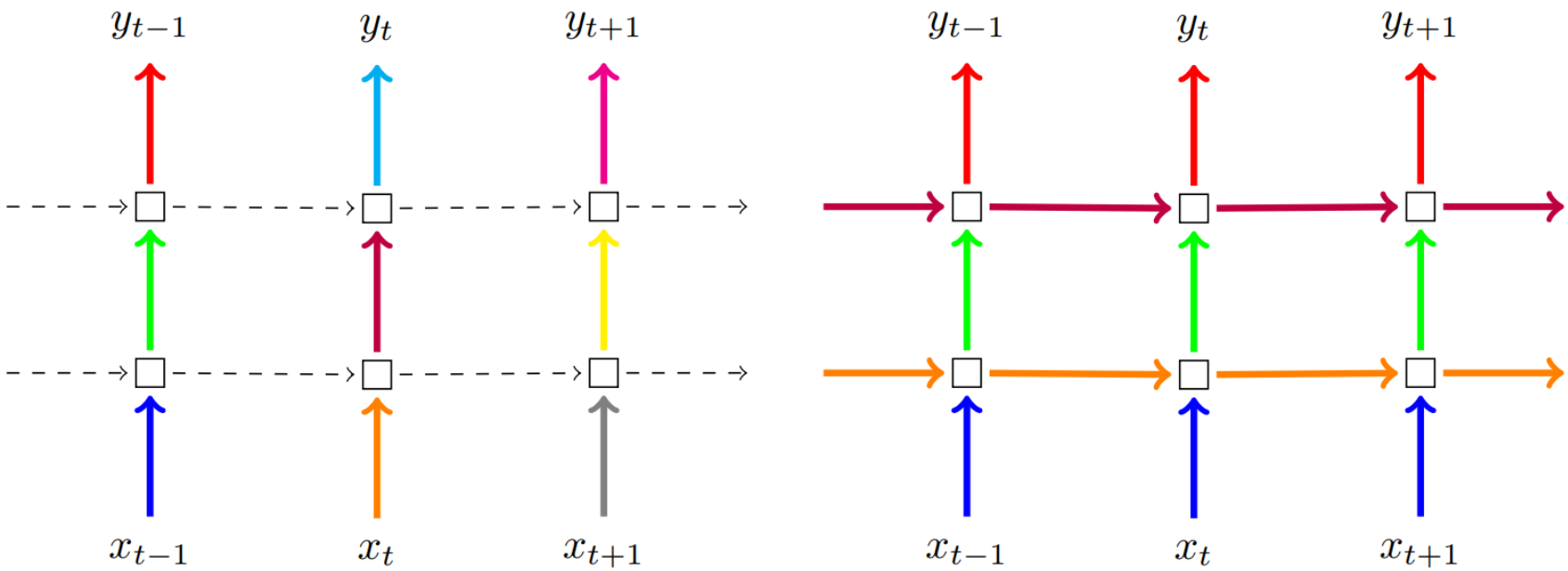
Regularizing and Optimizing LSTM Language Models

- Стратегии регуляризации, такие как dropout и batch normalization, не работают в случае рекуррентных нейронных сетей:
  - dropout нарушает способность RNN сохранять долгосрочные зависимости;
  - batch normalization усложняет модель.
- В статье предлагается набор эффективных стратегий регуляризации, которые могут быть использованы без изменения существующих реализаций LSTM.

[Merity et al., 2017]

Regularizing and Optimizing LSTM Language Models

- Обычный dropout использует разные маски выбрасывания на разных временных шагах, без выбрасывания на повторяющихся слоях.
- Вариационный dropout использует одну и ту же маску выбрасывания на каждом временном шаге, включая повторяющиеся слои.

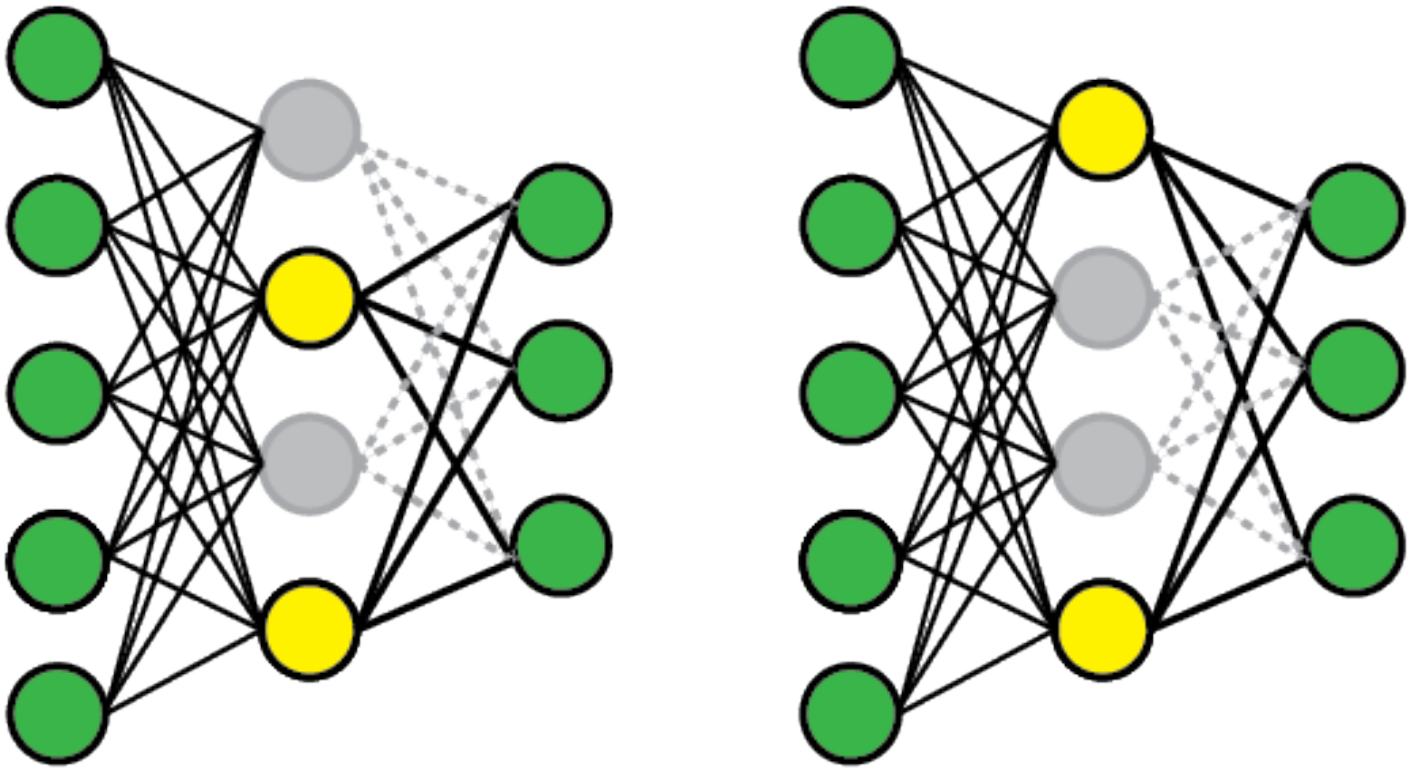


(a) Naive dropout RNN

(b) Variational RNN

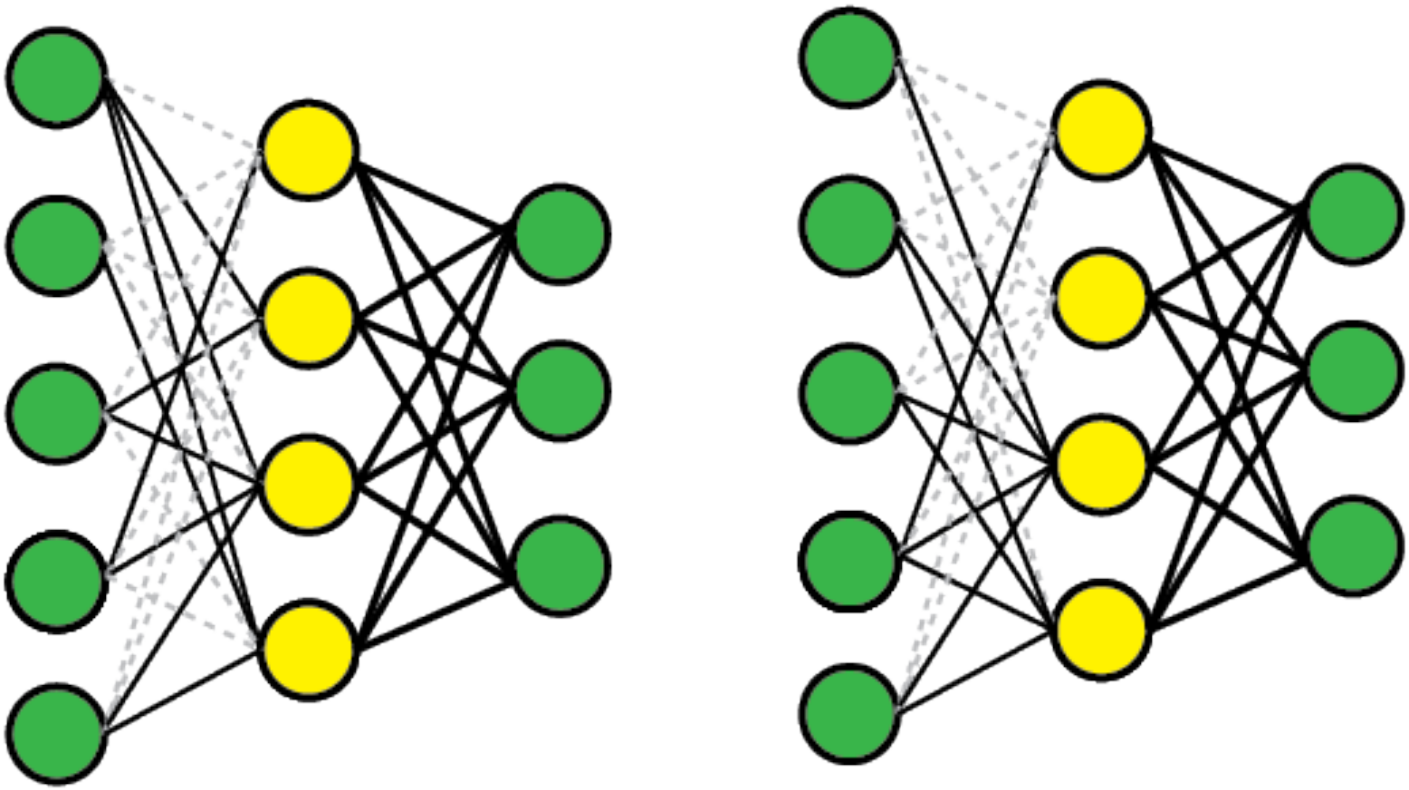
Regularizing and Optimizing LSTM Language Models

- Dropout



Regularizing and Optimizing LSTM Language Models

- DropConnect



Regularizing and Optimizing LSTM Language Models

- LSTM:
  - $i_t = \sigma\left(W^i x_t + U^i h_{t-1}\right)$
  - $f_t = \sigma\left(W^f x_t + U^f h_{t-1}\right)$
  - $o_t = \sigma\left(W^o x_t + U^o h_{t-1}\right)$
  - $\overline{c}_t = \tanh\left(W^c x_t + U^c h_{t-1}\right)$
  - $c_t = i_t \circ \overline{c}_t + f_t \circ c_{t-1}$
  - $h_t = o_t \circ \tanh\left(c_t\right)$
- Вариационный DropConnect (WeightDrop) для весов: $\left[U^i, U^f, U^o, U^c\right]$.
- Вариационный DropOut для весов: $\left[W^i, W^f, W^o, W^c\right]$.
- Embedding DropOut: эквивалентно удалению слов из словаря.
- Последовательности переменной длины.
- Weight tying: общие веса для embedding и softmax слоев.
- Activation Regularization (AR): $\alpha L_2(m \circ h_t)$.
- Temporal Activation Regularization (TAR): $\beta L_2(h_t - h_{t-1})$.

Regularizing and Optimizing LSTM Language Models

- Результаты (perplexity) на данных Penn Treebank:

Model	Parameters	Validation	Test
Mikolov & Zweig (2012) - KN-5	2M‡	–	141.2
Mikolov & Zweig (2012) - KN5 + cache	2M‡	–	125.7
Mikolov & Zweig (2012) - RNN	6M‡	–	124.7
Mikolov & Zweig (2012) - RNN-LDA	7M‡	–	113.7
Mikolov & Zweig (2012) - RNN-LDA + KN-5 + cache	9M‡	–	92.0
Zaremba et al. (2014) - LSTM (medium)	20M	86.2	82.7
Zaremba et al. (2014) - LSTM (large)	66M	82.2	78.4
Gal & Ghahramani (2016) - Variational LSTM (medium)	20M	81.9 ± 0.2	79.7 ± 0.1
Gal & Ghahramani (2016) - Variational LSTM (medium, MC)	20M	–	78.6 ± 0.1
Gal & Ghahramani (2016) - Variational LSTM (large)	66M	77.9 ± 0.3	75.2 ± 0.2
Gal & Ghahramani (2016) - Variational LSTM (large, MC)	66M	–	73.4 ± 0.0
Kim et al. (2016) - CharCNN	19M	–	78.9
Merity et al. (2016) - Pointer Sentinel-LSTM	21M	72.4	70.9
Grave et al. (2016) - LSTM	–	–	82.3
Grave et al. (2016) - LSTM + continuous cache pointer	–	–	72.1
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	24M	75.7	73.2
Inan et al. (2016) - Variational LSTM (tied) + augmented loss	51M	71.1	68.5
Zilly et al. (2016) - Variational RHN (tied)	23M	67.9	65.4
Zoph & Le (2016) - NAS Cell (tied)	25M	–	64.0
Zoph & Le (2016) - NAS Cell (tied)	54M	–	62.4
Melis et al. (2017) - 4-layer skip connection LSTM (tied)	24M	60.9	58.3
AWD-LSTM - 3-layer LSTM (tied)	24M	60.0	57.3
AWD-LSTM - 3-layer LSTM (tied) + continuous cache pointer	24M	53.9	52.8

Regularizing and Optimizing LSTM Language Models

- Результаты (perplexity) на данных WikiText-2:

Model	Parameters	Validation	Test
Inan et al. (2016) - Variational LSTM (tied) ($h = 650$)	28M	92.3	87.7
Inan et al. (2016) - Variational LSTM (tied) ($h = 650$) + augmented loss	28M	91.5	87.0
Grave et al. (2016) - LSTM	–	–	99.3
Grave et al. (2016) - LSTM + continuous cache pointer	–	–	68.9
Melis et al. (2017) - 1-layer LSTM (tied)	24M	69.3	65.9
Melis et al. (2017) - 2-layer skip connection LSTM (tied)	24M	69.1	65.9
AWD-LSTM - 3-layer LSTM (tied)	33M	68.6	65.8
AWD-LSTM - 3-layer LSTM (tied) + continuous cache pointer	33M	53.8	52.0

Regularizing and Optimizing LSTM Language Models

- Результаты при удалении каждой формы регуляризации:

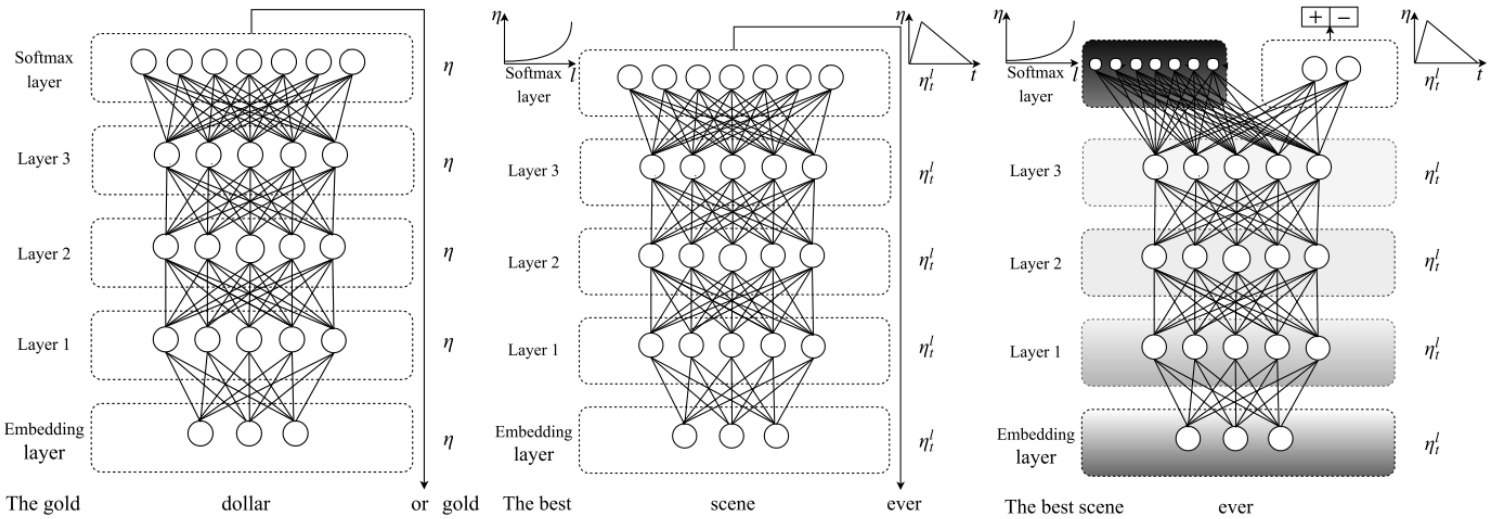
Model	PTB Validation	PTB Test	WT2 Validation	WT2 Test
AWD-LSTM (tied)	60.0	57.3	68.6	65.8
– fine-tuning	60.7	58.8	69.1	66.0
– NT-ASGD	66.3	63.7	73.3	69.7
– variable sequence lengths	61.3	58.9	69.3	66.2
– embedding dropout	65.1	62.7	71.1	68.1
– weight decay	63.7	61.0	71.9	68.7
– AR/TAR	62.7	60.3	73.2	70.1
– full sized embedding	68.0	65.6	73.7	70.7
– weight-dropping	71.1	68.9	78.4	74.9

Universal Language Model Fine-tuning for Text Classification

- Transfer learning сильно повлияло на компьютерное зрение, но плохо изучено в NLP.
- В статье предлагается предобученная модель ULMFiT, которая может быть применена к любой задаче в NLP.
- Метод превосходит state-of-the-art результаты на шести задачах классификации текстов.

[Howard et al., 2018]

Universal Language Model Fine-tuning for Text Classification

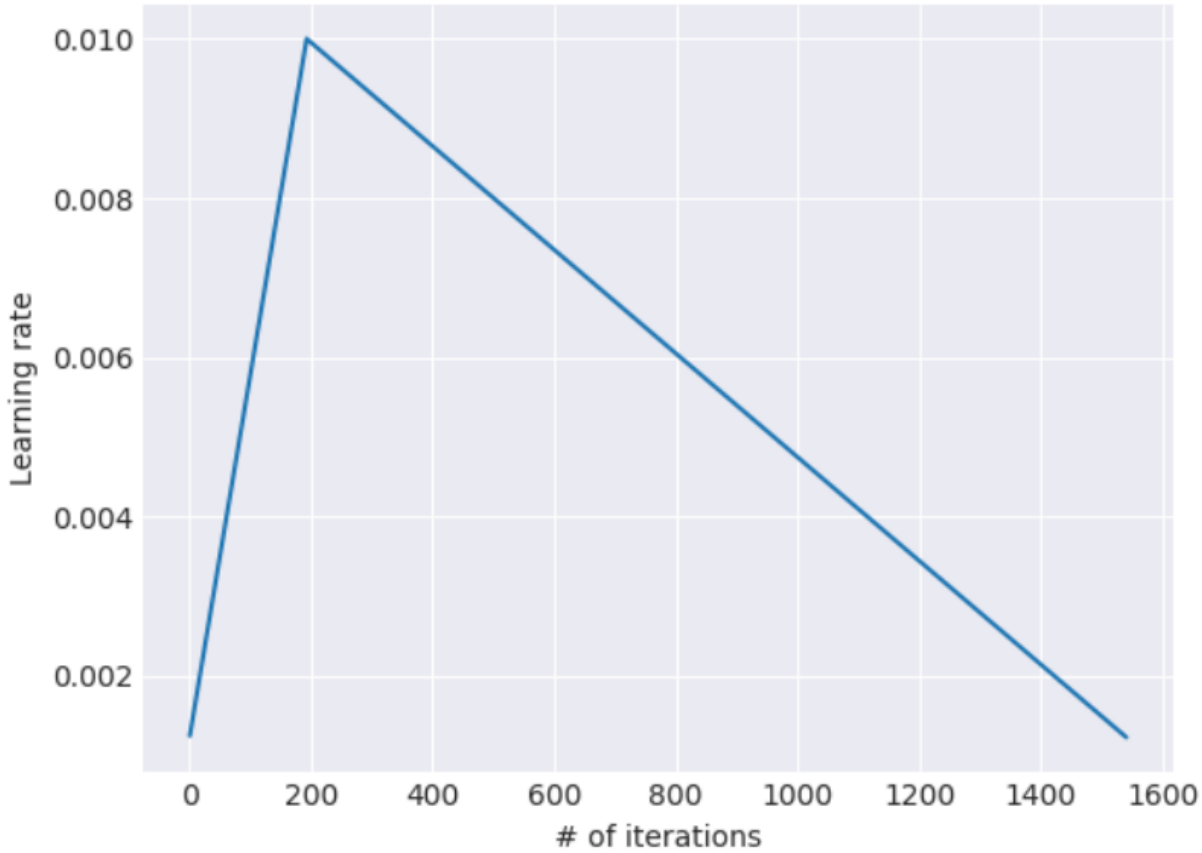


(a) LM pre-training (b) LM fine-tuning (c) Classifier fine-tuning

- AWD-LSTM с алгоритмом оптимизации SGD.
- Модель предобучается на задаче языковой модели на датасете Wikitext-103 из 103 млн. слов.
- Модель дообучается на данных целевой задачи.
- Полученная модель дообучается уже на целевой задаче классификации.

Universal Language Model Fine-tuning for Text Classification

- Slanted triangular learning rates (1cycle):



Universal Language Model Fine-tuning for Text Classification

- Discriminative fine-tuning: различные параметры learning rate для различных слоев сети (чем глубже слой, тем больше learning rate).
- Concat pooling: для классификации используется не только последнее скрытое состояние h_T, а среднее и максимум по всем скрытым состояниям $H = \{h_1, ..., h_T\}$:
  - $h_c = [h_T, maxpool(H), meanpool(H)]$;
- Gradual unfreezing: постепенно размораживаются слои начиная с последнего слоя, поскольку она содержит наименее общие знания и информация.

Universal Language Model Fine-tuning for Text Classification

- Результаты:

	Model	Test		Model	Test
IMDb	CoVe (McCann et al., 2017)	8.2	TREC-6	CoVe (McCann et al., 2017)	4.2
	oh-LSTM (Johnson and Zhang, 2016)	5.9		TBCNN (Mou et al., 2015)	4.0
	Virtual (Miyato et al., 2016)	5.9		LSTM-CNN (Zhou et al., 2016)	3.9
	ULMFiT (ours)	**4.6**		ULMFiT (ours)	**3.6**

Table 2: Test error rates (%) on two text classification datasets used by McCann et al. (2017).

	AG	DBpedia	Yelp-bi	Yelp-full
Char-level CNN (Zhang et al., 2015)	9.51	1.55	4.88	37.95
CNN (Johnson and Zhang, 2016)	6.57	0.84	2.90	32.39
DPCNN (Johnson and Zhang, 2017)	6.87	0.88	2.64	30.58
ULMFiT (ours)	**5.01**	**0.80**	**2.16**	**29.98**

Table 3: Test error rates (%) on text classification datasets used by Johnson and Zhang (2017).

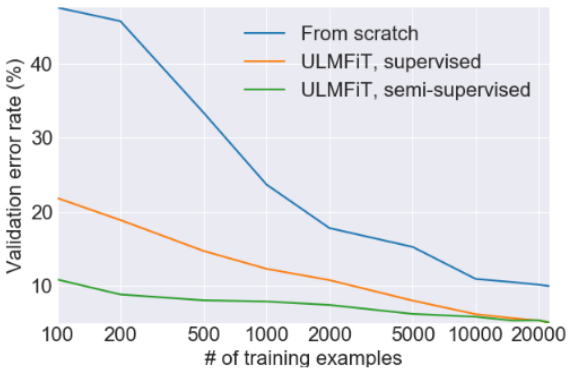
Universal Language Model Fine-tuning for Text Classification

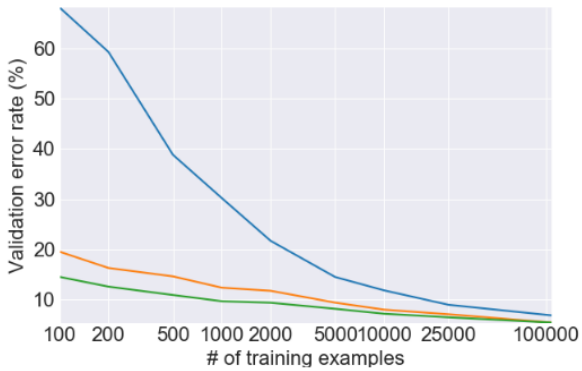
- Сравнение предобученной модели с моделью без предобучения:

Pretraining	IMDb	TREC-6	AG
Without pretraining	5.63	10.67	5.52
With pretraining	**5.00**	**5.69**	**5.38**

Universal Language Model Fine-tuning for Text Classification

- Зависимость ошибки на данных IMDb, TREC-6 и AG (слева направо) от количества данных в обучении:



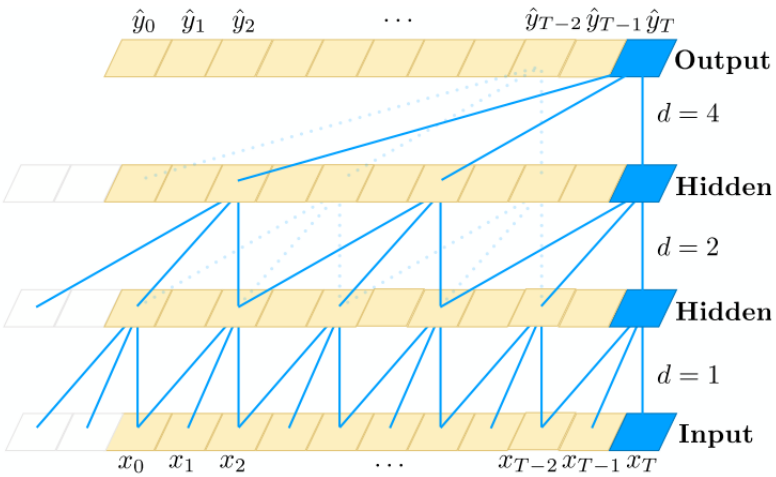


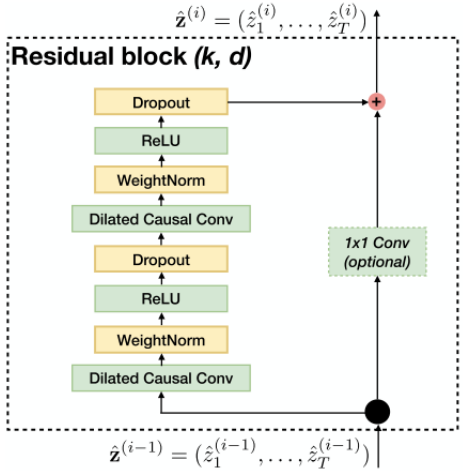
An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling

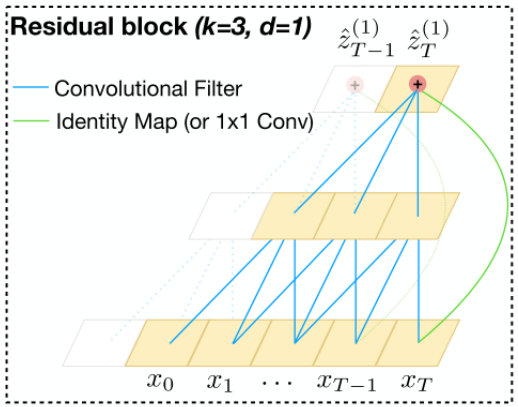
- Утверждается, что сверточные нейронные сети подходят для задач NLP лучше RNN.
- Предлагаются методы обучения глубоких временных сверточных сетей (TCN).

[Bai et al., 2018]

An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling







- Causal convolutions: выход в момент времени t зависит только от элементов из времени t и ранее в предыдущем слое.
- Dilated Convolutions: экспоненциальный receptive field по глубине сети.
- Residual Connections: способ проброса градиентов в глубину, не проходя через нелинейные функции активации.

An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling

- В итоге:
  - Параллелизм.
  - Гибкий размер receptive field.
  - Стабильные градиенты.
  - Входы переменной длины.

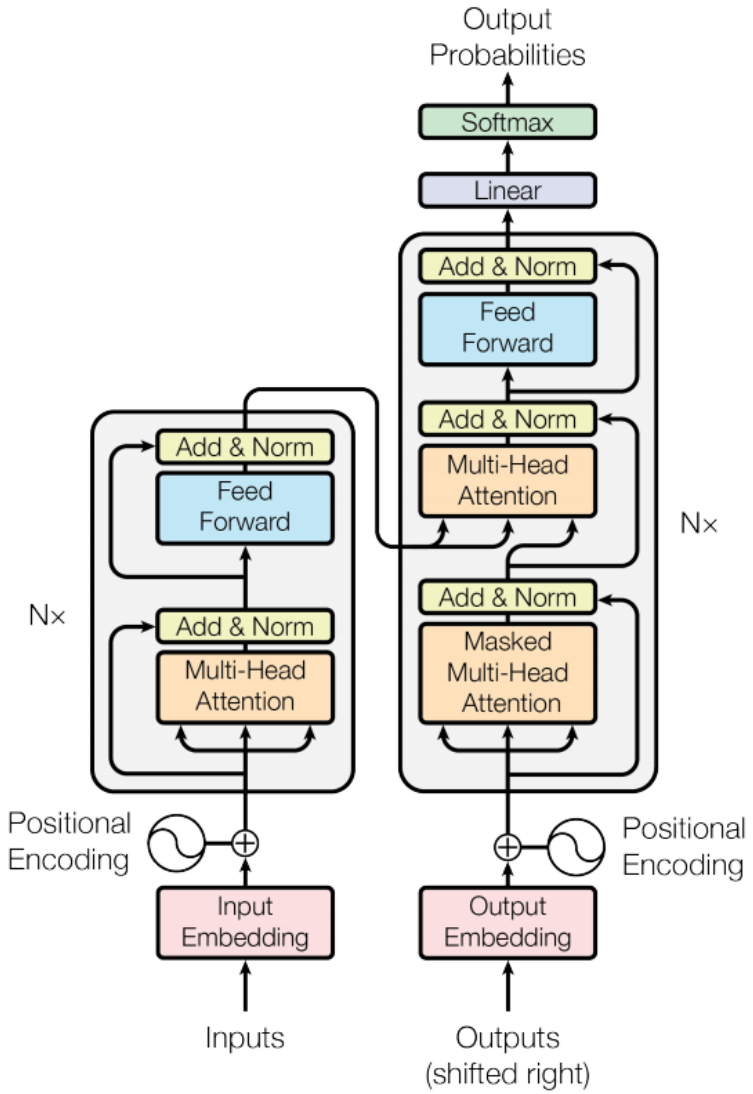
An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling

- Результаты на задаче языкового моделирования:

Sequence Modeling Task	Model Size ($\approx$)	Models			
		LSTM	GRU	RNN	**TCN**
Seq. MNIST (accuracy^h)	70K	87.2	96.2	21.5	**99.0**
Permuted MNIST (accuracy)	70K	85.7	87.3	25.3	**97.2**
Adding problem T=600 (loss$^\ell$)	70K	0.164	**5.3e-5**	0.177	**5.8e-5**
Copy memory T=1000 (loss)	16K	0.0204	0.0197	0.0202	**3.5e-5**
Music JSB Chorales (loss)	300K	8.45	8.43	8.91	**8.10**
Music Nottingham (loss)	1M	3.29	3.46	4.05	**3.07**
Word-level PTB (perplexity$^\ell$)	13M	**78.93**	92.48	114.50	88.68
Word-level Wiki-103 (perplexity)	-	48.4	-	-	**45.19**
Word-level LAMBADA (perplexity)	-	4186	-	14725	**1279**
Char-level PTB (bpc$^\ell$)	3M	1.36	1.37	1.48	**1.31**
Char-level text8 (bpc)	5M	1.50	1.53	1.69	**1.45**

Attention is all you need

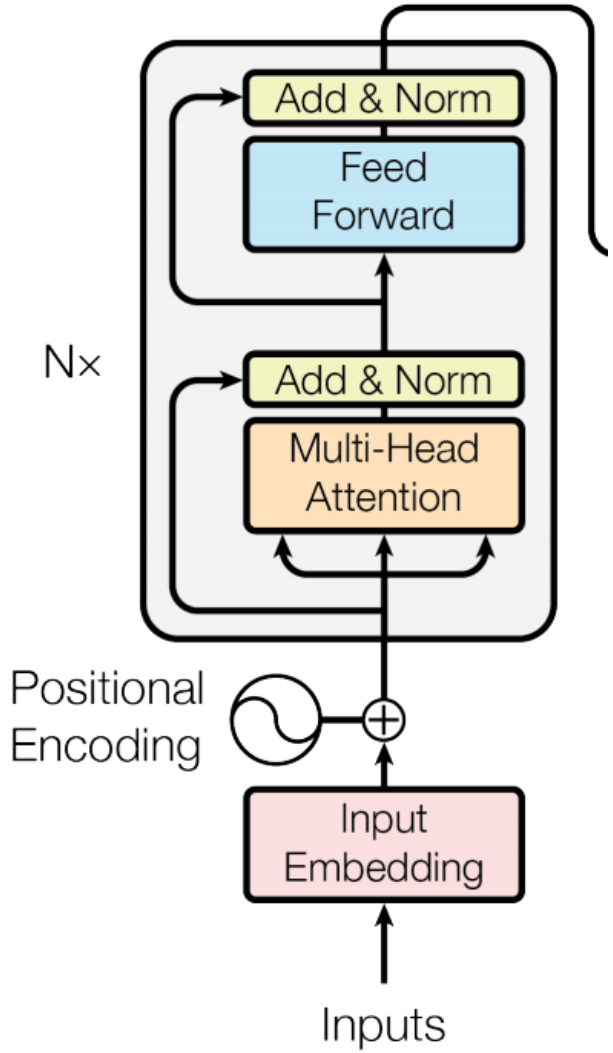
- Предлагается новая архитектура для решения задачи машинного перевода, которая не является ни RNN, ни CNN.



[Vaswani et al., 2017]

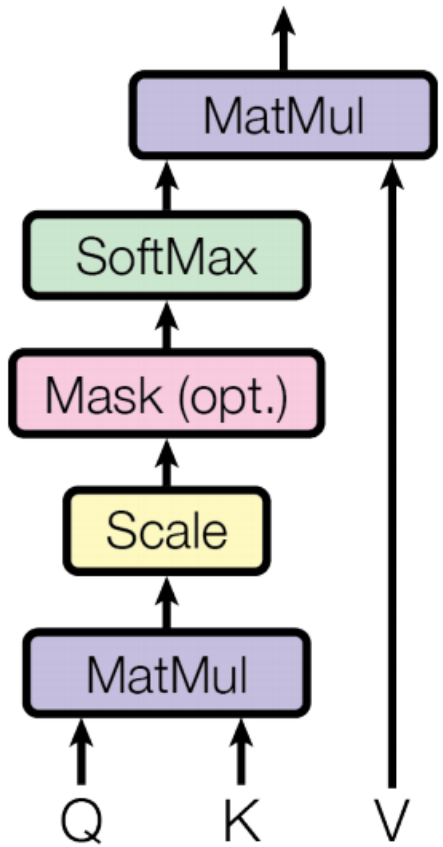
Attention is all you need

- Каждое слово параллельно проходит через слои кодировщика:
  - Каждый слой состоит из **multi-head attention** и полносвязной сети.
  - Каждый подслой использует residual connections и layer normalization.
  - Стекаем $N = 6$ таких слоев.



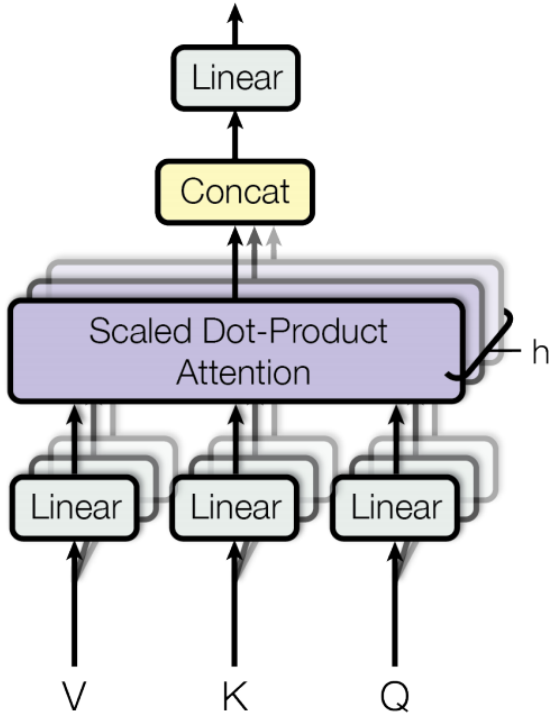
Attention is all you need

- Scaled dot-product attention: $Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{n}})V$.
- K и V – это всегда один и тот же вектор.



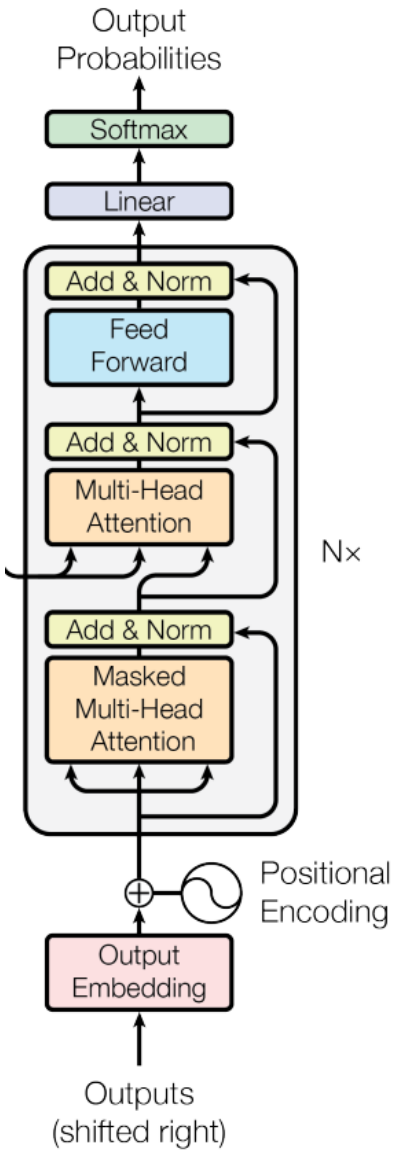
Attention is all you need

- Multi-head attention: специальный новый слой, который дает возможность каждому входному вектору взаимодействовать с другими словами через attention mechanism.
- $MultiHead(Q, K, V) = [head_1, \cdots, head_h]W^O$, где $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$



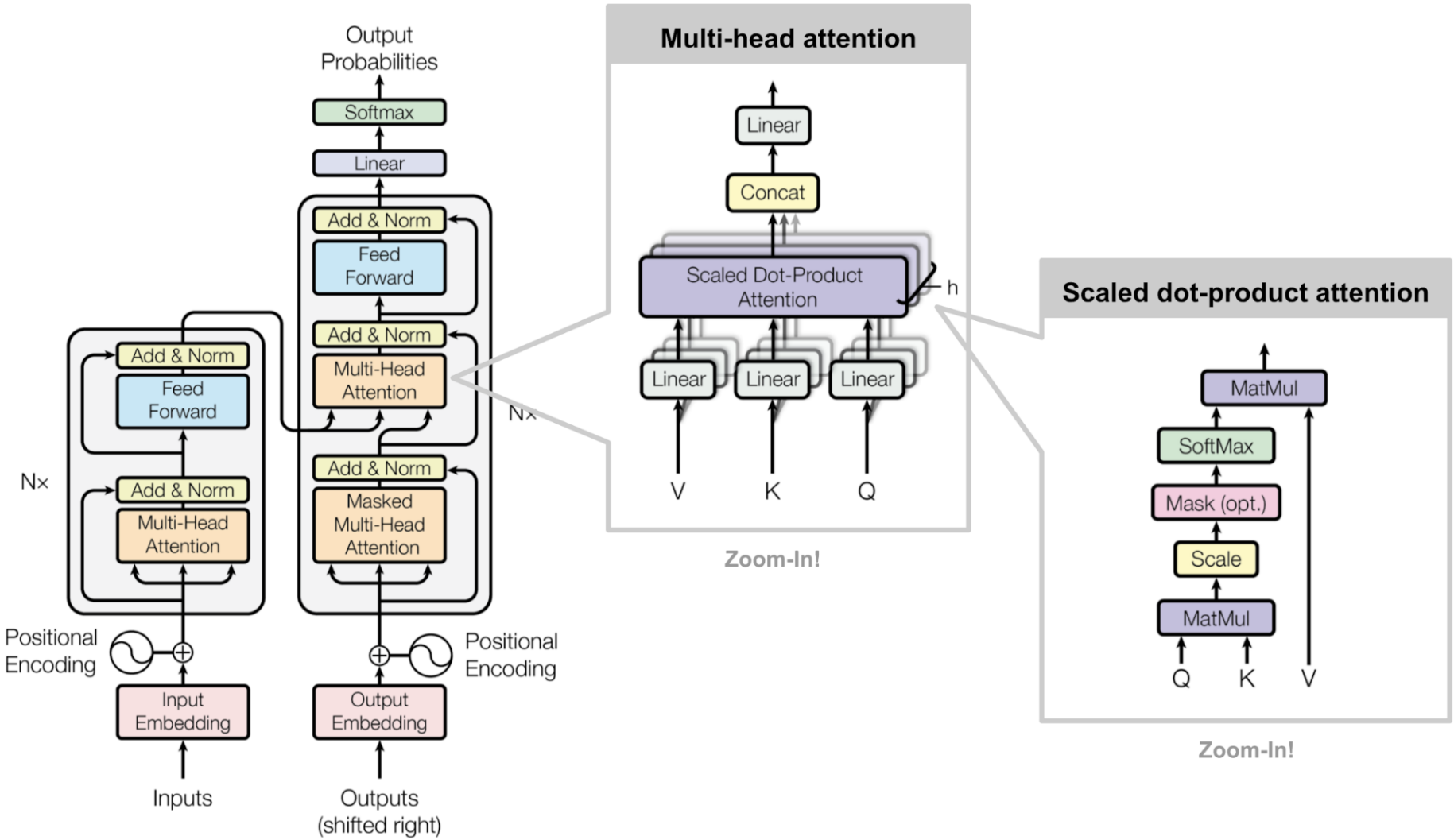
Attention is all you need

- Декодировщик запускается по слову за раз: получает на вход прошлое слово и должен выдать следующее.
- Два типа использования Multi-head attention:
  - Возможность обратиться к векторам прошлых декодированных слов.
  - Возможность обратиться к выходу кодировщика.



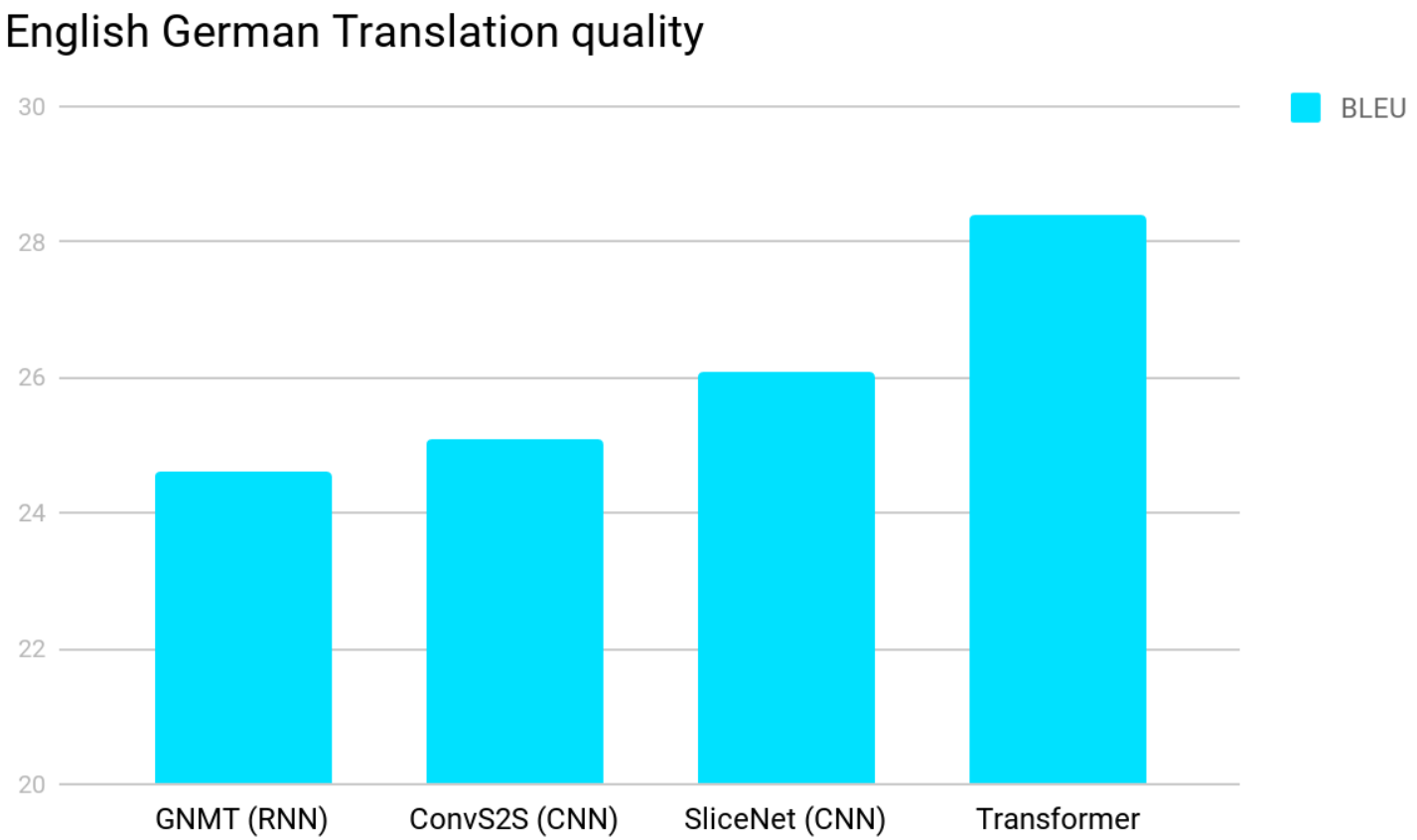
Attention is all you need

- В итоге:



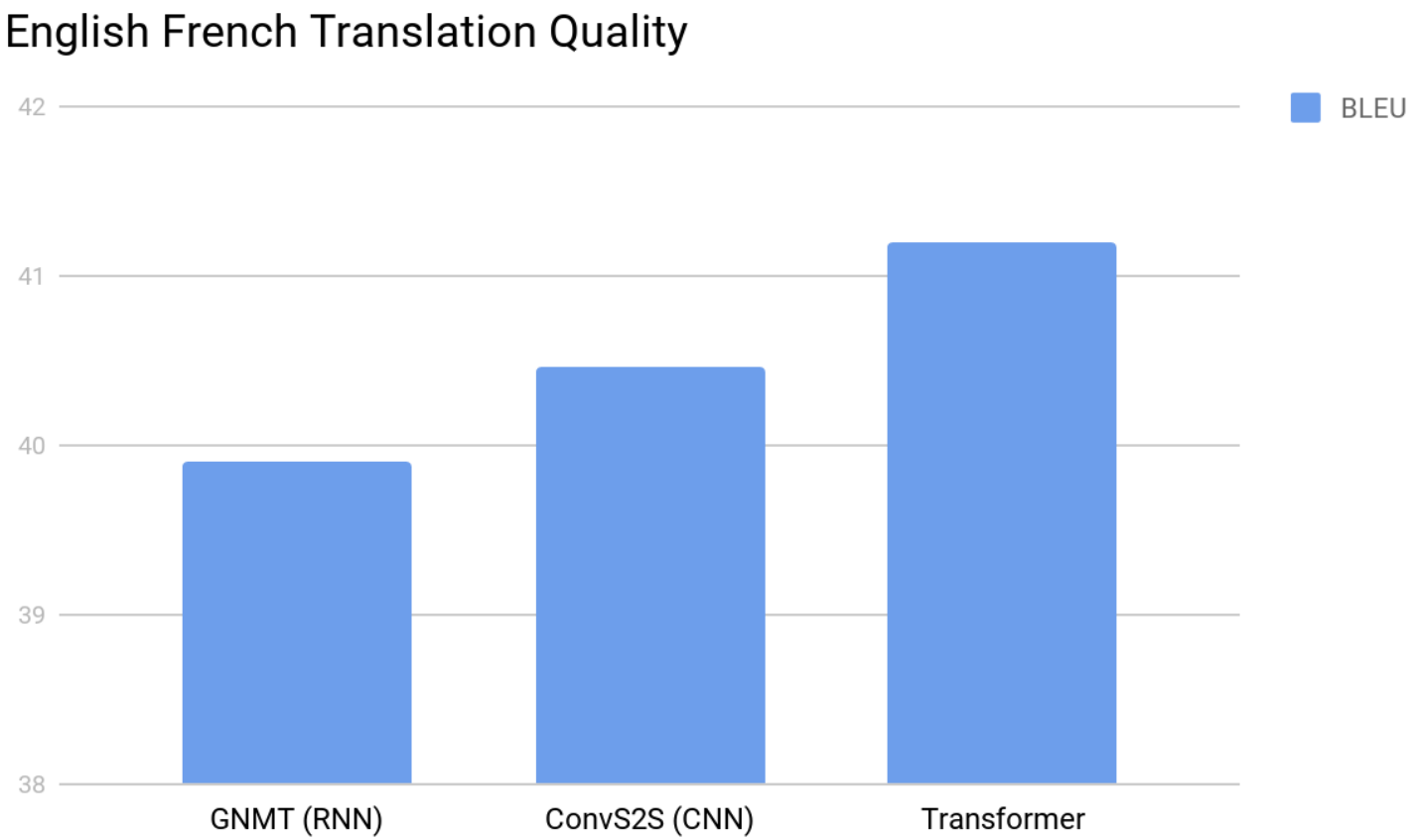
Attention is all you need

- Результаты:



Attention is all you need

- Результаты:



Ссылки

1. Прогресс в NLP.
2. Блог компании AYLIEN.
3. Блог Sebastian Ruder.
4. Статья про Transformer.
5. Книга «Глубокое обучение. Погружение в мир нейронных сетей».