Introduction to BERT and Transformer: pre-trained self-attention models to leverage unlabeled corpus data

PremiLab @ XJTLU, 4 April 2019

presented by Hang Dong

Presentation of the two papers:

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). **BERT: Pre-training of deep bidirectional transformers for language understanding.** (NAACL 2019)

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). **Attention is all you need.** (NIPS 2017)

Acknowledgement to all used slides, figures, tables, equations, texts from the papers, blogs and codes!

Acknowledgement to background image from http://ruder.io/nlp-imagenet/

Pre-training general language representations

- Feature-based approaches
  - Non-neural word representations
  - Neural embedding
    - Word embedding: Word2Vec, Glove, …
    - Sentence embedding, paragraph embedding, …
  - Deep contextualised word representation (ELMo, Embeddings from Language Models) (Peters *et al.*, 2018)
- Fine-tuning approaches
  - OpenAI GPT (Generative Pre-trained Transformer) (Radford *et al.*, 2018a)
  - **BERT** (Bi-directional Encoder Representations from Transformers) (Devlin *et al.*, 2018)

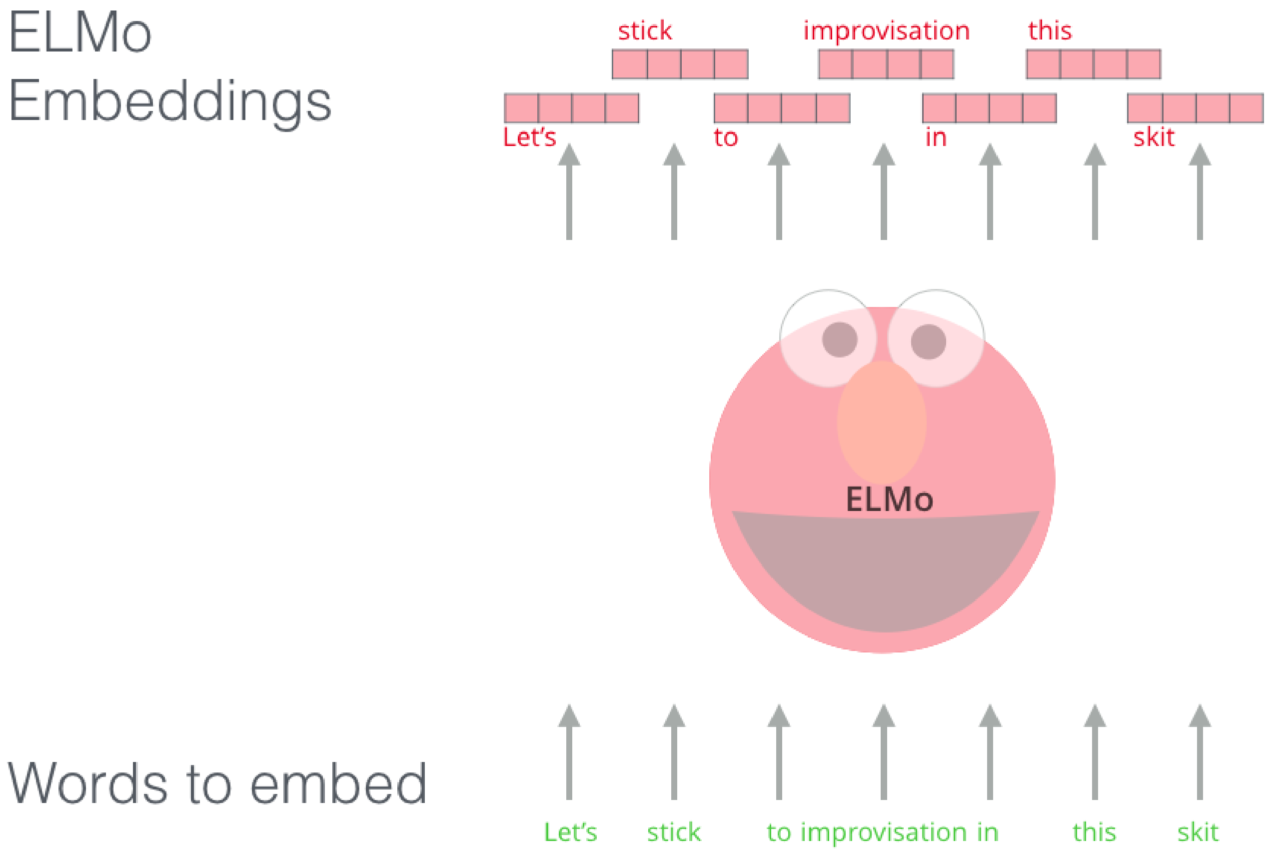
Content

- ELMo (Peters *et al.*, 2018)
- OpenAI GPT (Radford *et al.*, 2018a)
- Transformer (especially **self-attention**) (Vaswani *et al.*, 2017)
- **BERT** (Devlin *et al.*, 2018)
- Analyses & Future Studies

ELMo: deep contextualised word representation

(Peters *et al.*, 2018)

- "Instead of using a fixed embedding for each word, ELMo looks at the entire sentence before assigning each word in it an embedding."

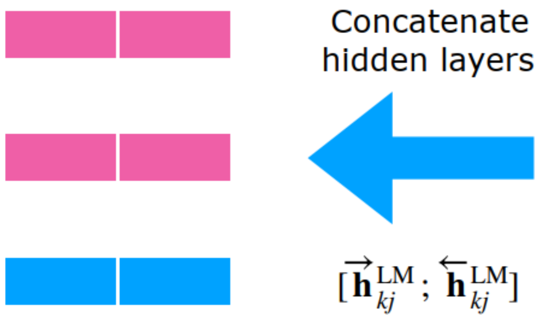


Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/

ELMo represents a word t_k as a linear combination of corresponding hidden layers (inc. its embedding)

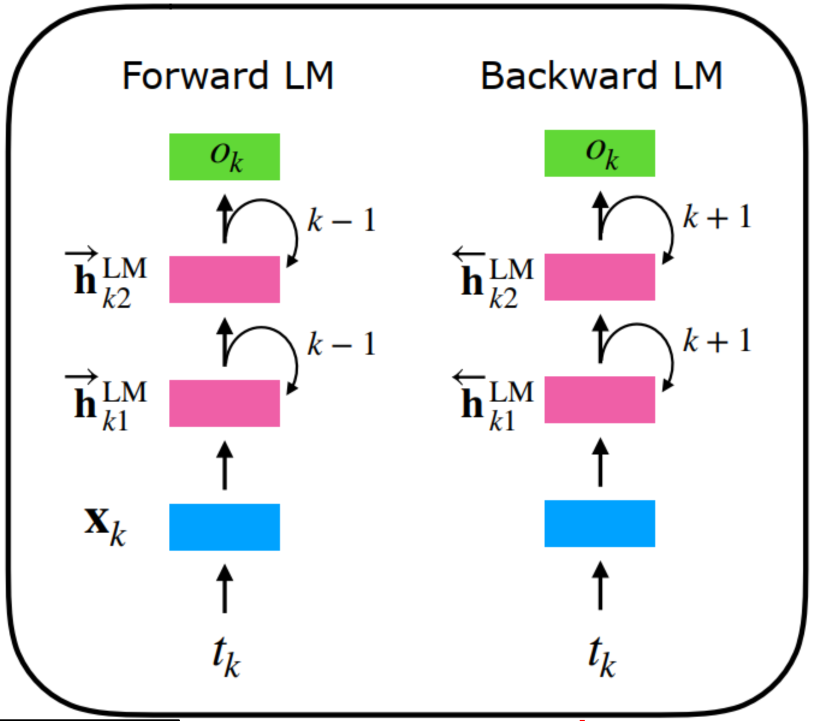
ELMo is a task specific representation. A down-stream task learns weighting parameters

$$\mathbf{ELMo}_k^{task} = \gamma^{task} \times \sum \left\{ \begin{array}{l} s_2^{task} \times \mathbf{h}_{k2}^{LM} \\ s_1^{task} \times \mathbf{h}_{k1}^{LM} \\ s_0^{task} \times \mathbf{h}_{k0}^{LM} \\ \quad ([\mathbf{x}_k\,;\,\mathbf{x}_k]) \end{array} \right.$$



Unlike usual word embeddings, ELMo is assigned to every *token* instead of a *type*

biLMs



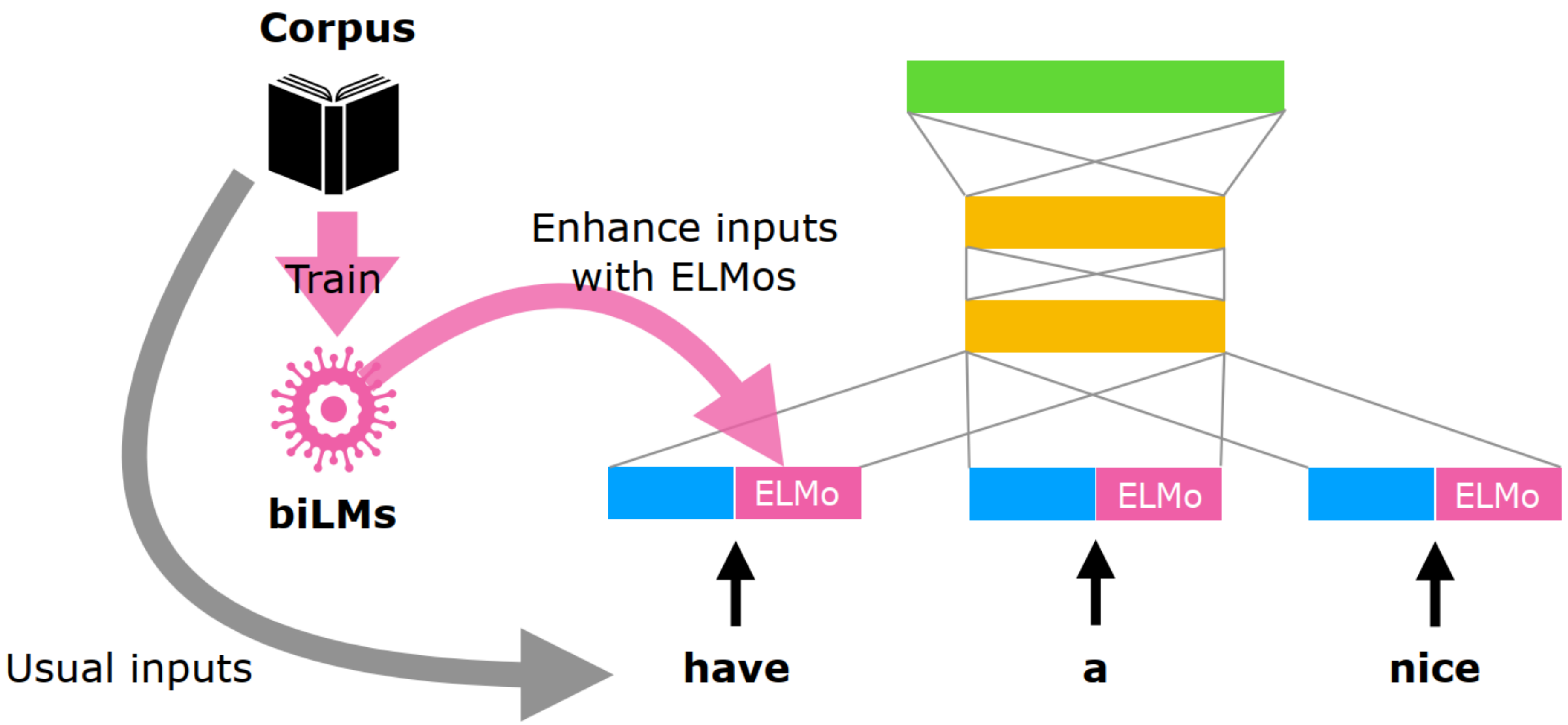
- biLMs consist of forward and backward LMs
  - Forward: $p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_1, t_2, \ldots, t_{k-1})$
  - Backward: $p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_{k+1}, t_{k+2}, \ldots, t_N)$

Acknowledgement to slides from https://www.slideshare.net/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018

ELMo

Acknowledgement to slides from https://www.slideshare.net/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018

ELMo can be integrated to almost all neural NLP tasks with simple concatenation to the embedding layer



Acknowledgement to slides from
https://www.slideshare.net/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018

ELMo

Many linguistic tasks are improved by using ELMo

	TASK	PREVIOUS SOTA		OUR BASELINE	ELMO + BASELINE	INCREASE (ABSOLUTE/ RELATIVE)
Q&A	SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
Textual entailment	SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
Semantic role labelling	SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coreference resolution	Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
Named entity recognition	NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
Sentiment analysis	SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

Table 1: Test set comparison of ELMo enhanced neural models with state-of-the-art single model baselines across six benchmark NLP tasks. The performance metric varies across tasks – accuracy for SNLI and SST-5; F_1 for SQuAD, SRL and NER; average F_1 for Coref. Due to the small test sizes for NER and SST-5, we report the mean and standard deviation across five runs with different random seeds. The "increase" column lists both the absolute and relative improvements over our baseline.

OpenAI GPT (Generative Pre-trained Transformer) – (1) pre-training

- Unsupervised pre-training, maximising the log-likelihood,

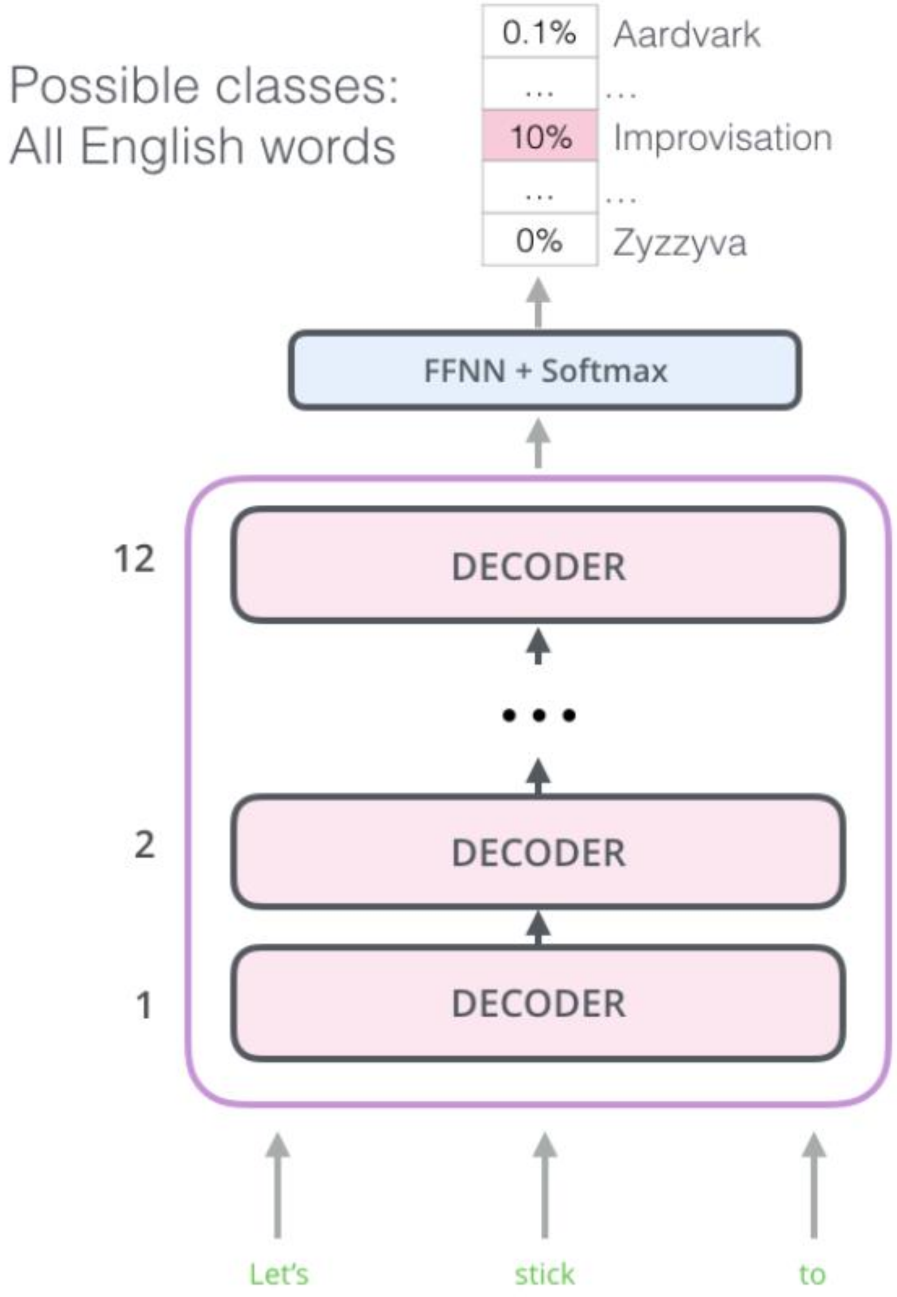
$$L_1(\mathcal{U}) = \sum_i \log P(u_i|u_{i-k}, \ldots, u_{i-1}; \Theta)$$

- where $\mathcal{U} = \{u_1, \ldots, u_n\}$ is an **unsupervised corpus of tokens**, k is the size of context window, P is modelled as a neural network with parameters Θ.

$$\begin{aligned} h_0 &= UW_e + W_p \\ h_l &= \texttt{transformer\_block}(h_{l-1}) \forall i \in [1, n] \\ P(u) &= \texttt{softmax}(h_n W_e^T) \end{aligned}$$

- where U is one-hot representation of tokens in the window, n is the total number of transformer layers, **`transformer_block`**() denotes the ***decoder of the Transformer model*** (multi-headed self-attention and position-wise feedfoward layers).

Equations in (Radford *et al.*, 2018)



Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/

GPT: (2) Fine-tuning

Given labelled data C, including each input as a sequence of tokens $x^1, x^2, \ldots, x^m$, each label as y .

$$P(y|x^1, \ldots, x^m) = \texttt{softmax}(h_l^m W_y)$$

$$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y|x^1, \ldots, x^m)$$

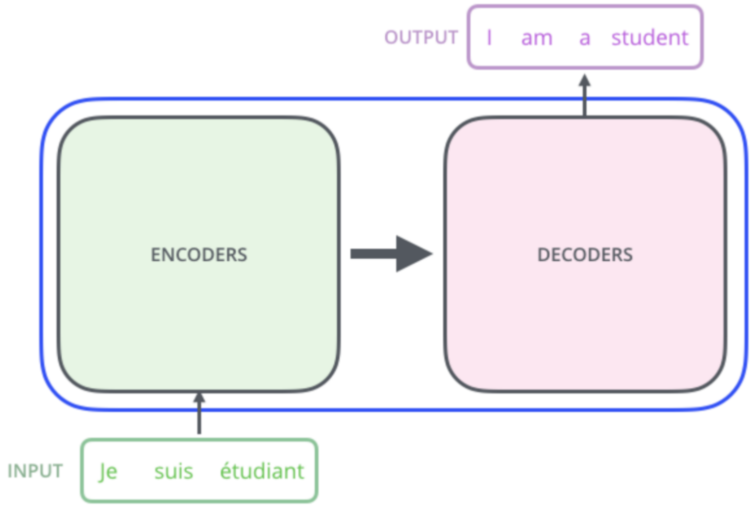
Then maximise the final objective function:

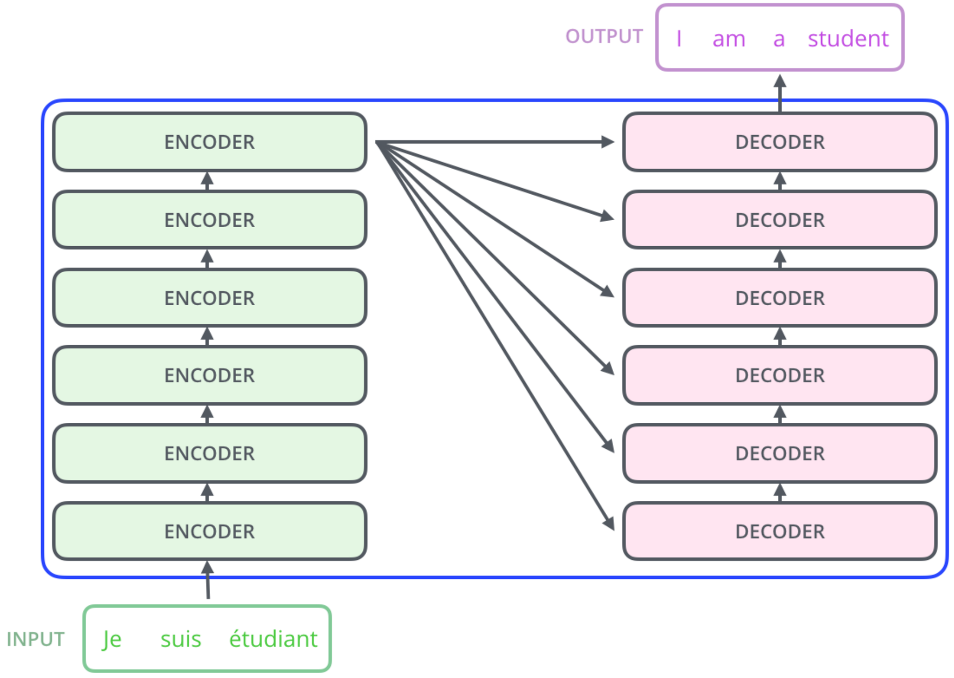
$$L_3(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C})$$

λ is set as 0.5 in the experiment.

Equations in (Radford *et al.*, 2018)

Transformer: a seq2seq model





$$N = 6$$
$$d_{\text{model}} = 512$$

Residual connection & Layer normalisation

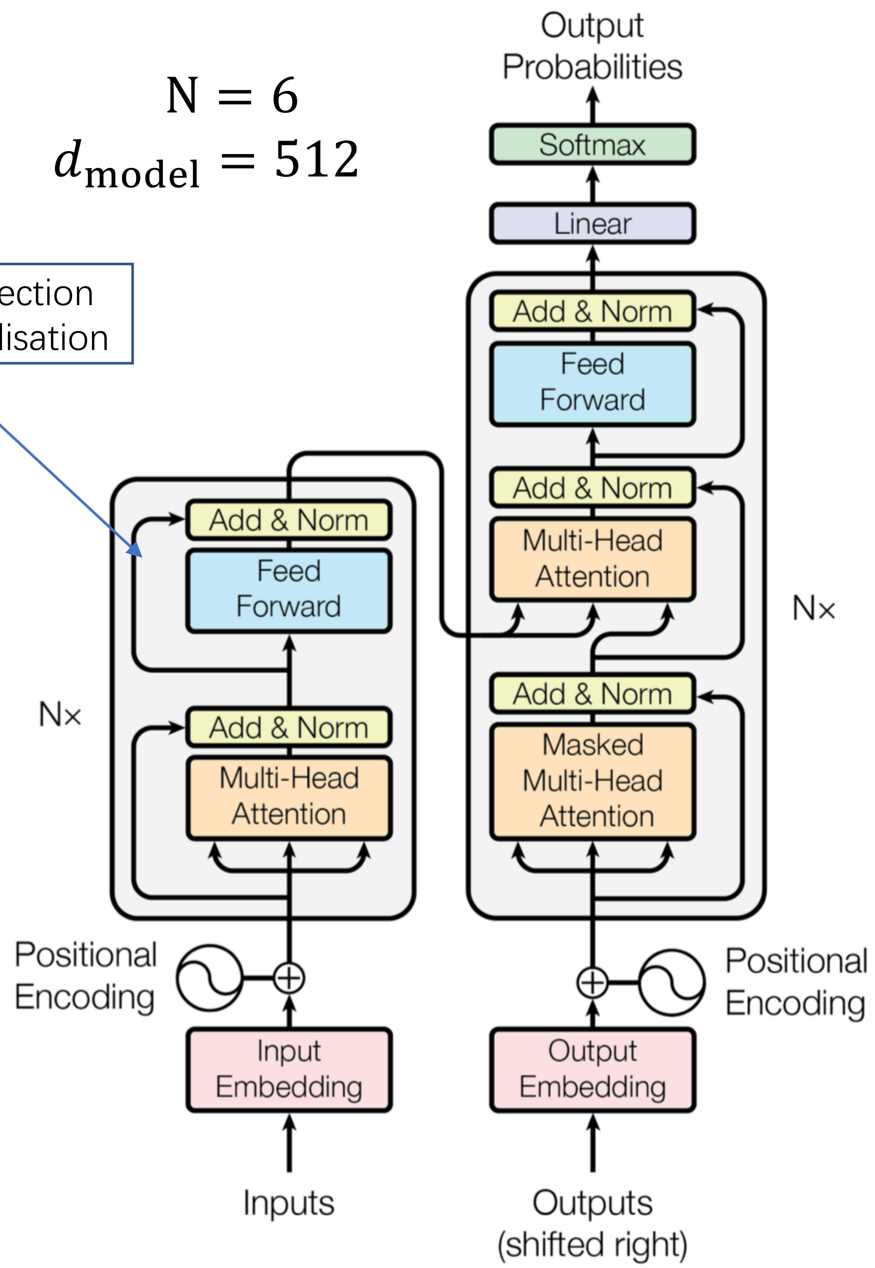
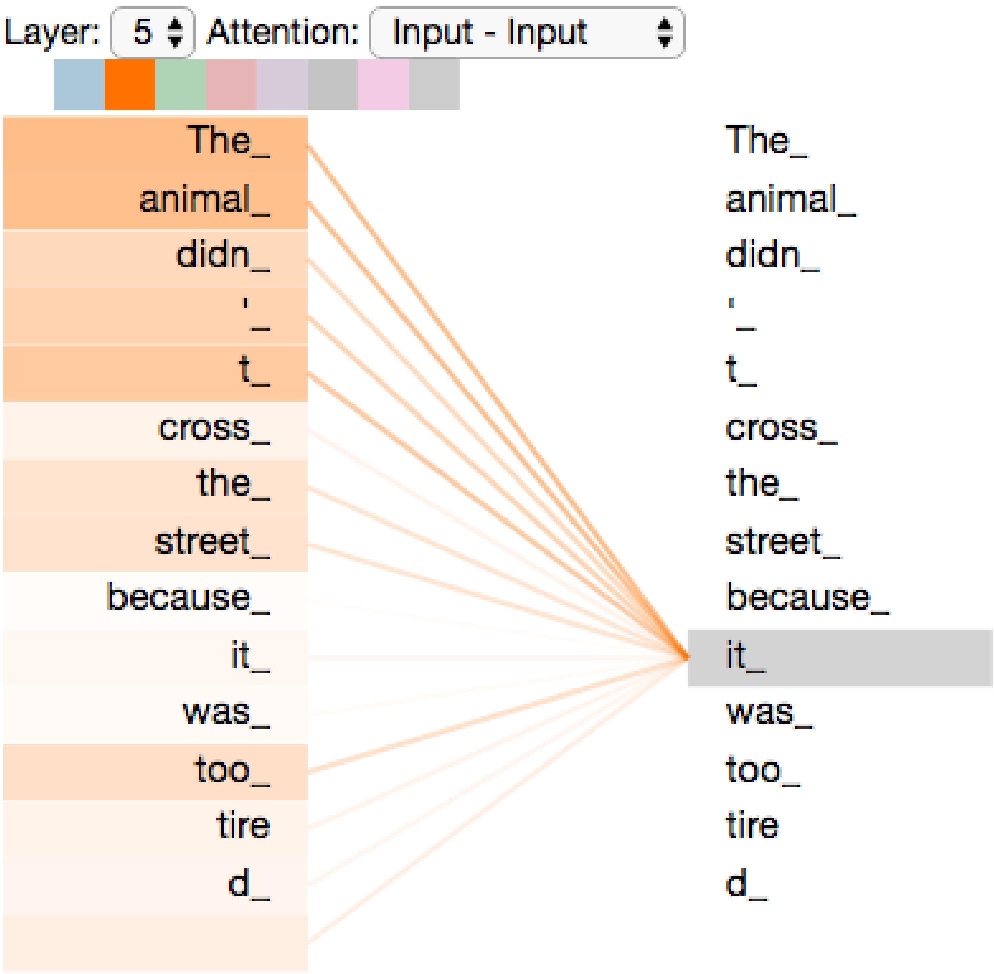


Figure 1: The Transformer - model architecture.

Figure in (Vaswani *et al.*, 2017)

Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/

Self-attention (1)

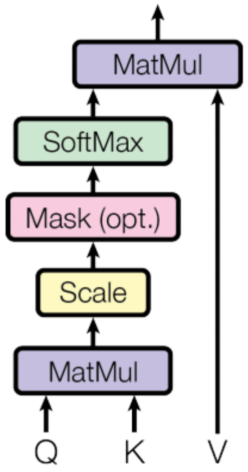


"The animal didn't cross the street because it was too tired"

"The animal didn't cross the street because it was too wide"

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

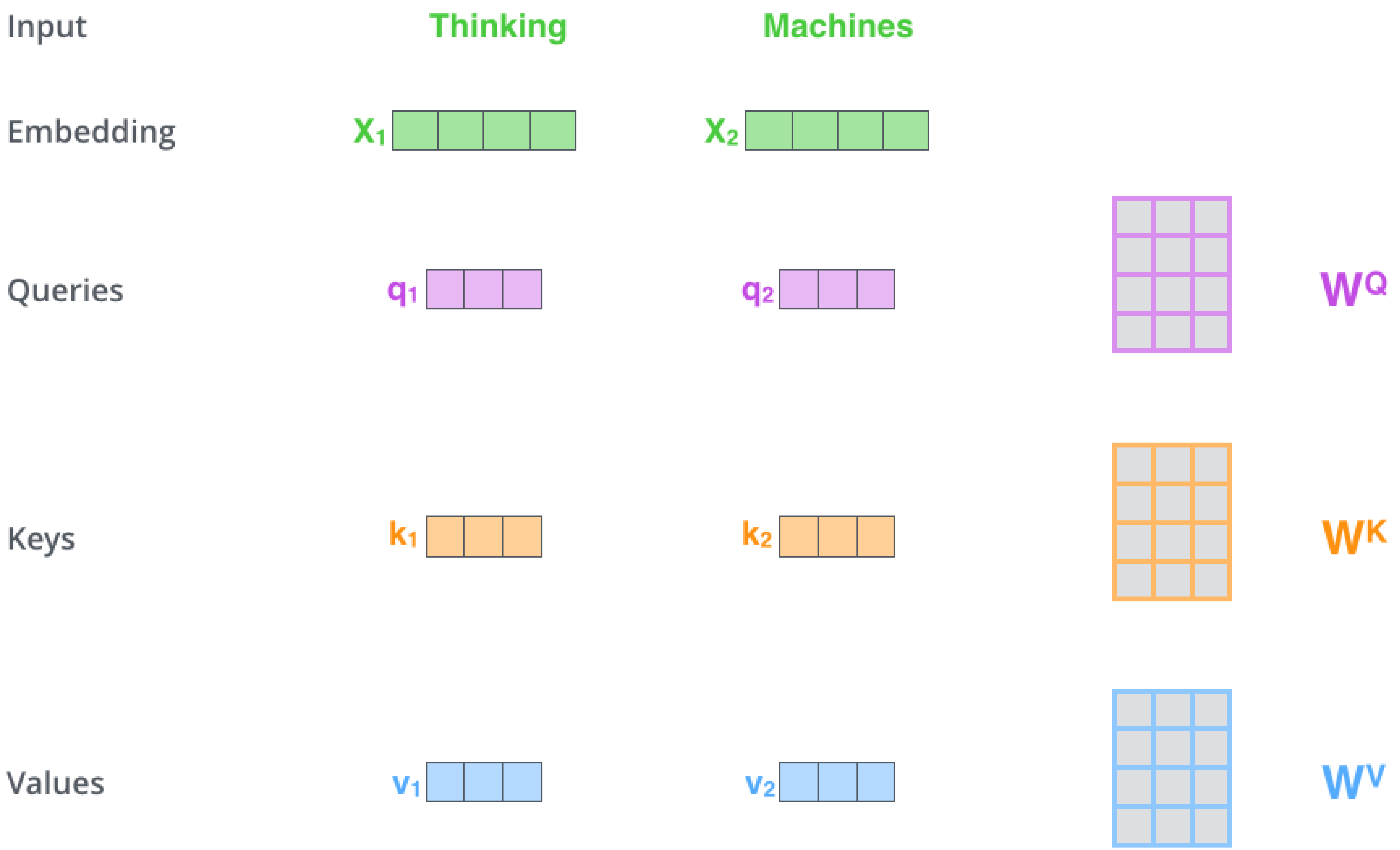
Scaled Dot-Product Attention



Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/

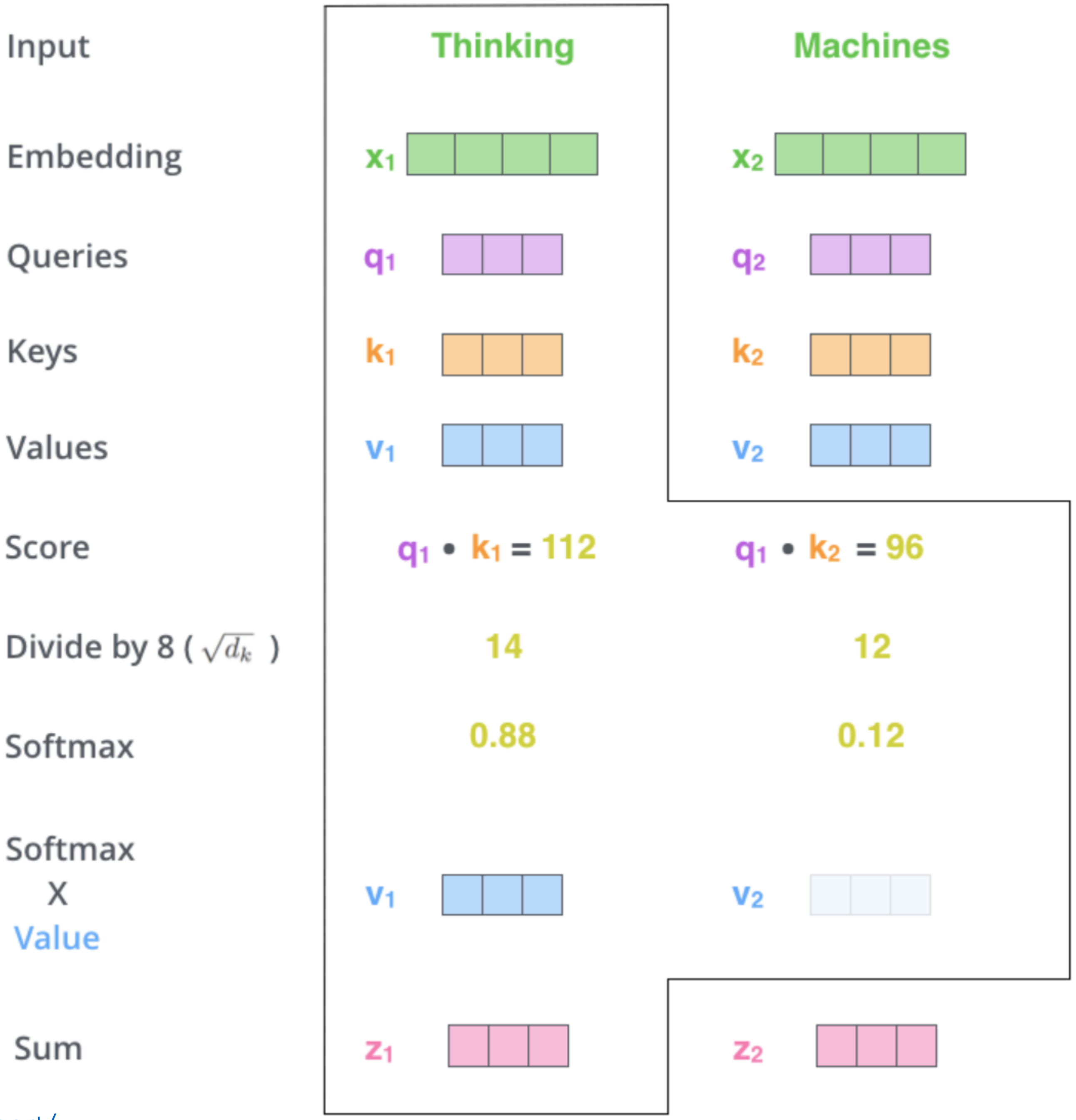
Equation and Figure in (Vaswani *et al.*, 2017)

Self-attention (2)

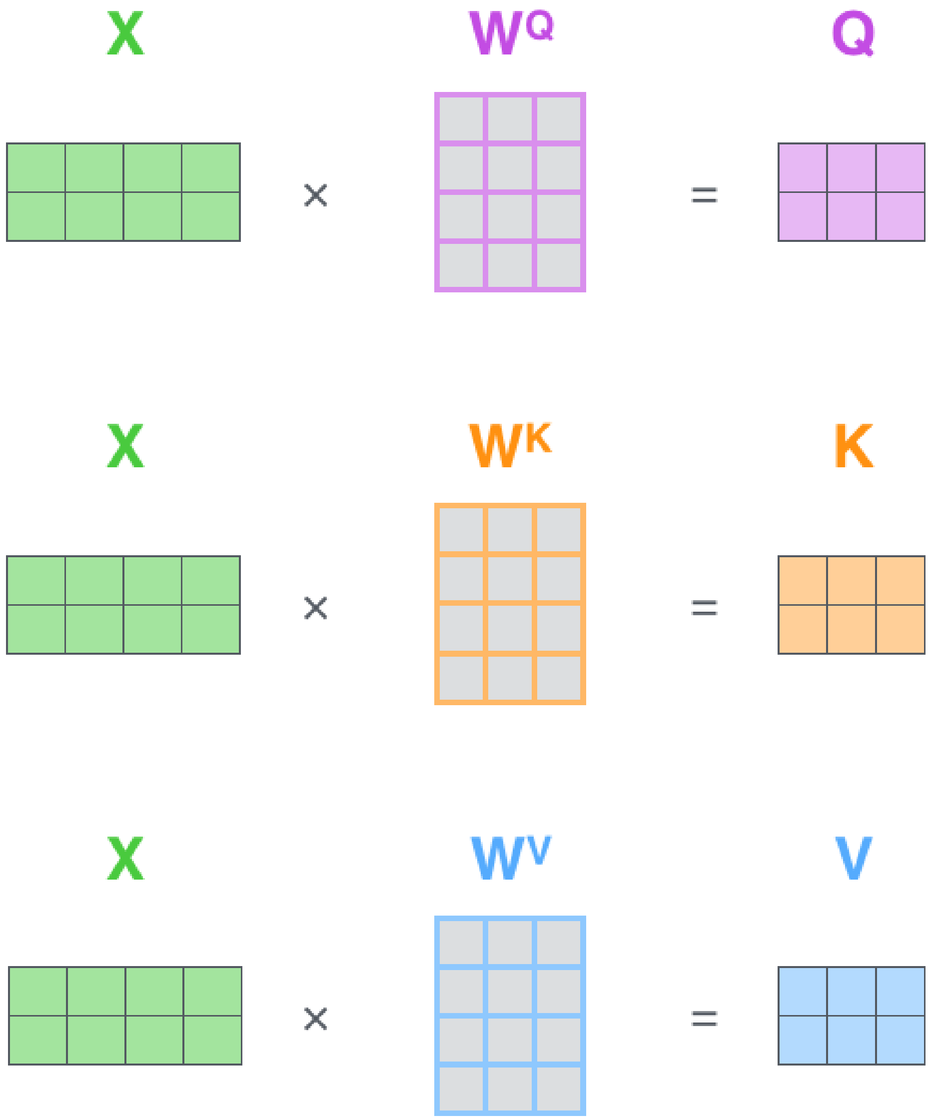


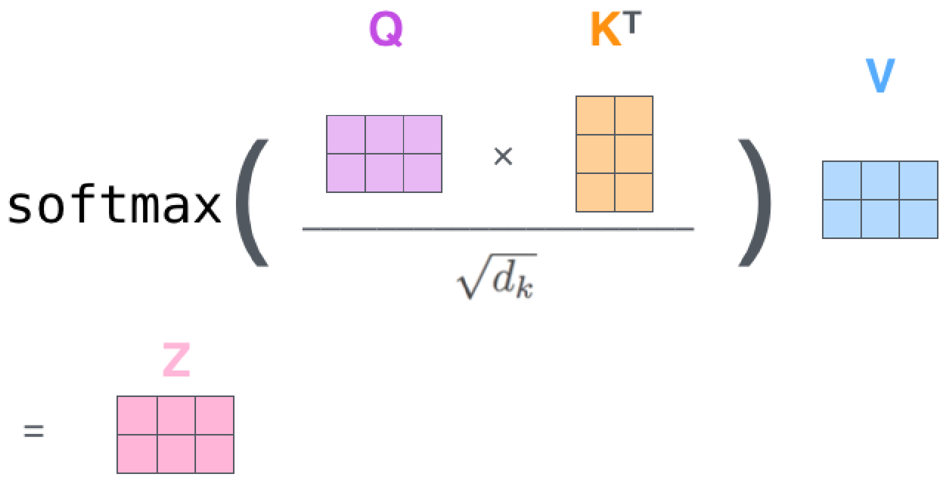
Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/

Self-attention (3)



Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/





Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/

Multi-head attention

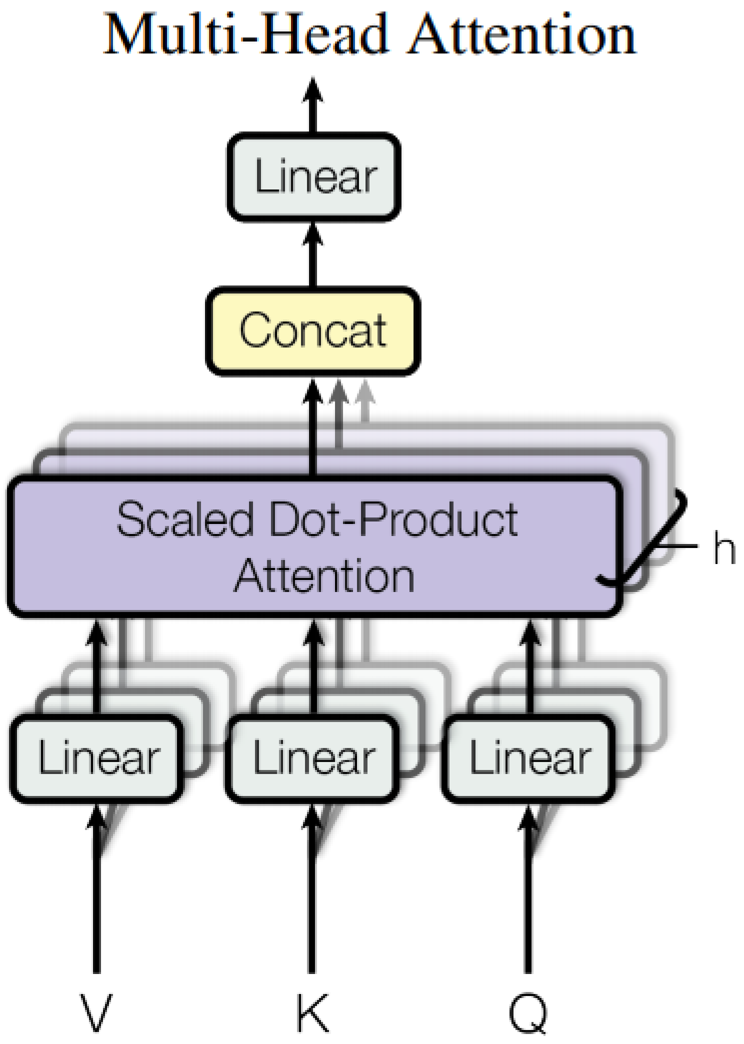
$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

$$\mathrm{MultiHead}(Q,K,V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

$$W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$$

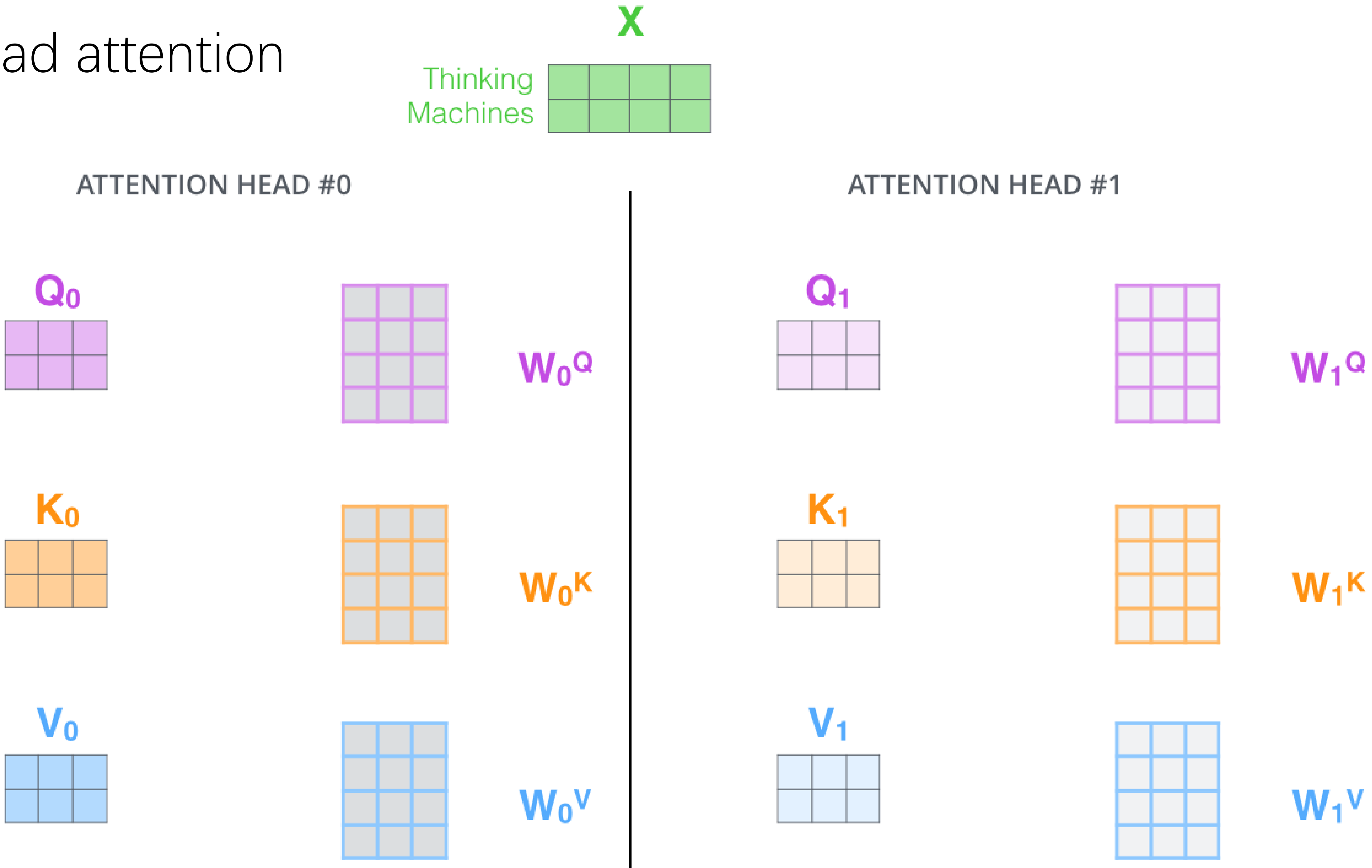
$$W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$$

$$h = 8, d_k = d_v = \frac{d_{model}}{h} = 64$$

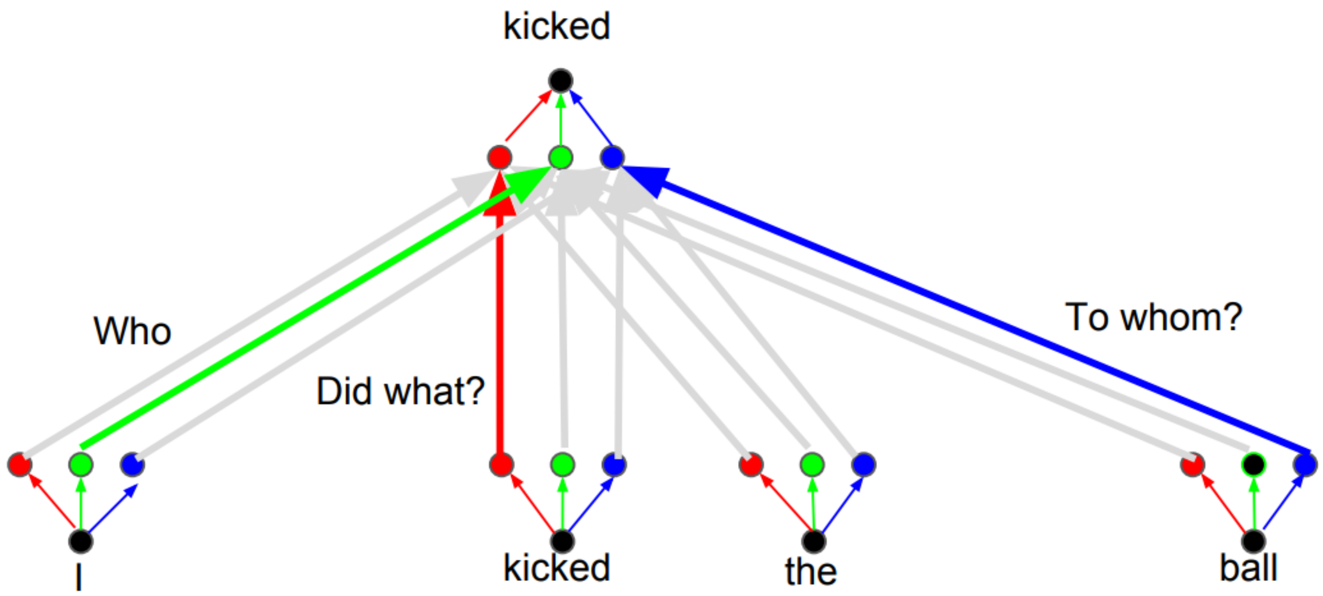


(Vaswani *et al.*, 2017)

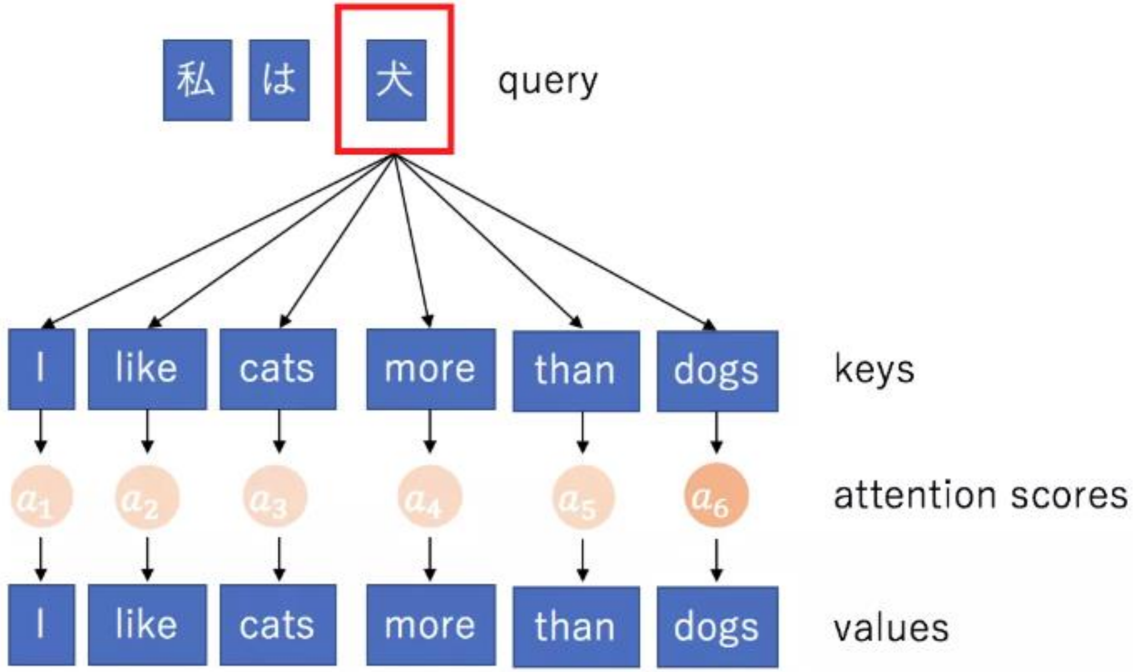
Multi-head attention



Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/



Acknowledgement to Figure from
http://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture14-transformers.pdf



Acknowledgement to Figure from (keitakurita, 2019)
http://mlexplained.com/2017/12/29/attention-is-all-you-need-explained/

Modelling the dependencies between

(keitakurita, 2019)

(1) the input and output tokens

(2) the input tokens themselves

(3) the output tokens themselves.

Three Multi-Head attention blocks

- Encoder Multi-Head Attention (left)
  - Keys, values and queries are the output of the previous layer in the encoder.
  - Multiple word-word alignments.
- Decoder Masked Multi-Head Attention (lower right)
  - Set the word-word attention weights for the connections to illegal "future" words to $-\infty$.
- Encoder-Decoder Multi-Head Attention (upper right)
  - Keys and values from the output of the encoder, queries from the previous decoder layer.

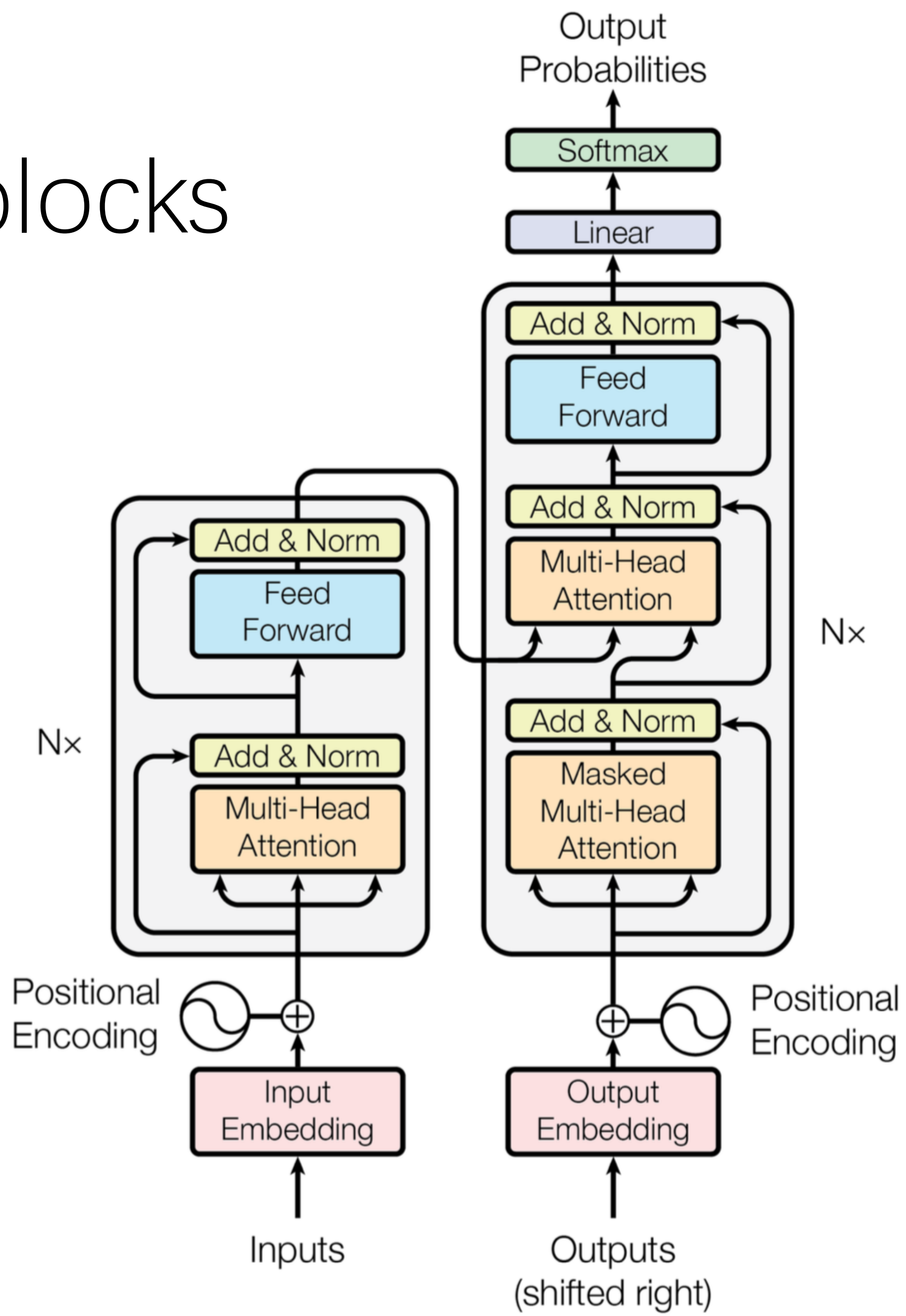
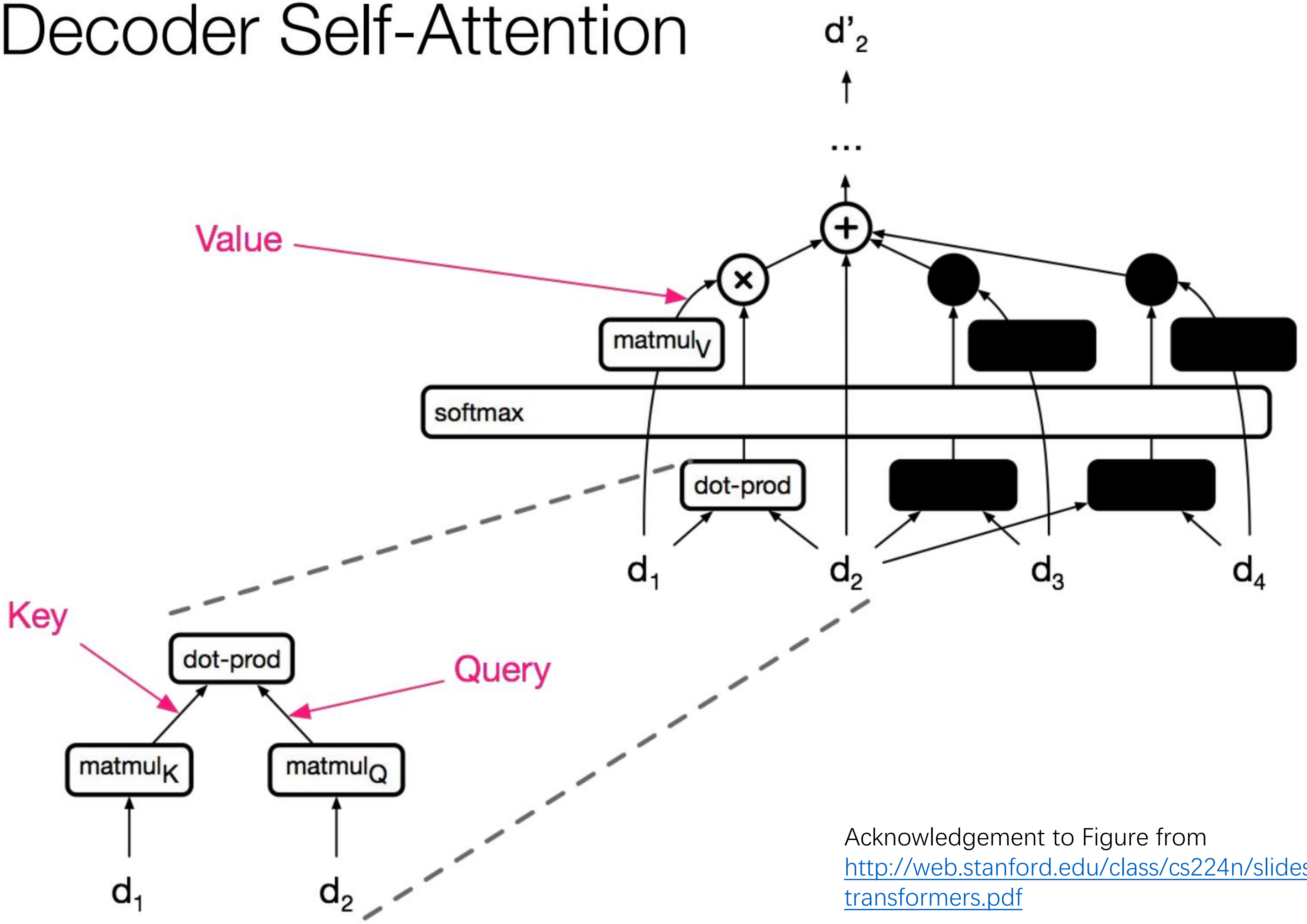


Figure 1: The Transformer - model architecture.

Figure in (Vaswani *et al.*, 2017)

Decoder Self-Attention



Acknowledgement to Figure from
http://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture14-transformers.pdf

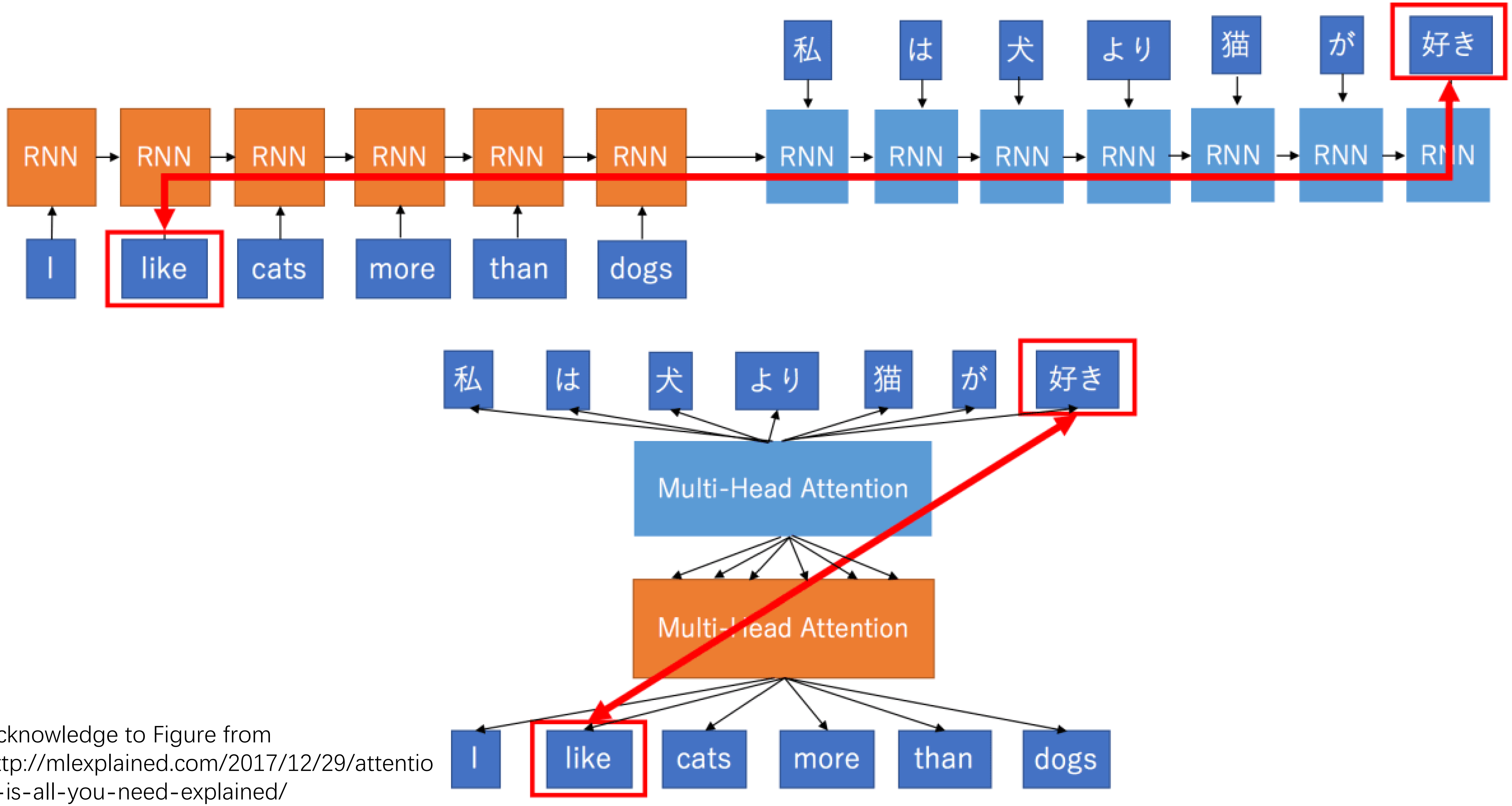
Why self-attention? - Efficiency and Path

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

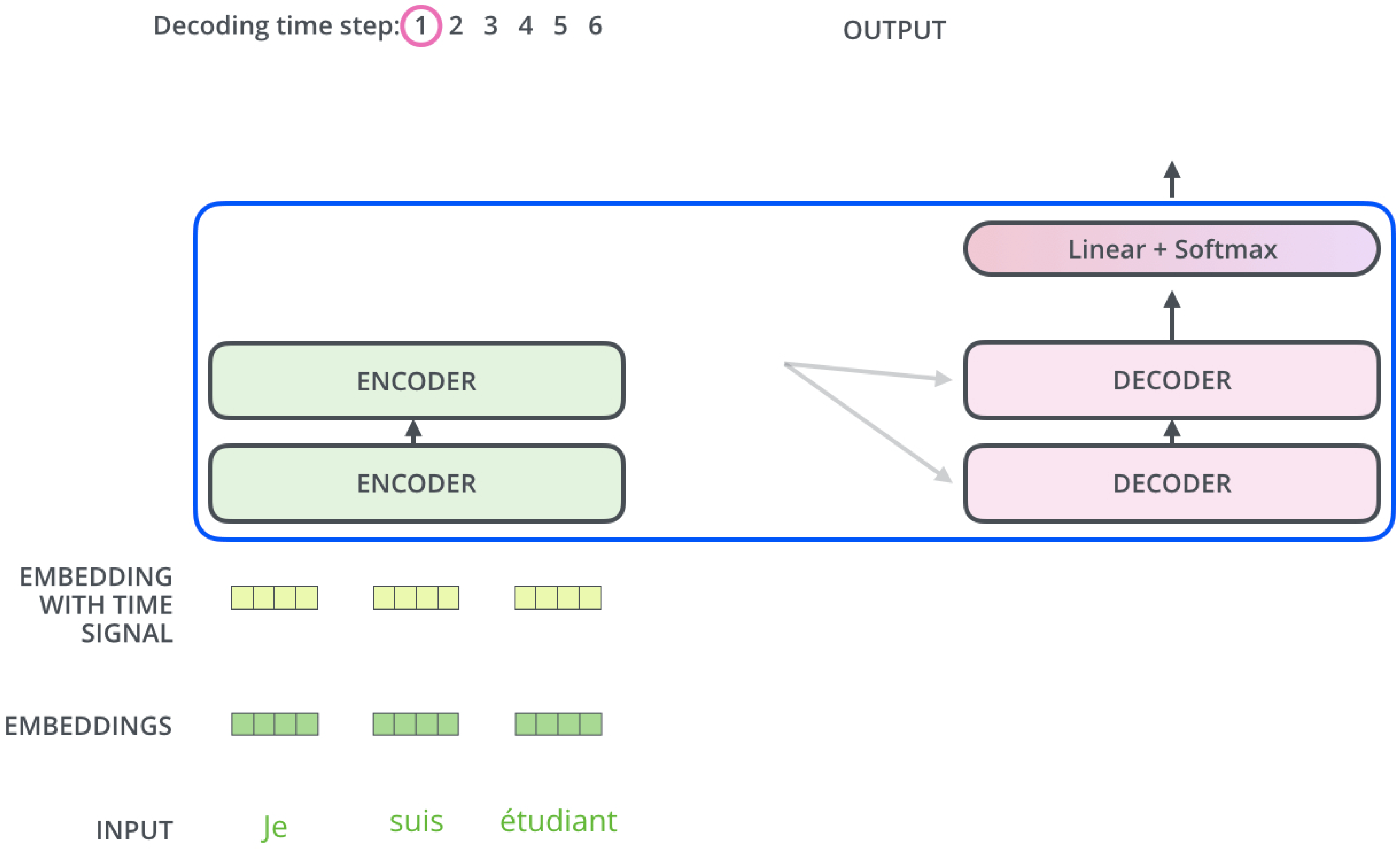
Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Table in (Vaswani *et al.*, 2017)

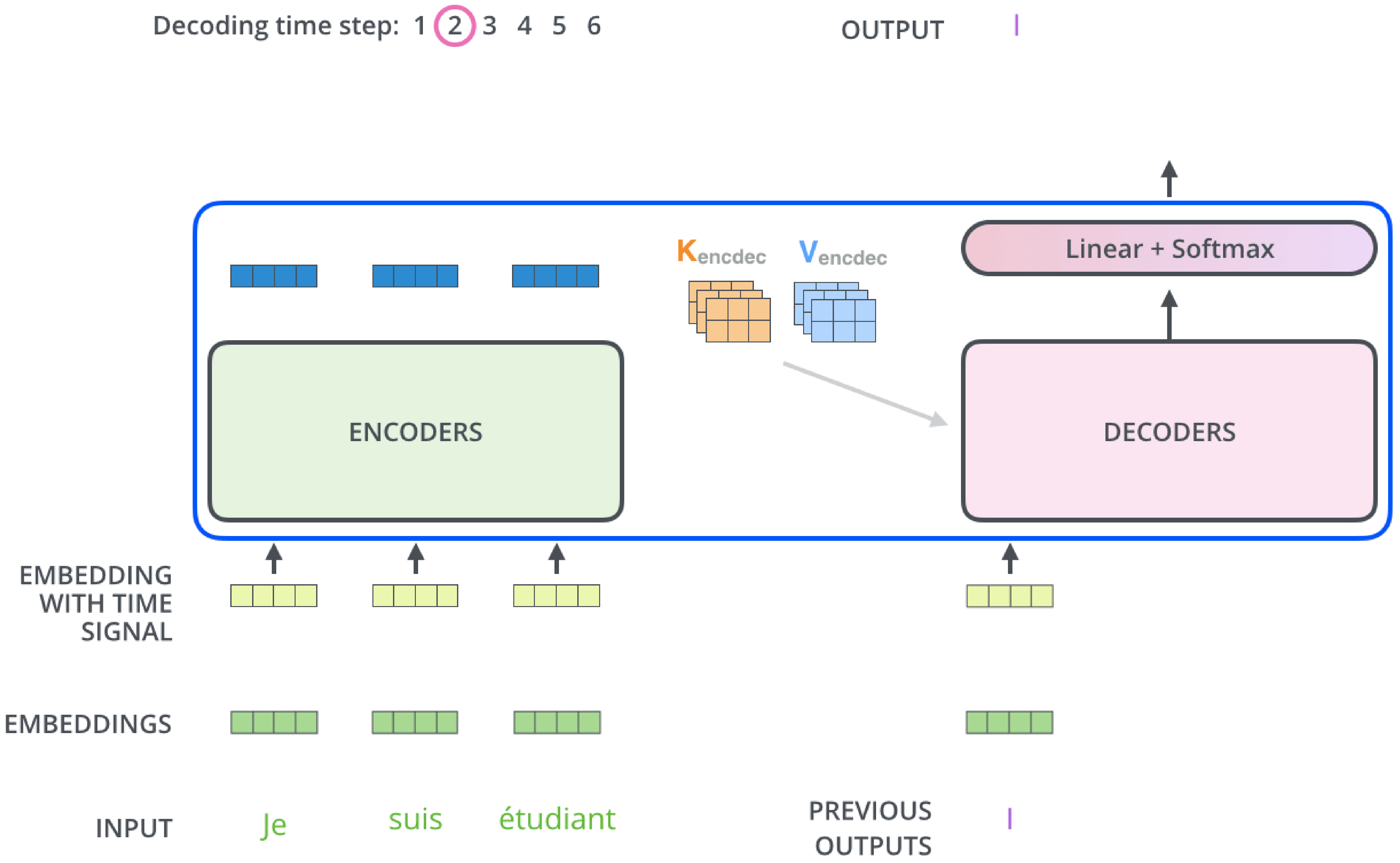
Maximum Path Length in RNN and Self-attention



Acknowledge to Figure from http://mlexplained.com/2017/12/29/attention-is-all-you-need-explained/



Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/



Acknowledgement to Figure from http://jalammar.github.io/illustrated-bert/

Positional Embedding

- In order to add position information (order of the sequence)

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

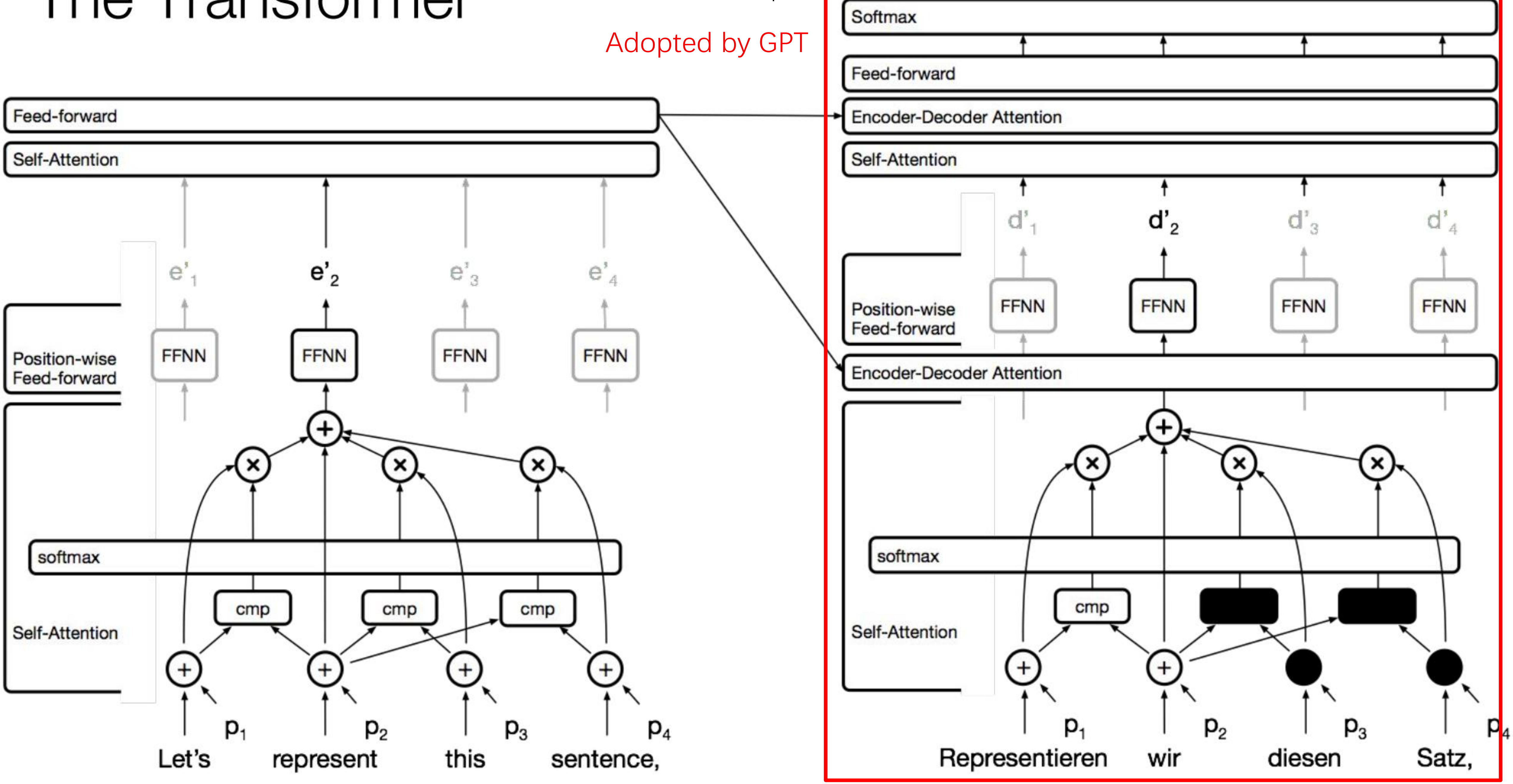
- Each dimension of the positional encoding corresponds to a sinusoid.

- For any fixed offset k, PE_{pos+k} can be represented as a linear transformation of PE_{pos}. This would allow the model to easily learn to attend by relative positions.

Equations in (Vaswani *et al.*, 2017)

The Transformer

Acknowledgement to the slide adapted from http://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture14-transformers.pdf



Evaluation for Transformer

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Table in (Vaswani *et al.*, 2017)

Table in (Vaswani *et al.*, 2017)

Evaluation for Transformer – parameter tuning

Table 3: Variations on the Transformer architecture. Unlisted values are identical to those of the base model. All metrics are on the English-to-German translation development set, newstest2013. Listed perplexities are per-wordpiece, according to our byte-pair encoding, and should not be compared to per-word perplexities.

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

What is BERT (Bidirectional Encoder Representations from Transformers)?

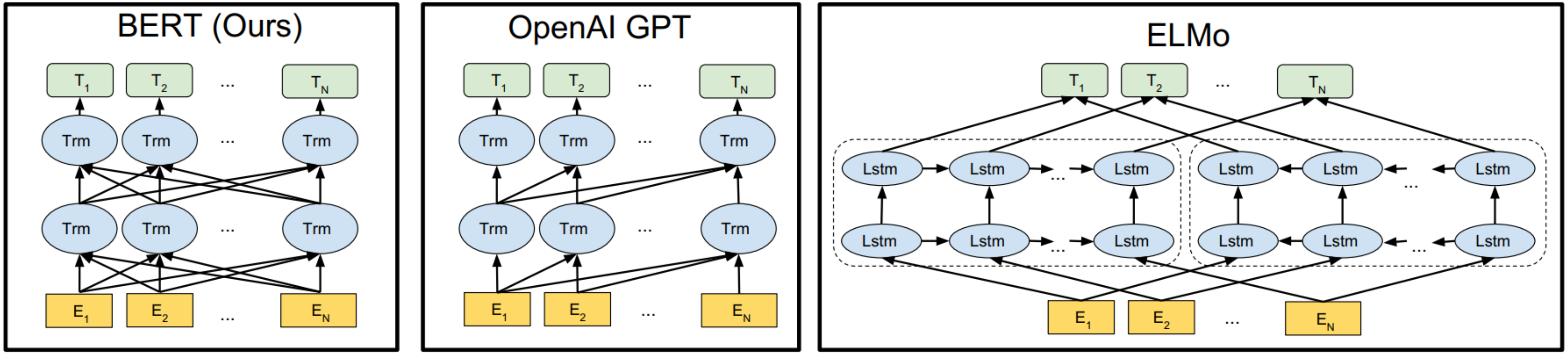
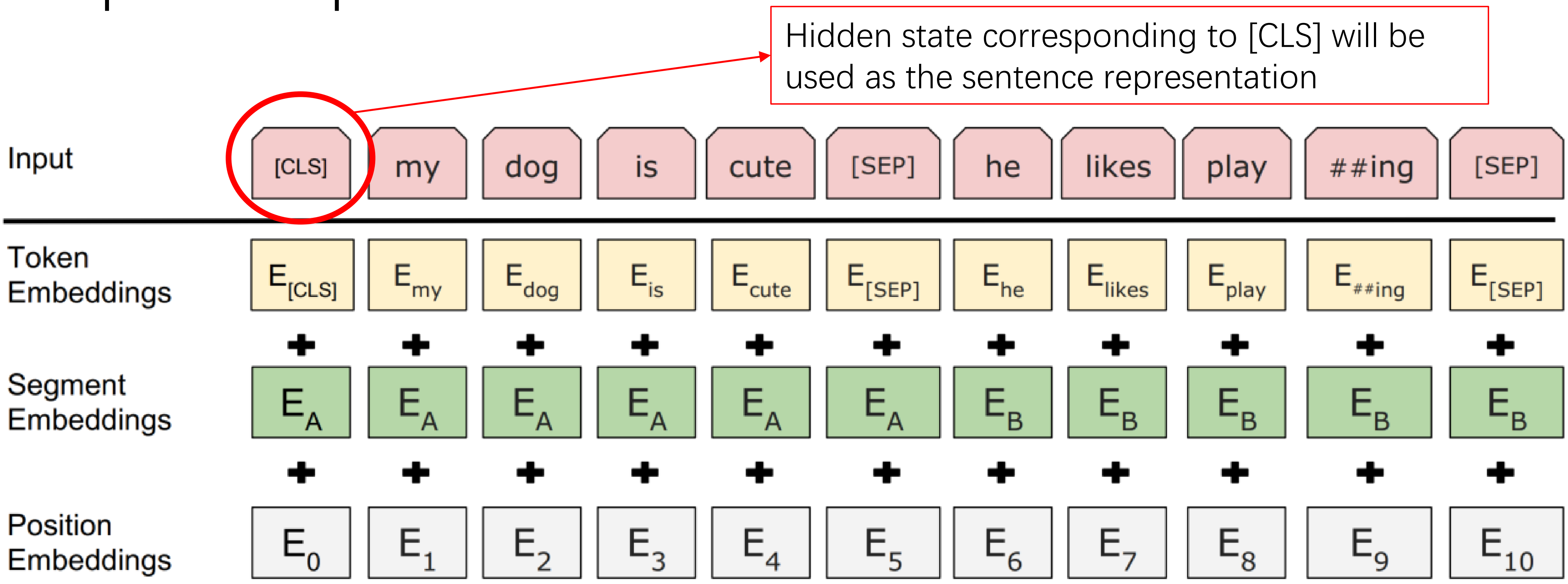


Figure 1: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTM to generate features for downstream tasks. Among three, only BERT representations are jointly conditioned on both left and right context in all layers.

Figure in (Devlin *et al.*, 2018)

Input Representation



- Token Embeddings: WordPiece embedding (Wu *et al.*, 2016)
- Segment Embeddings: randomly initialized and learned; single sentence input only adds E_A
- Position embeddings: randomly initialized and learned

Figure in (Devlin *et al.*, 2018)

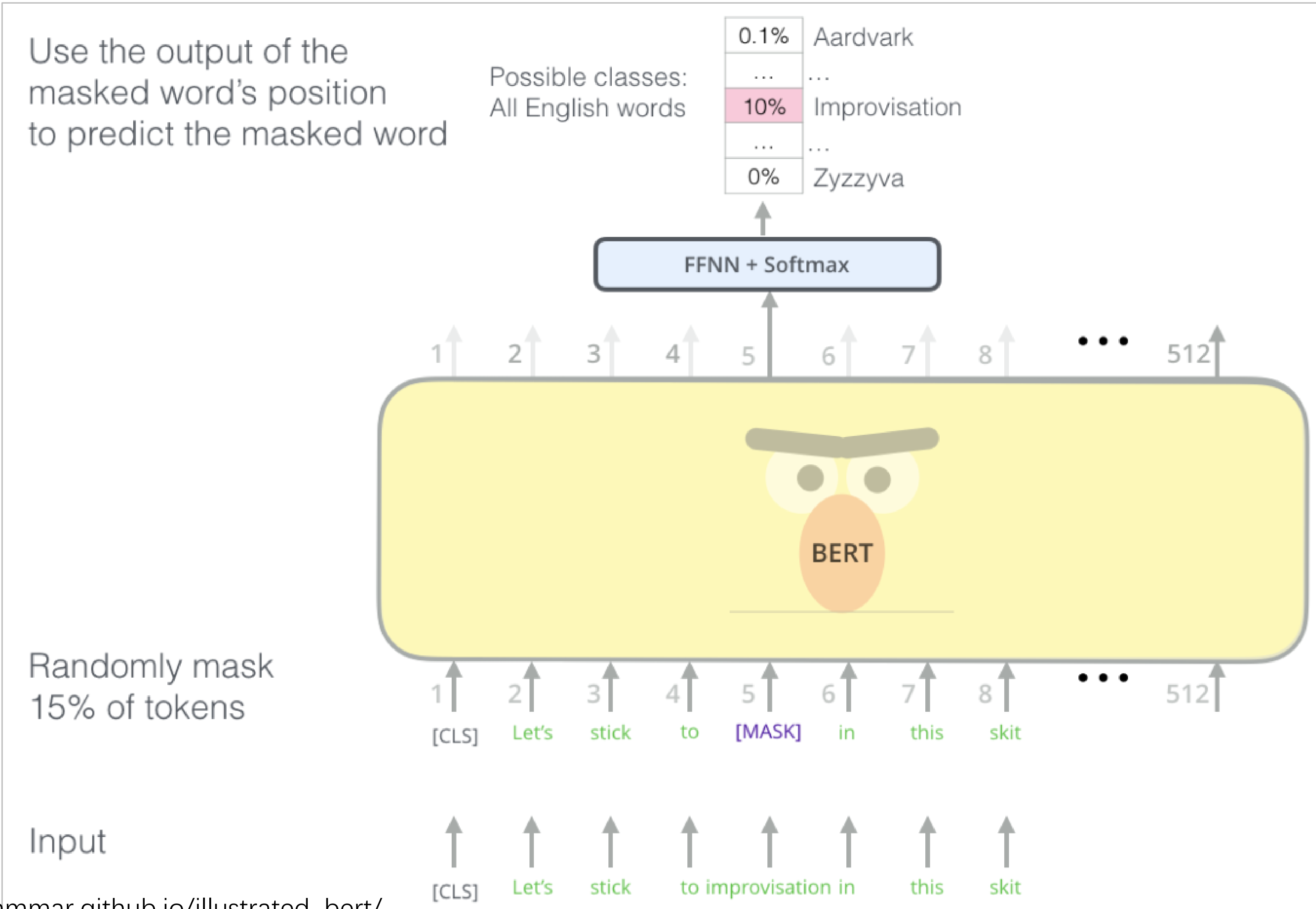
Training tasks (1) - Masked Language Model

- Masked Language Model: Cloze Task

- Masking(input_seq):

  For every input_seq :
  - Randomly select 15% of tokens (not more than 20 per seq)
    - For 80% of the time:
      - Replace the word with the [MASK] token.
    - For 10% of the time:
      - Replace the word with a random word
    - For 10% of the time
      - Keep the word unchanged..

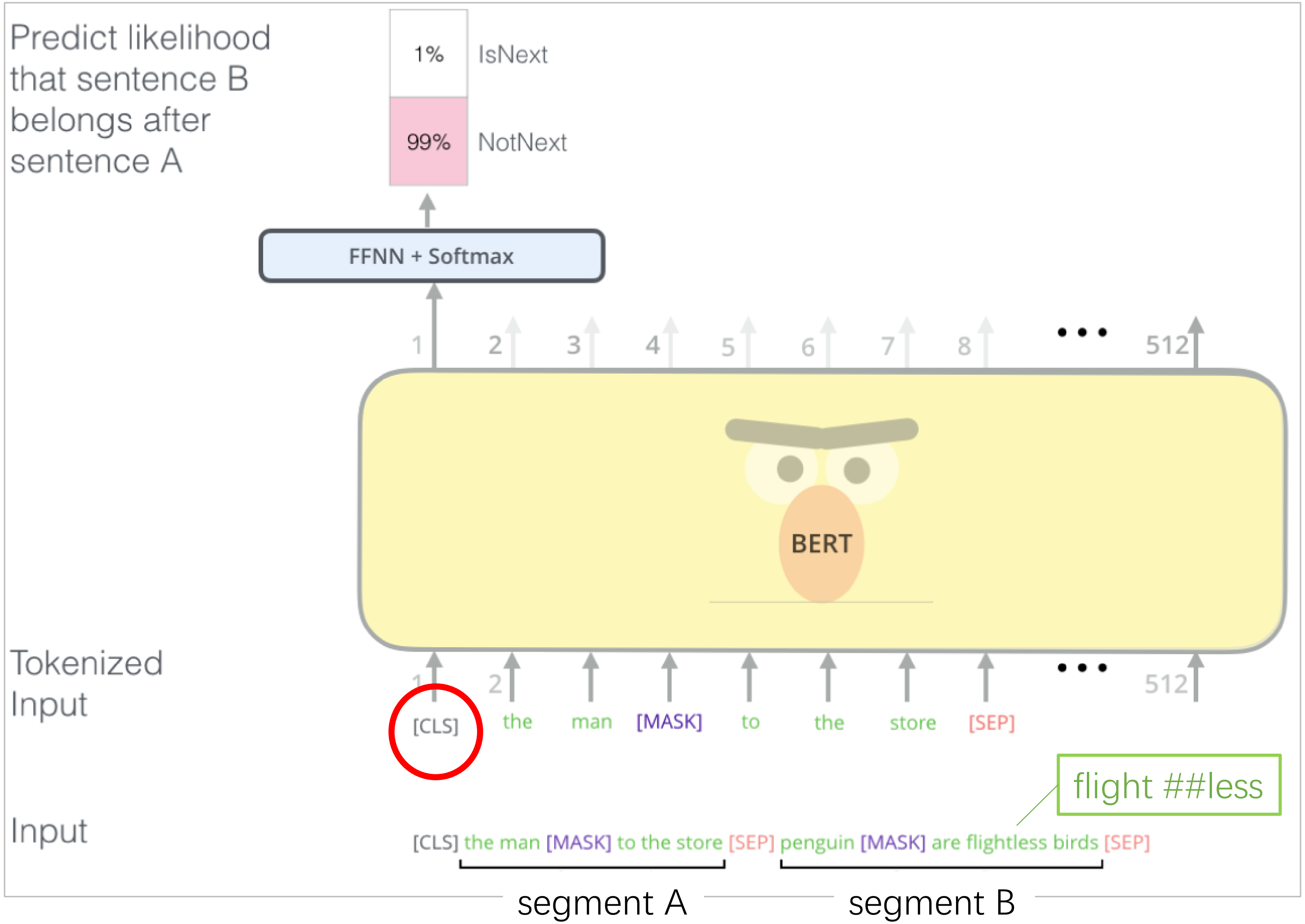
- For related code see *def create_masked_lm_predictions(…)* in https://github.com/google-research/bert/blob/master/create_pretraining_data.py



Acknowledgement to the Figure from http://jalammar.github.io/illustrated-bert/

Training tasks (2) – Next Sentence Prediction

- Next sentence prediction – Binary classification
- For every input document as a sentence-token 2D list:
  - Randomly select a split over sentences:
    - Store the segment A
    - For 50% of the time:
      - Sample random sentence split from *another* document as segment B.
    - For 50% of the time:
      - Use the actual sentences as segment B.
  - Masking (Truncate([segment A, segment B]))
- For related code see *def create_instances_from_document (…)* in https://github.com/google-research/bert/blob/master/create_pretraining_data.py



Acknowledgement to the Figure adapted from http://jalammar.github.io/illustrated-bert/

Pre-Training datasets and details

- Training loss L is the sum of the **mean masked LM likelihood and mean next sentence prediction likelihood**.

- Dataset: Long contiguous word sequences.
  - BooksCorpus (800M words), about 7,000 unique unpublished books from a variety of genres including Adventure,
    Fantasy, and Romance.
  - English Wikipedia (2,500M words), excluding lists, tables, headers.

- Sequence length 512; Batch size 256; trained for 1M steps (approximately 40 epochs); learning rate 1e-4; Adam optimiser, β_1 as 0.9, β_2 as 0.999; dropout as 0.1 on all layers; GELU activation; L2 weight decay of 0.01; learning rate warmup over the first 10,000 steps, linear decay of learning rate ⋯

- $\mathrm{BERT_{BASE}}$: $\mathrm{N} = 6$, $d_{\mathbf{model}} = 512$, $h = 12$, Total Parameters=110M
- 4 cloud TPUs in Pod configuration (16 TPU chips total)

- $\mathrm{BERT_{LARGE}}$: $\mathrm{N} = 24$, $d_{\mathbf{model}} = 1024$, $h = 16$, Total Parameters=340M
- 16 Cloud TPUs (64 TPU chips total)

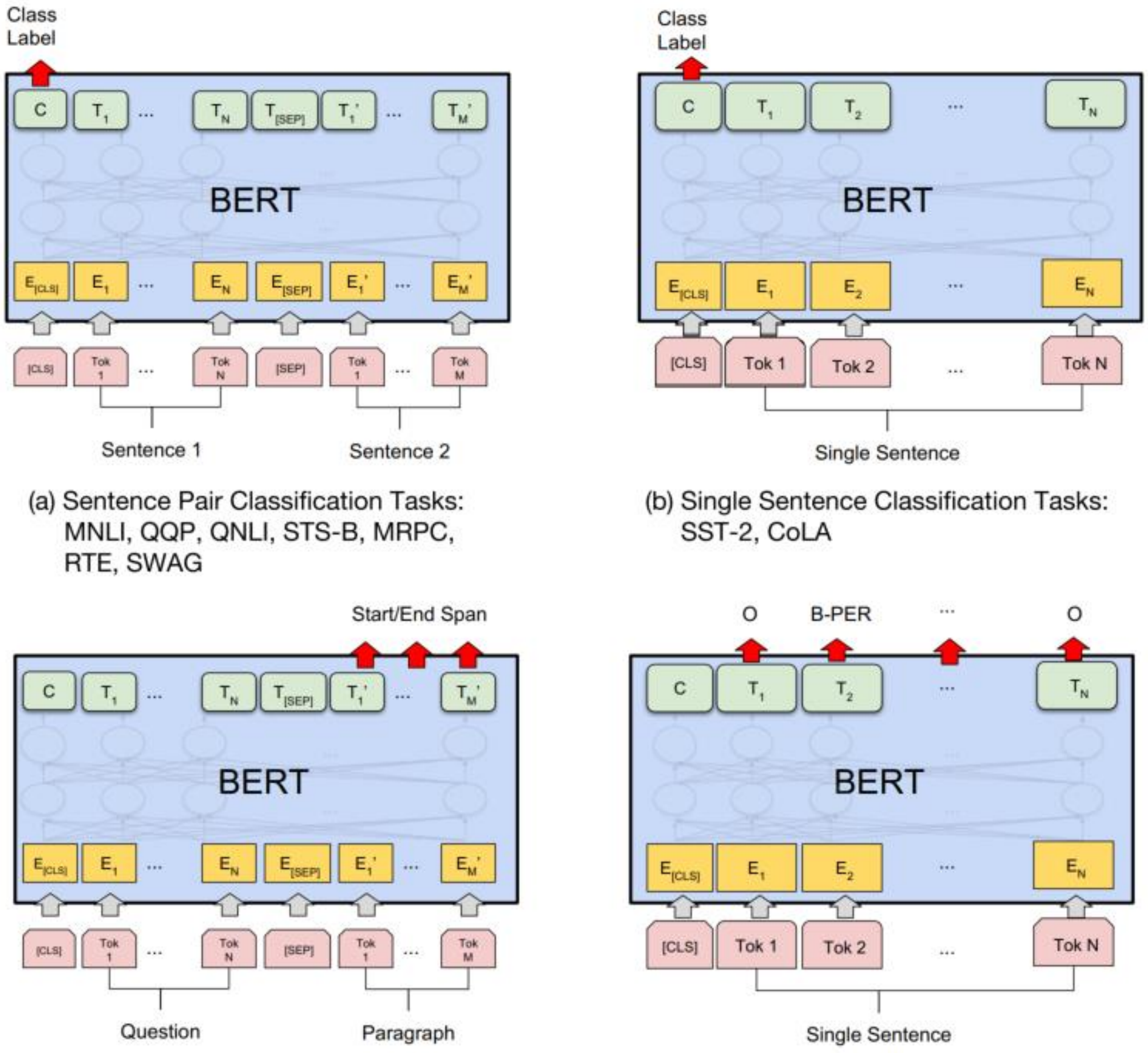
- Each pretraining took 4 days to complete.

Fine-tuning with BERT

- Context vector C: Take the final hidden state corresponding to the first token in the input: [CLS].
- Transform to a probability distribution of the class labels:

$$P = \mathrm{softmax}(CW^T)$$

- **Batch size**: 16, 32
- **Learning rate (Adam)**: 5e-5, 3e-5, 2e-5
- **Number of epochs**: 3, 4



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

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

(c) Question Answering Tasks: SQuAD v1.1

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

Figure in (Devlin *et al.*, 2018)

Evaluation for BERT: GLUE

- General Language Understanding Evaluation (**GLUE**) benchmark: Standard split of data to train, validation, test, where labels for the test set is only held in the server.

- Sentence pair tasks
  - **MNLI**, Multi-Genre Natural Language Inference
  - **QQP**, Quora Question Pairs
  - **QNLI,** Question Natural Language Inference
  - **STS-B** The Semantic Textual Similarity Benchmark
  - **MRPC** Microsoft Research Paraphrase Corpus
  - **RTE** Recognizing Textual Entailment
  - **WNLI** Winograd NLI is a small natural language inference dataset
- Single sentence classification
  - **SST-2** The Stanford Sentiment Treebank
  - **CoLA** The Corpus of Linguistic Acceptability

Evaluation for BERT: GLUE

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
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
$\mathrm{BERT}_{\mathrm{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\mathrm{BERT}_{\mathrm{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

Table 1: GLUE Test results, scored by the GLUE evaluation server. The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set. OpenAI GPT = (L=12, H=768, A=12); $\mathrm{BERT}_{\mathrm{BASE}}$ = (L=12, H=768, A=12); $\mathrm{BERT}_{\mathrm{LARGE}}$ = (L=24, H=1024, A=16). BERT and OpenAI GPT are single-model, single task. All results obtained from https://gluebenchmark.com/leaderboard and https://blog.openai.com/language-unsupervised/.

Table in (Devlin *et al.*, 2018)

Evaluation on SQUAD

- The Standford Question Answering Dataset (SQuAD) is a collection of 100k crowdsourced question/answer pairs.

- **Input Question:**

```
Where do water droplets collide with ice crystals to form precipitation?
```

- **Input Paragraph:**

```
... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. ...
```

- **Output Answer:**

```
within a cloud
```

System	Dev EM	Dev F1	Test EM	Test F1
Leaderboard (Oct 8th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
#1 Single - nlnet	-	-	83.5	90.1
#2 Single - QANet	-	-	82.5	89.3
Published				
BiDAF+ELMo (Single)	-	85.8	-	-
R.M. Reader (Single)	78.9	86.3	79.5	86.6
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

Table 2: SQuAD results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.

Table in (Devlin *et al.*, 2018)

Evaluation on Named Entity Recognition

- The CoNLL 2003 Named Entity Recognition (NER) dataset. This dataset consists of 200k training words which have been annotated as **Person**, **Organization**, **Location**, **Miscellaneous**, or **Other** (non-named entity).

Jim	Hen	##son	was	a	puppet	##eer
I-PER	I-PER	X	O	O	O	X

System	Dev F1	Test F1
ELMo+BiLSTM+CRF	95.7	92.2
CVT+Multi (Clark et al., 2018)	-	92.6
$\text{BERT}_{\text{BASE}}$	96.4	92.4
$\text{BERT}_{\text{LARGE}}$	**96.6**	**92.8**

Table 3: CoNLL-2003 Named Entity Recognition results. The hyperparameters were selected using the Dev set, and the reported Dev and Test scores are averaged over 5 random restarts using those hyperparameters.

Table in (Devlin *et al.*, 2018)

Ablation Study (1) – on pre-train tasks

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Table 5: Ablation over the pre-training tasks using the $\text{BERT}_{\text{BASE}}$ architecture. "No NSP" is trained without the next sentence prediction task. "LTR & No NSP" is trained as a left-to-right LM without the next sentence prediction, like OpenAI GPT. "+ BiLSTM" adds a randomly initialized BiLSTM on top of the "LTR + No NSP" model during fine-tuning.

Table in (Devlin *et al.*, 2018)

Ablation Study (2) – on model sizes

Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

Table 6: Ablation over BERT model size. #L = the number of layers; #H = hidden size; #A = number of attention heads. "LM (ppl)" is the masked LM perplexity of held-out training data.

Table in (Devlin *et al.*, 2018)

Ablation Study (3) – on pre-training steps

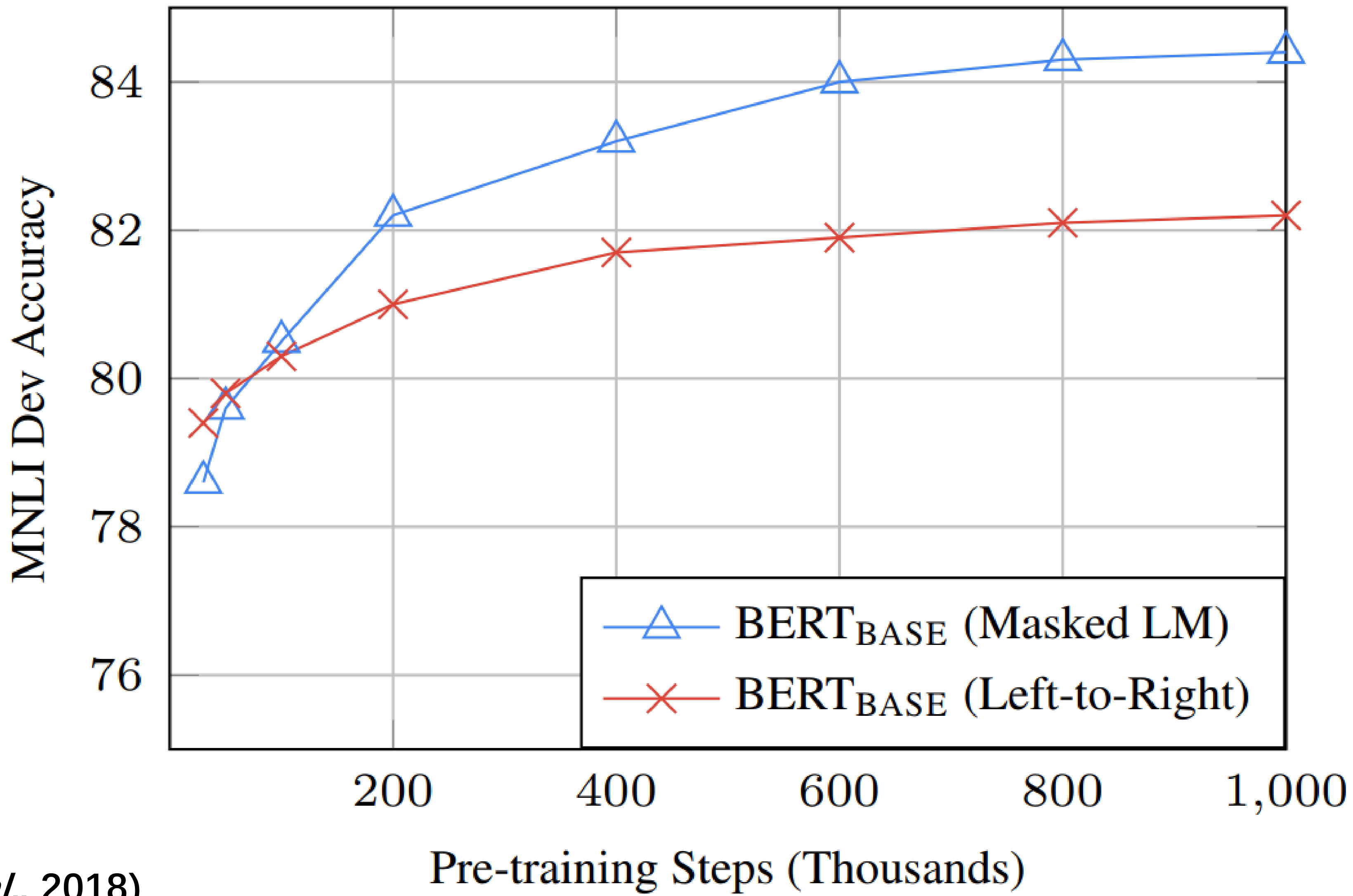


Figure in (Devlin *et al.*, 2018)

Ablation Study (4) – using BERT as feature extractor (*without* fine-tuning)

Layers	Dev F1
Finetune All	96.4
First Layer (Embeddings)	91.0
Second-to-Last Hidden	95.6
Last Hidden	94.9
Sum Last Four Hidden	95.9
Concat Last Four Hidden	96.1
Sum All 12 Layers	95.5

Table 7: Ablation using BERT with a feature-based approach on CoNLL-2003 NER. The activations from the specified layers are combined and fed into a two-layer BiLSTM, without backpropagation to BERT.

Table in (Devlin *et al.*, 2018)

Why BERT works?

- Leveraging huge unlabeled and high quality data: 7000 books + Wikipedia (together 3300M words)

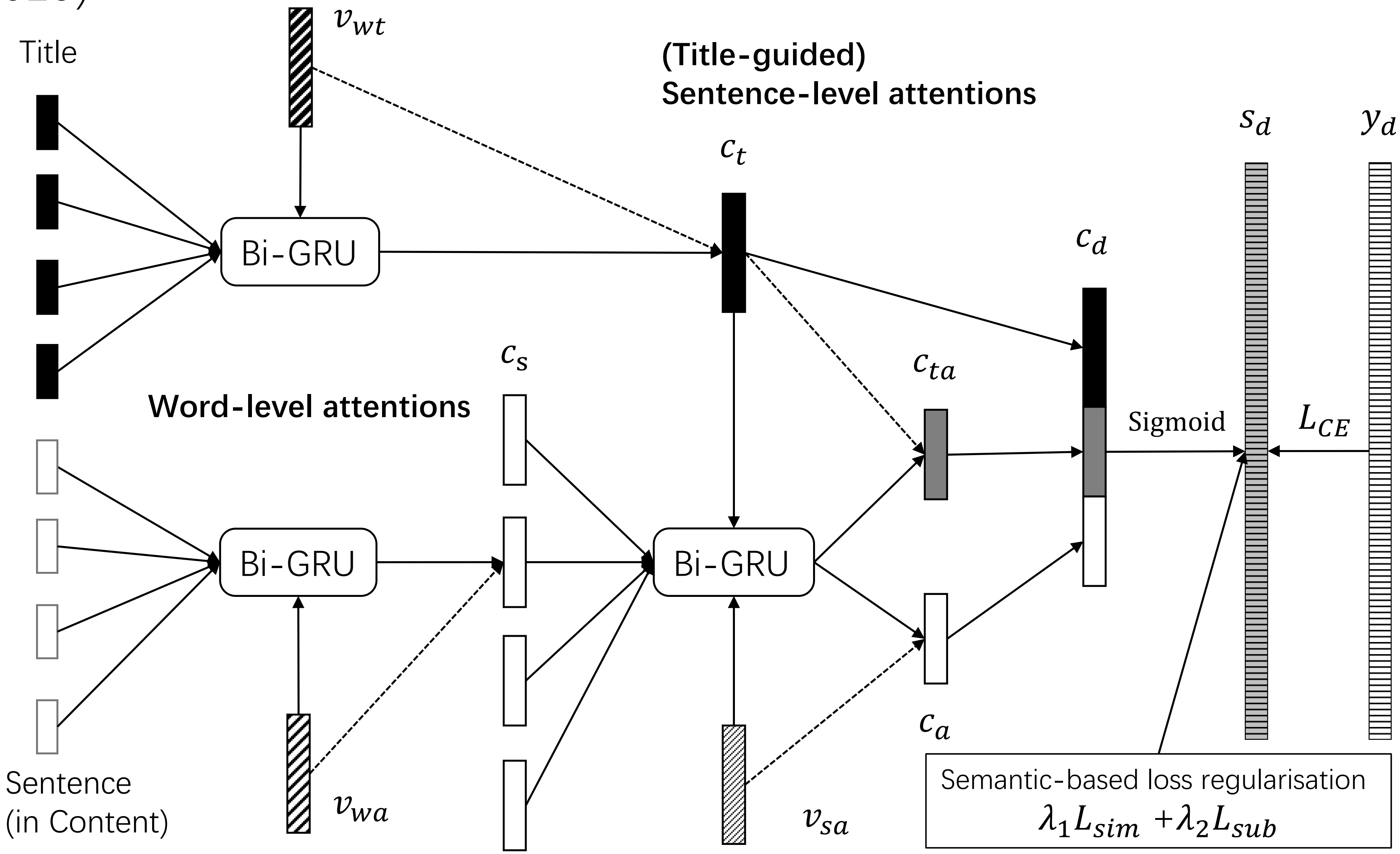
- Multi-head self-attention blocks in Transformer:
    - modelling the intra- and extra- word-word relations
    - parallelable within instance and thus efficient

- Task similarity: masked language modelling + next sentence prediction

How to improve BERT?

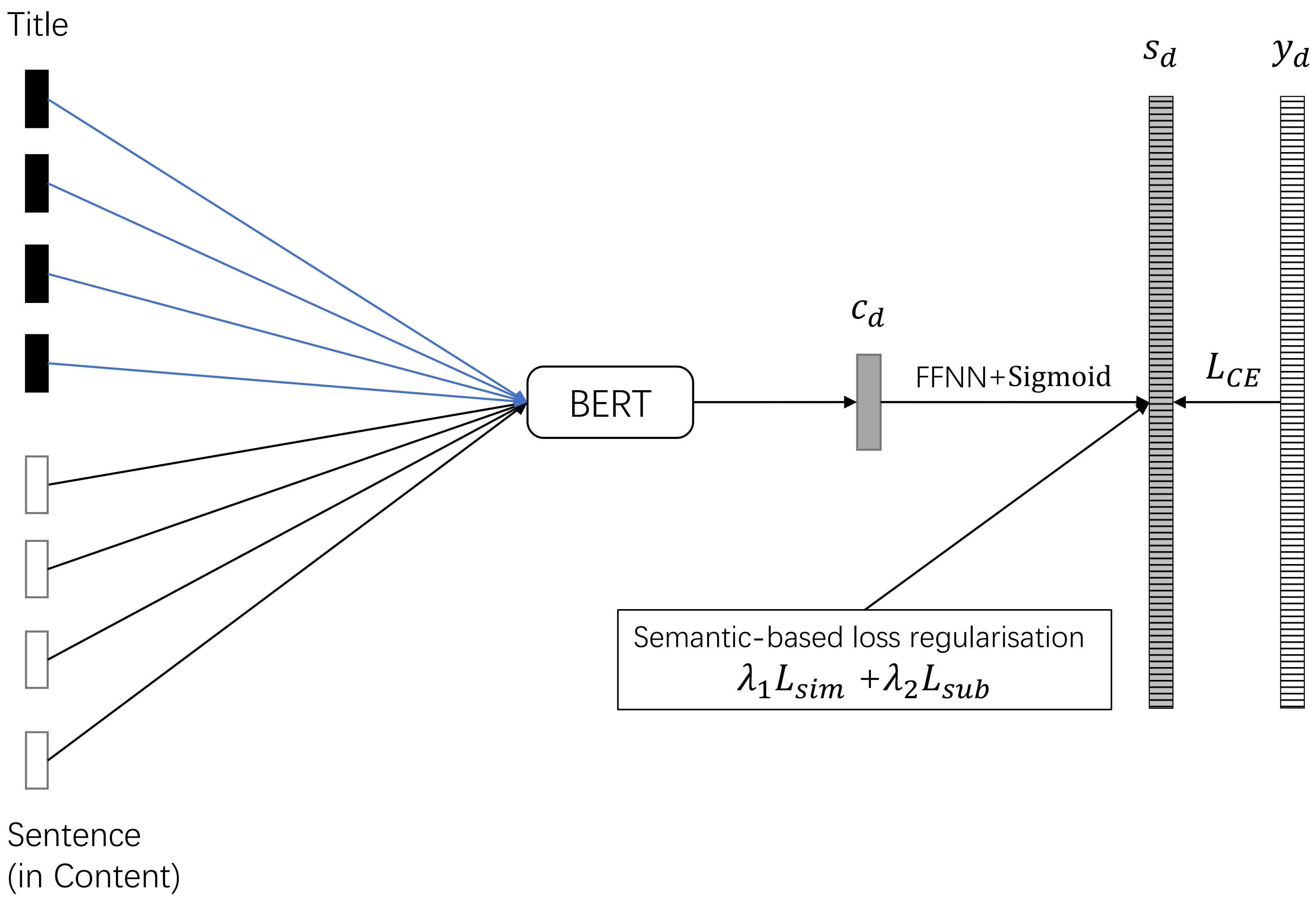
- **Pre-training**
  - Better tasks for pre-training for more complex usage
  - Better (larger, high-quality) data
  - Cross-lingual BERT for unsupervised learning (Lample & Conneau, 2019)
  - Even larger model, GPT-2: zero shot to outperform the SOTA (Radford *et al.*, 2018b)
- **Fine-tuning**
  - Better loss in fine-tuning
  - Introduce new tasks in fine-tuning

An architecture for multi-label classification
(Dong, 2019)



In H. Dong, W. Wang, H. Kaizhu, F. Coenen. Joint Multi-Label Attention Networks for Social Text Annotation, in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (***NAACL-HLT 2019***), Volume 2 (Short Papers), Minneapolis, USA, 2-7 June, 2019.

Is it possible? Any further thought?



Recommended Learning Resources

- Jay Alammar. **The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)**. Dec 2018. http://jalammar.github.io/illustrated-bert/
- Jay Alammar. **The Illustrated Transformer.** http://jalammar.github.io/illustrated-transformer/. June 2018.
- Ashish Vaswani and Anna Huang. **Transformers and Self-Attention For Generative Models**. Feb 2019. CS224n. Stanford University. http://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture14-transformers.pdf
- Kevin Clark. Future of NLP + Deep Learning. Mar 2019. CS224n. Stanford University. http://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture20-future.pdf
- keitakurita. **Paper Dissected: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Explained** http://mlexplained.com/2019/01/07/paper-dissected-bert-pre-training-of-deep-bidirectional-transformers-for-language-understanding-explained/
- keitakurita. **Paper Dissected: "Attention is All You Need" Explained** http://mlexplained.com/2017/12/29/attention-is-all-you-need-explained/

References

- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). **BERT: Pre-training of deep bidirectional transformers for language understanding.** In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)
- Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). **Deep Contextualized Word Representations**. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (Vol. 1, pp. 2227-2237).
- Lample, G., & Conneau, A. (2019). **Cross-lingual Language Model Pretraining.** arXiv preprint arXiv:1901.07291.
- Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018a). **Improving Language Understanding by Generative Pre-Training**.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2018b). **Language models are unsupervised multitask learners**. Technical report, OpenAI.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). **Attention is all you need**. In Advances in Neural Information Processing Systems(pp. 5998-6008).
- Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., ... & Klingner, J. (2016). **Google's neural machine translation system: Bridging the gap between human and machine translation**. arXiv preprint arXiv:1609.08144.