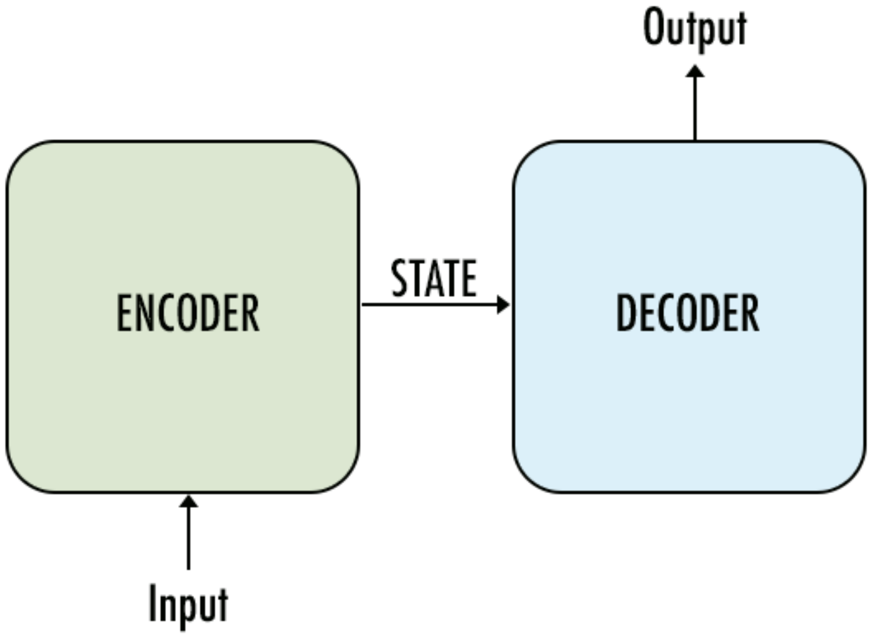
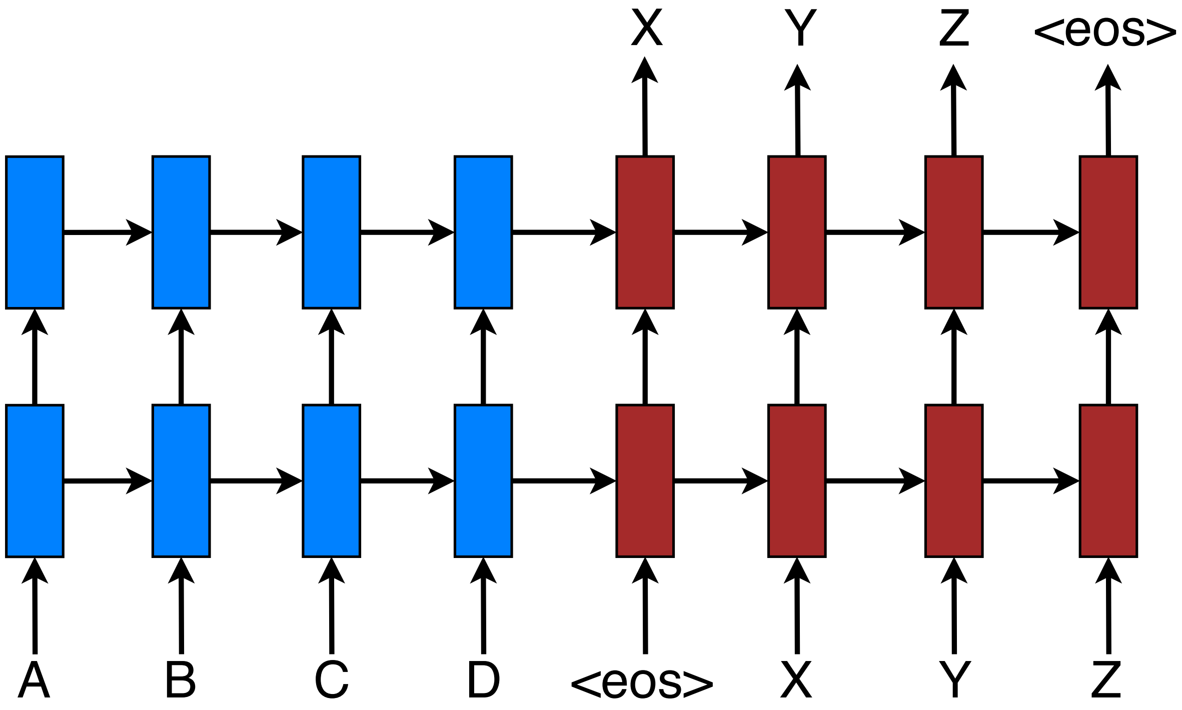
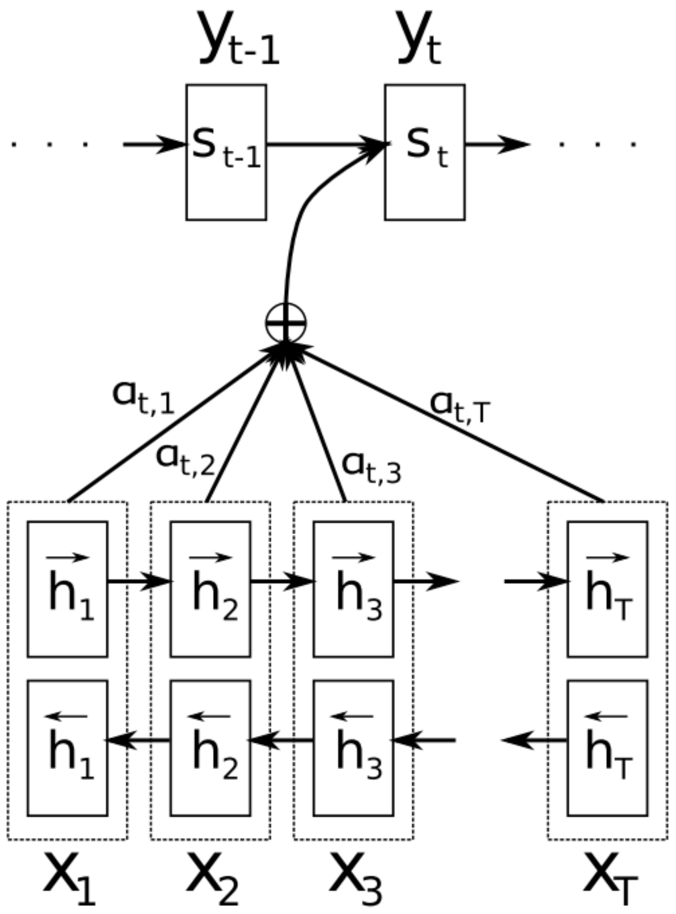
Attention Mechanism in Neural Networks

Encoder-Decoder Models

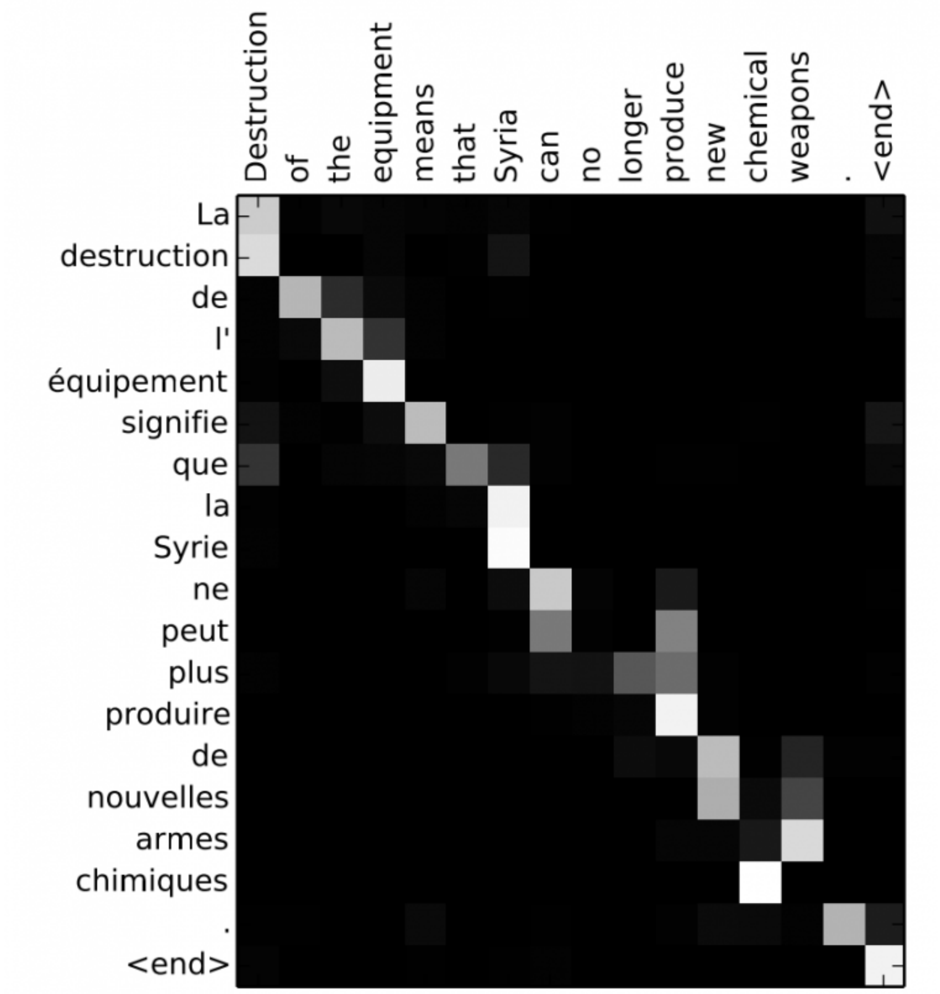




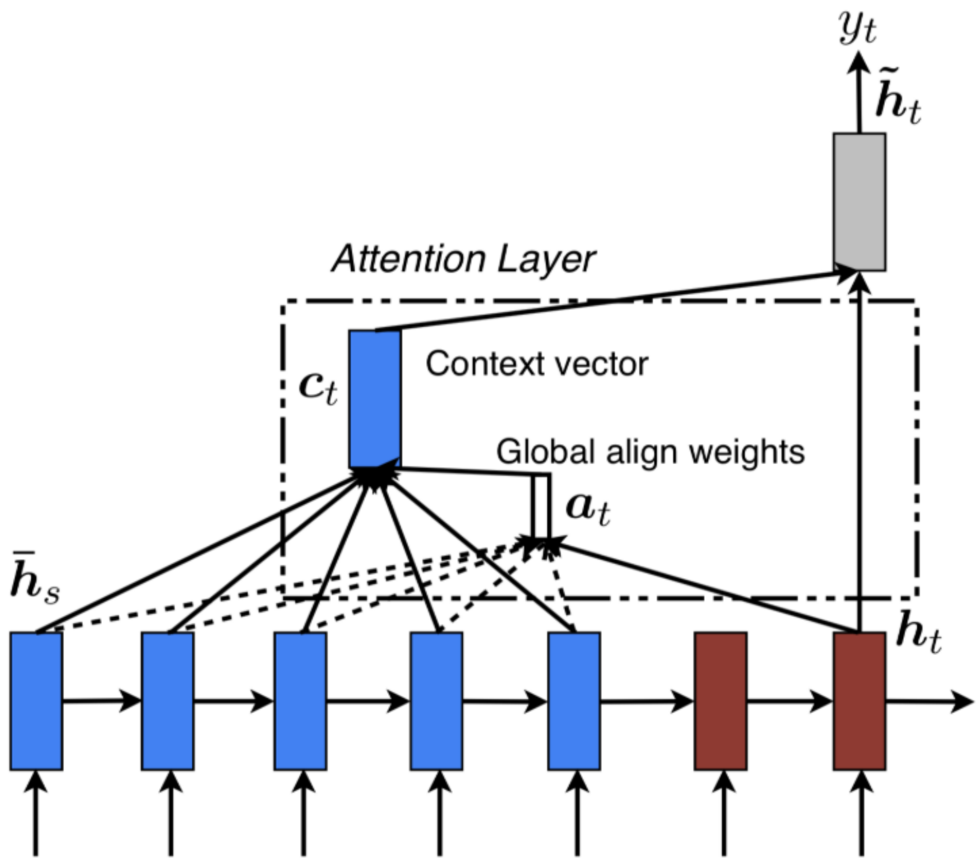
Encoder-Decoder with Attention



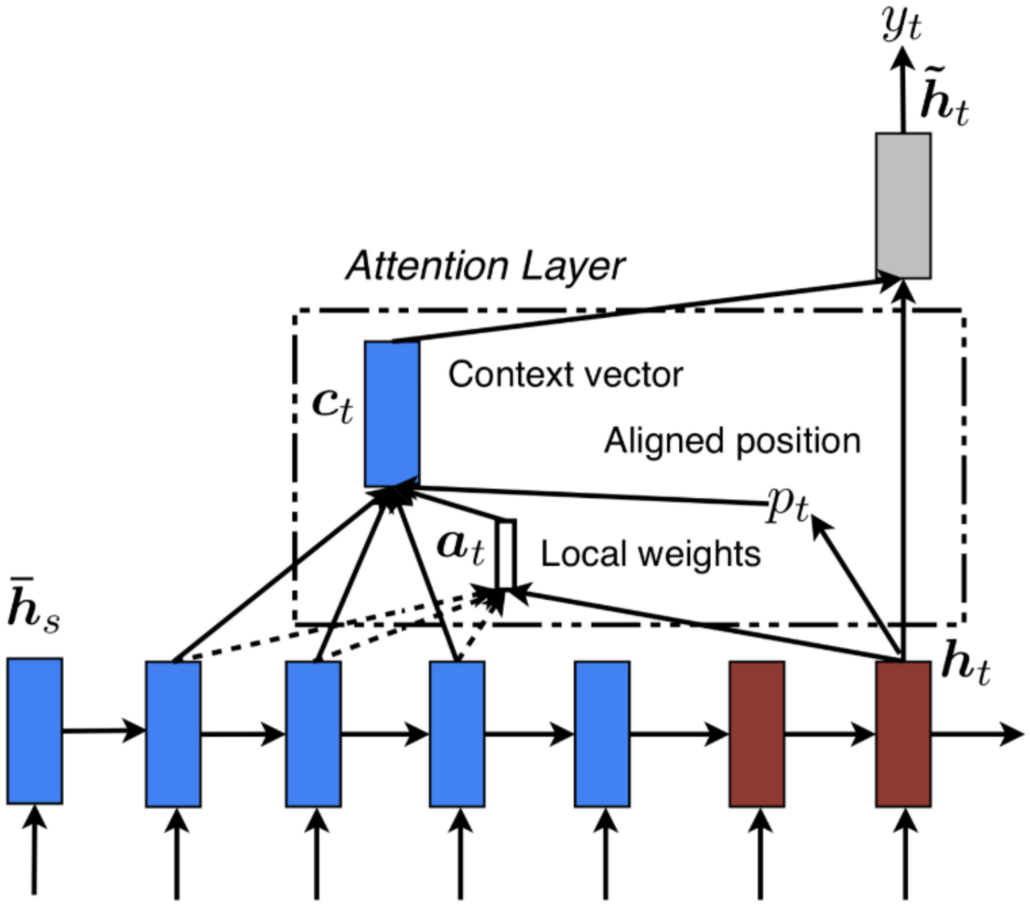
Encoder-Decoder with Attention



Attention Mechanisms



Global Attention Model



Local Attention Model

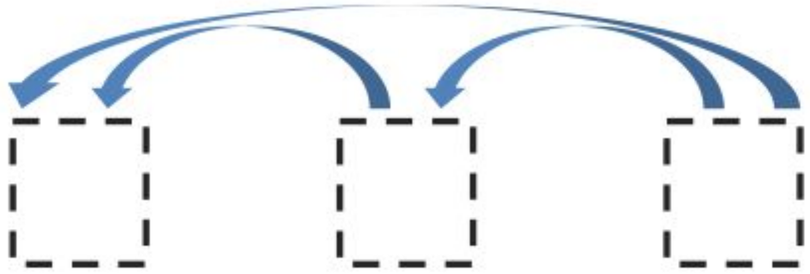
Computing Attention Weights

$$a_i = \frac{\exp(f_{att}(q, k_i))}{\sum_{j=1}^{|M|} \exp(f_{att}(q, k_j))}$$

Three ways of attention

Encoder-Decoder Attention

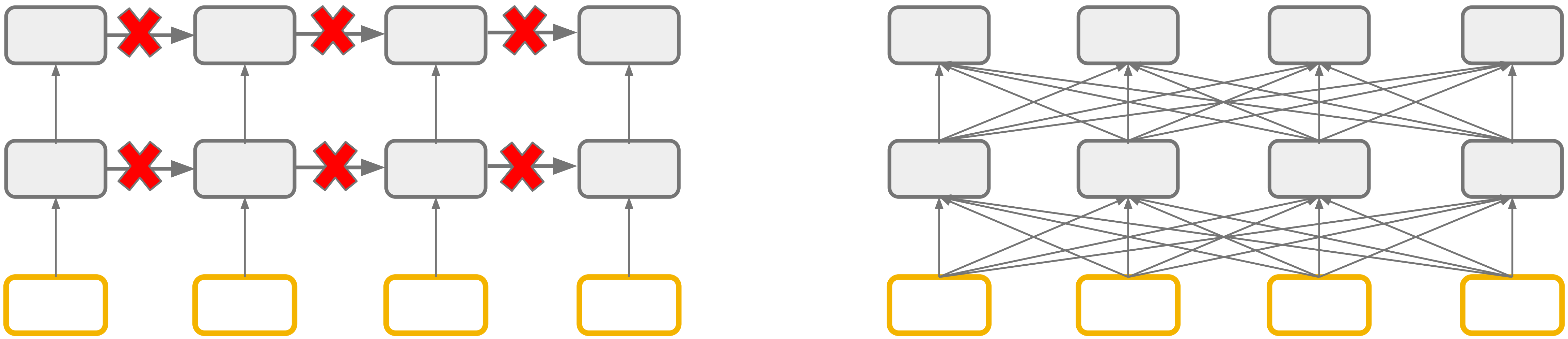
Encoder Self-Attention



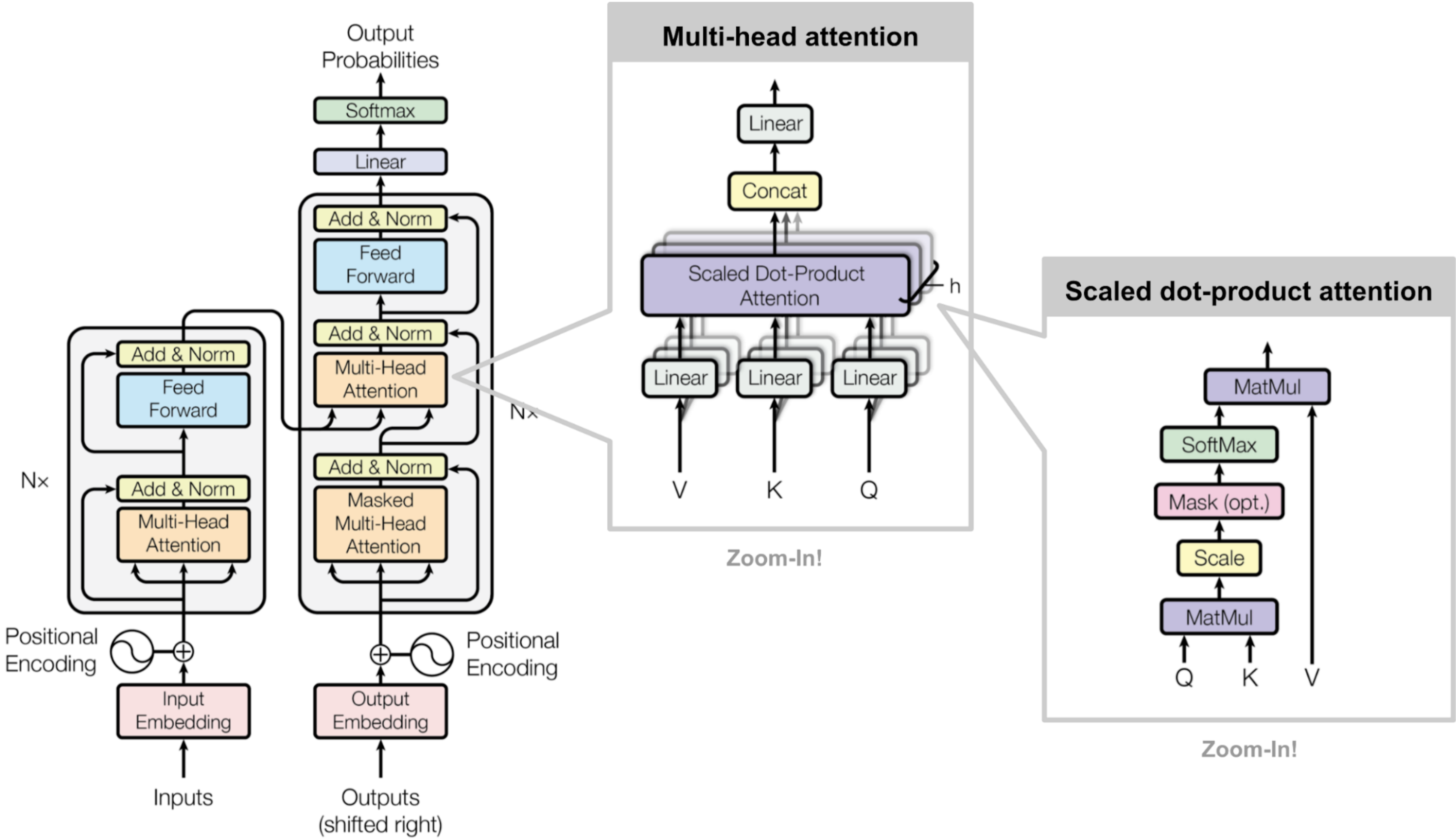
MaskedDecoder Self-Attention

Transformer

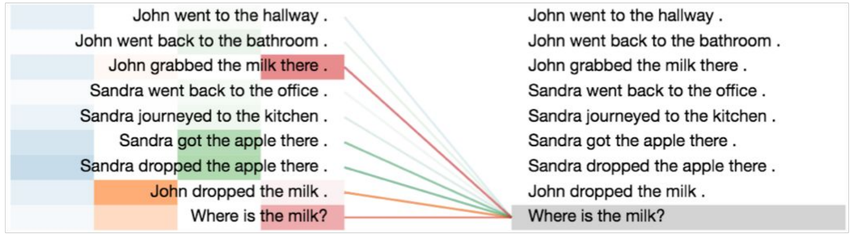
Transformers



Transformers



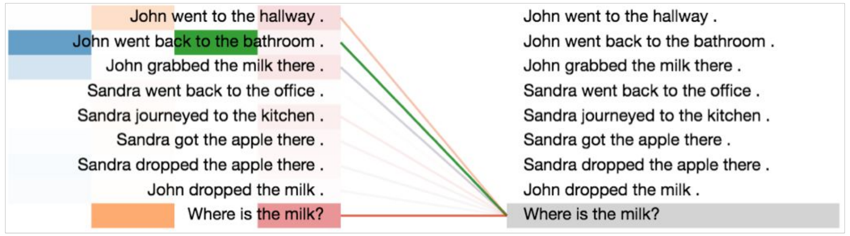
Attention Visualization



(a) Step 1

(b) Step 2

(c) Step 3



(d) Step 4

Related Papers

- Attention for Machine Translation
  - https://nlp.stanford.edu/pubs/emnlp15_attn.pdf
- Transformer
  - https://arxiv.org/abs/1706.03762
- Universal Transformer
  - https://arxiv.org/abs/1807.03819
- Transformer-XL
  - https://arxiv.org/abs/1901.02860

Useful Resources

- Source Codes:
  - https://github.com/tensorflow/tensor2tensor
  - https://github.com/tensorflow/models/tree/master/official/transformer/model
- Visualization of attention heads for BERT:
  - https://github.com/jessevig/bertviz
- Nice blogposts:
  - https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html
  - http://jalammar.github.io/illustrated-transformer/
  - http://nlp.seas.harvard.edu/2018/04/03/attention.html
  - http://jalammar.github.io/illustrated-bert/
- Blogpost comparing Transformers with Capsule Networks:
  - https://staff.fnwi.uva.nl/s.abnar/?p=108

Bidirectional Encoder Representations from Transformers (BERT)

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova

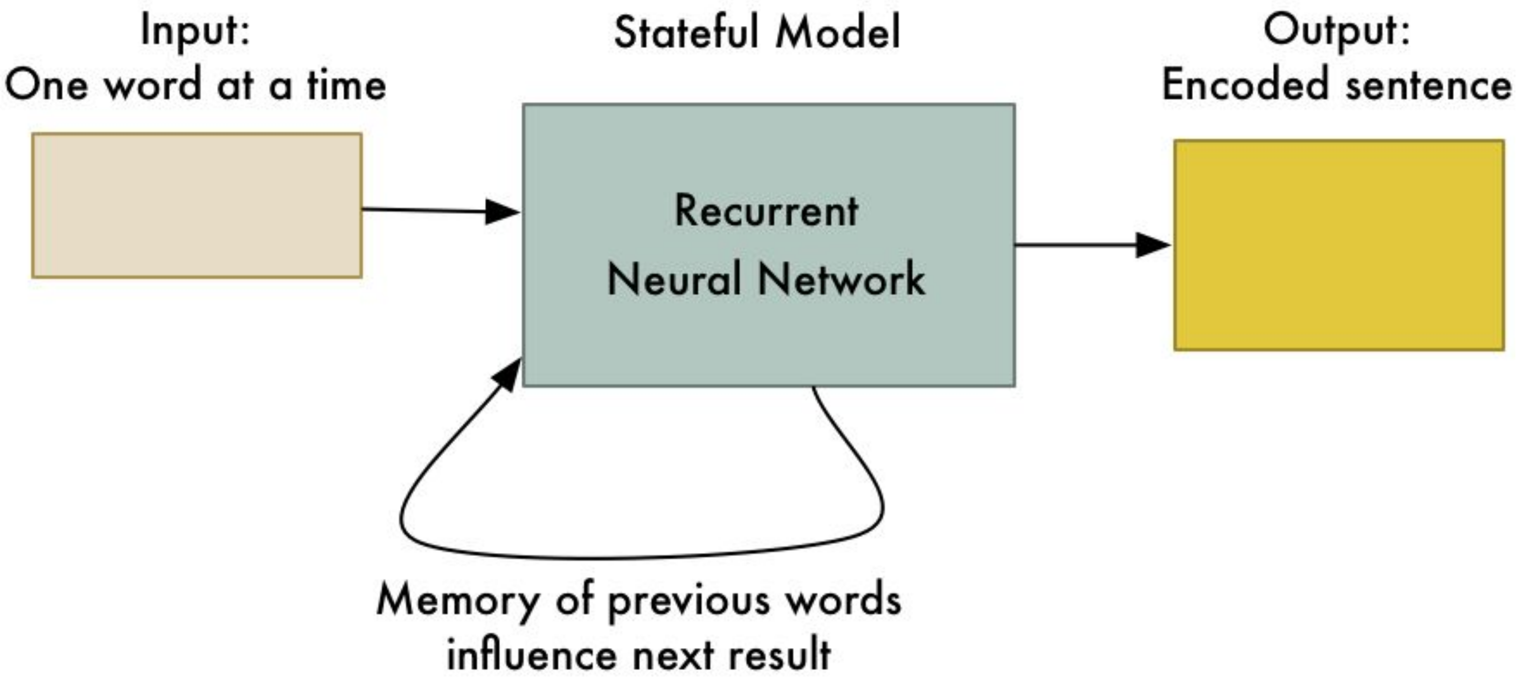
Google AI Language

Presented by: Shantanu Chandra

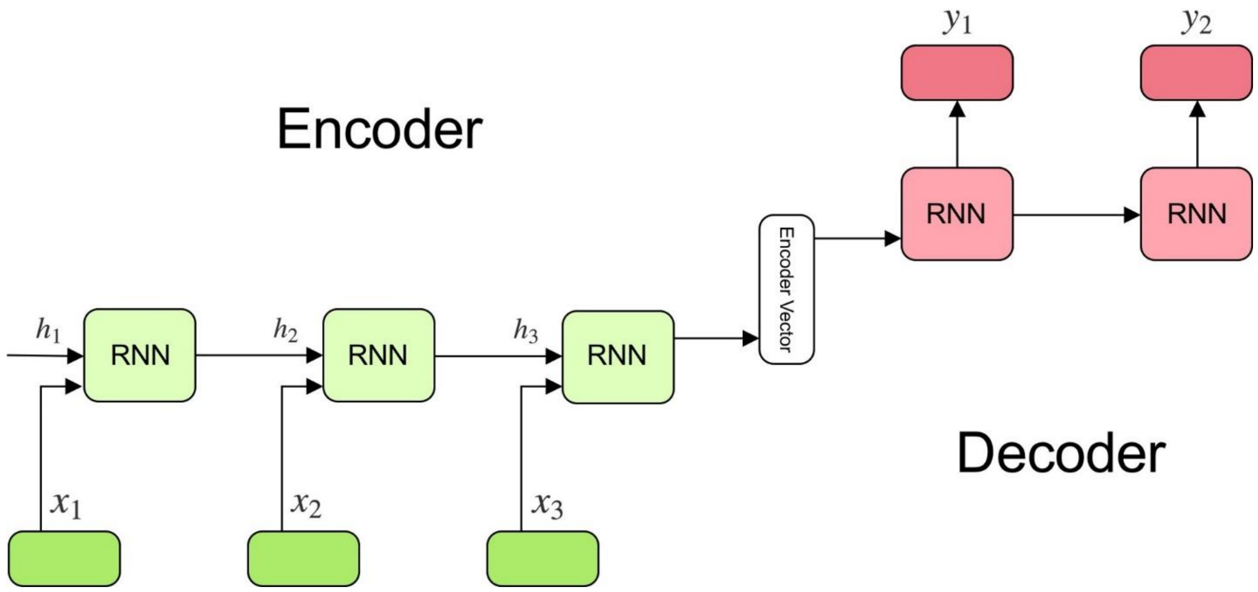
Agenda

- Recap – Traditional Language Models
- Limitations
- BERT to the rescue!
- BERT Architecture and Training
- BERT Results and Discussion
- Summary
- Future work

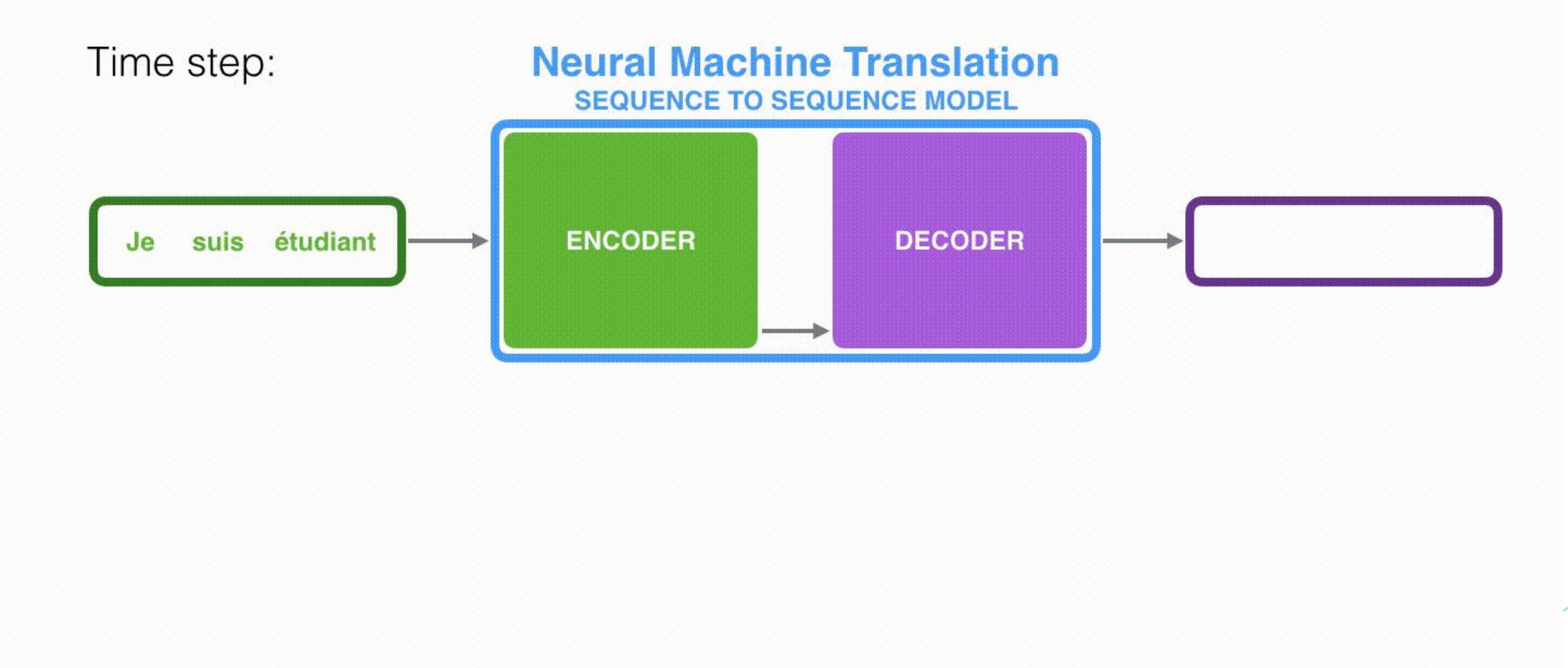
Traditional Seq2Seq – recap



Traditional Seq2Seq

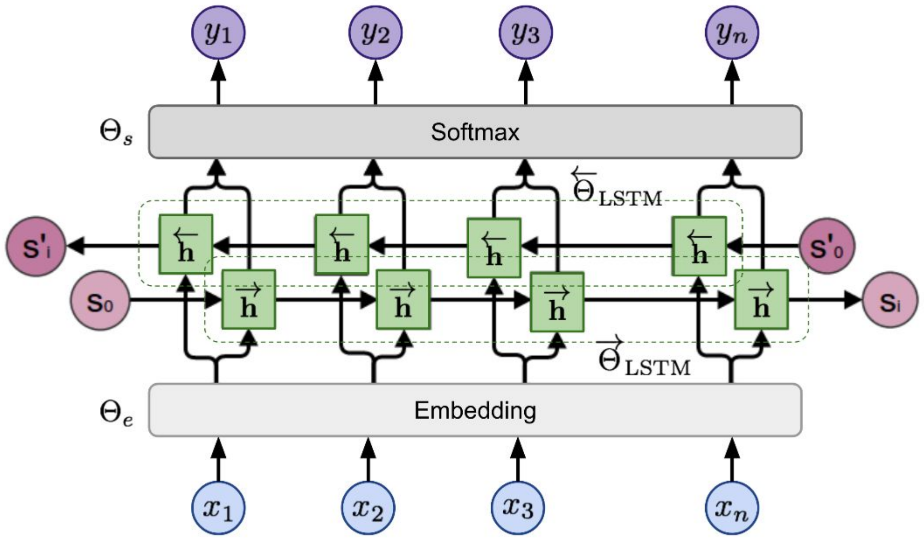


Traditional Seq2Seq



ELMo

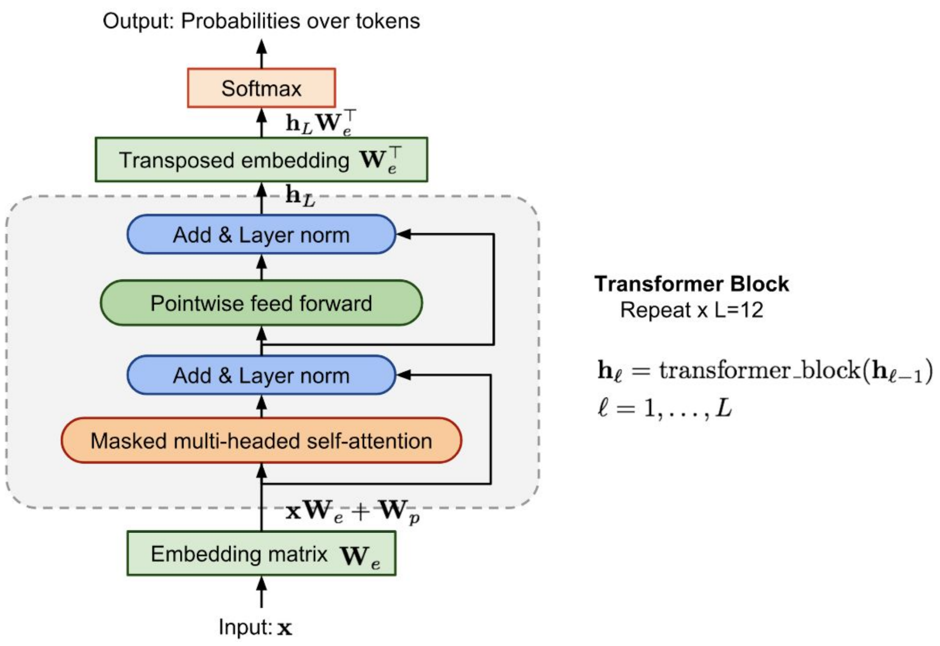
- Bi-directional LSTM
- Lower layers – Syntactic
- Higher layers – Semantic
- Feature based use – feed embeddings to model
- Dependency on task-customized models



OpenAI GPT

- Multi-layer Transformer architecture

(not shallow concat of independently trained LSTMs)



- Fine tune usage – fine tune the same base model for all end tasks

- Attention applied left-to-right, hence unidirectional!

BERT to the rescue!

- B – Bidirectional
- E – Encoder
- R- Representation
- T- Transformers

BERT

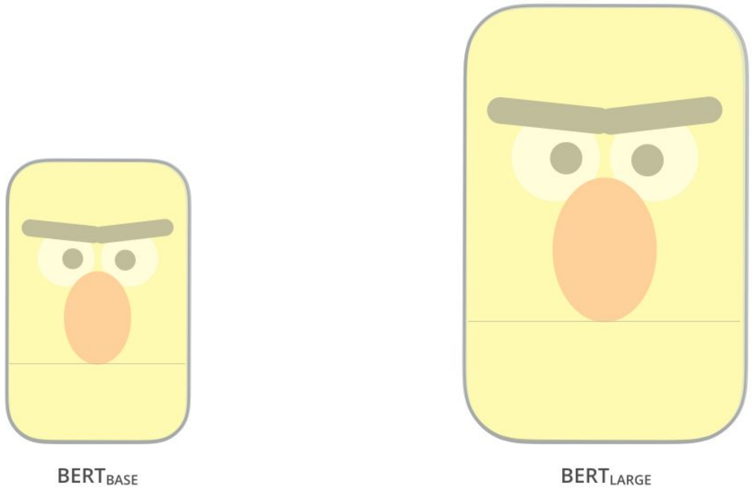
- **Bidirectional**
  - naturally bidirectional in all the layers. (ELMo is “shallow bidirectional”)
- **General purpose representations:** plug-and-play in real sense
- **Transformer Architecture** (new paradigm!)
- **Novel training methods -** not one, but TWO!!

BERT in numbers

- Pushed the GLUE benchmark to **80.4%** (**7.6%** absolute improvement).
- Pushed MultiNLI accuracy to **86.7%** (**5.6%** absolute improvement).
- Pushed the SQuAD v1.1 question-answering Test F1 to **93.2** (**1.5** absolute improvement), outperforming human performance by **2.0**.

BERT in numbers

- BERT - Base shattered OpenAI GPT
- BERT - Large beat BERT-Base

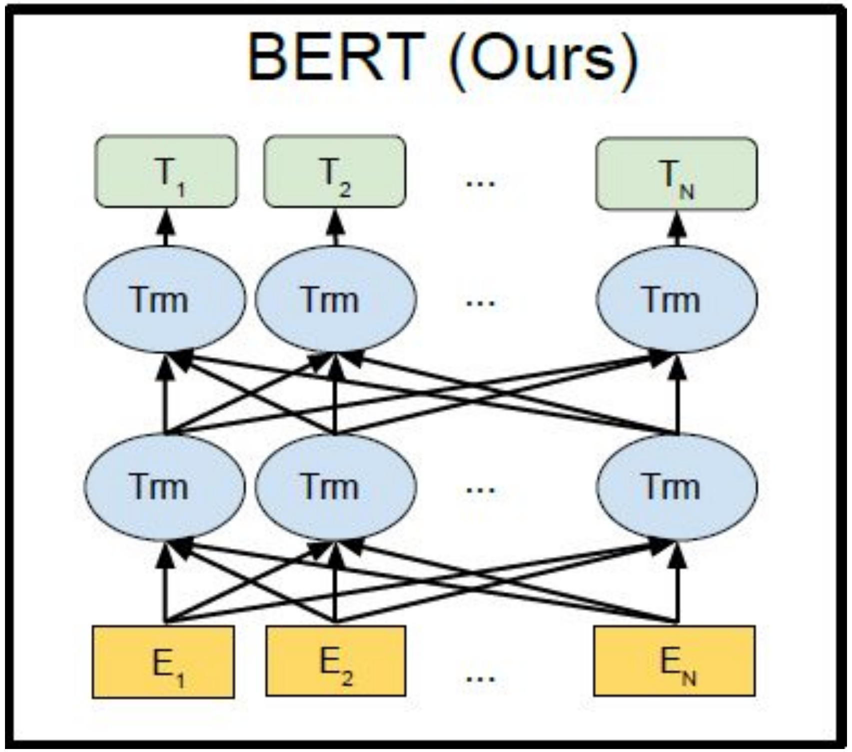


BERT to the rescue!

- B – Bidirectional
- E – Encoder
- R- Representation
- T- Transformers

B – Bidirectional

- Actually *non-directional*



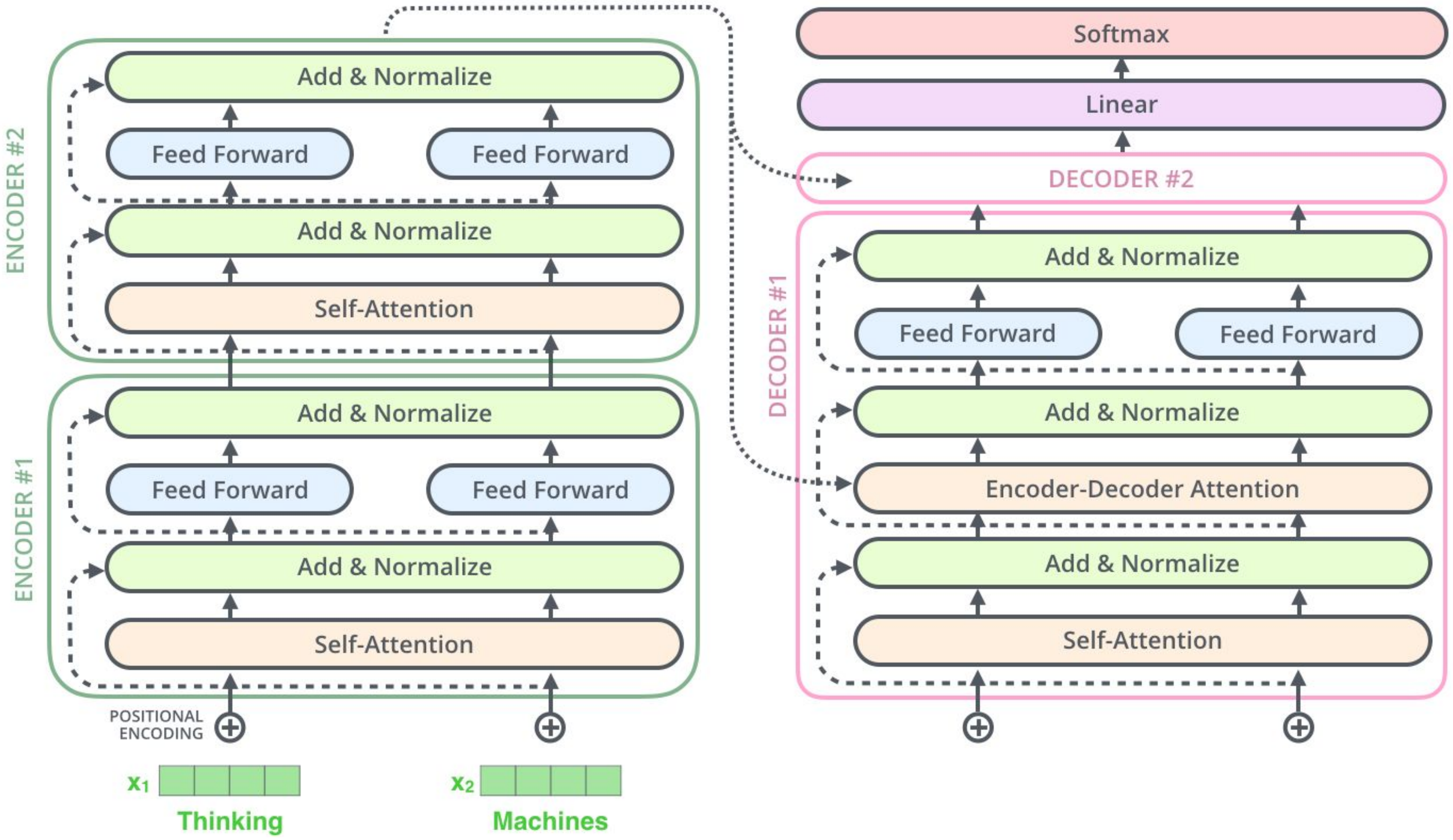
Context is everything

- No Context (Word2Vec)
  - River **[bank]**
  - **[Bank]** deposit
- Left-to-right context
  - I made a [bank] deposit = I made a **[….]**
- **Bi-directional context**
  - I made a [bank] deposit = I made a **[….]** deposit

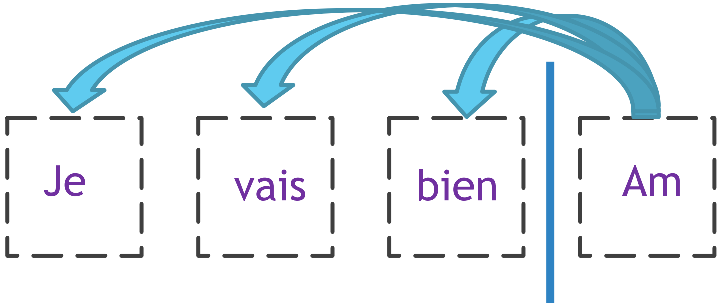
BERT to the rescue!

- B – Bidirectional
- E – Encoder
- R- Representation
- T- Transformers

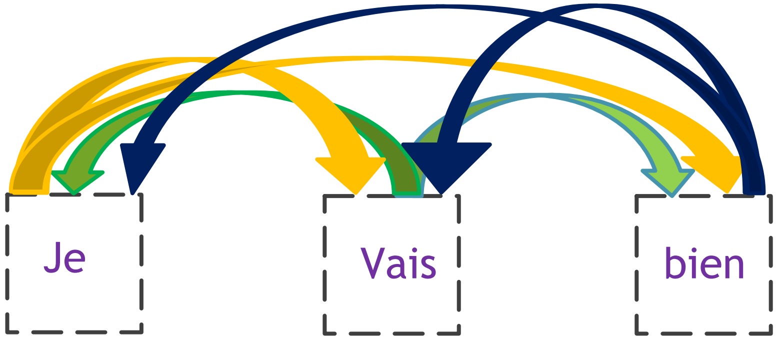
Transformer Block



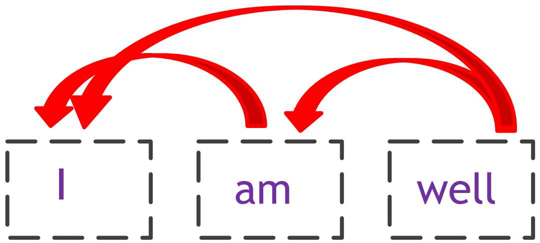
Transformer: Types of Attention



Encoder-Decoder Attention



Encoder-Self Attention

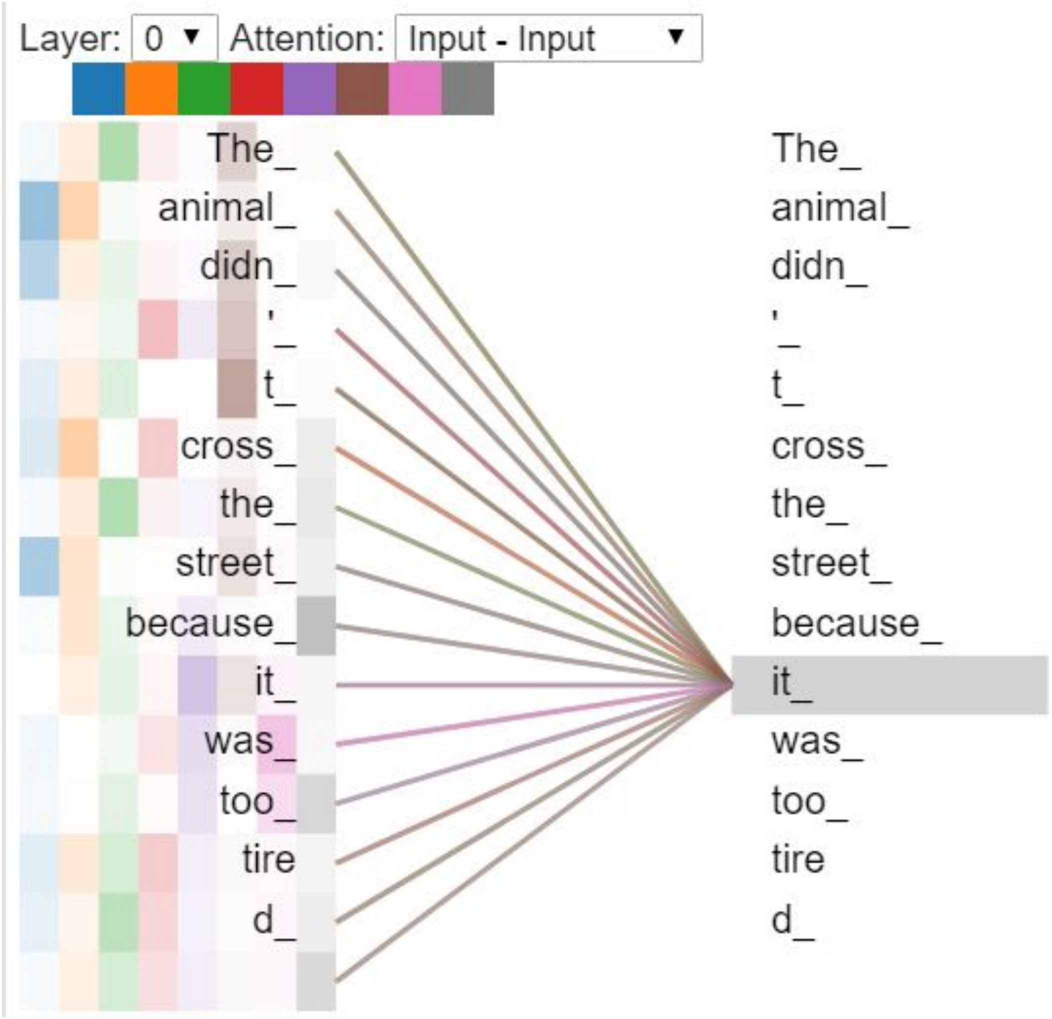


Decoder - Self Attention

Transformer: in action

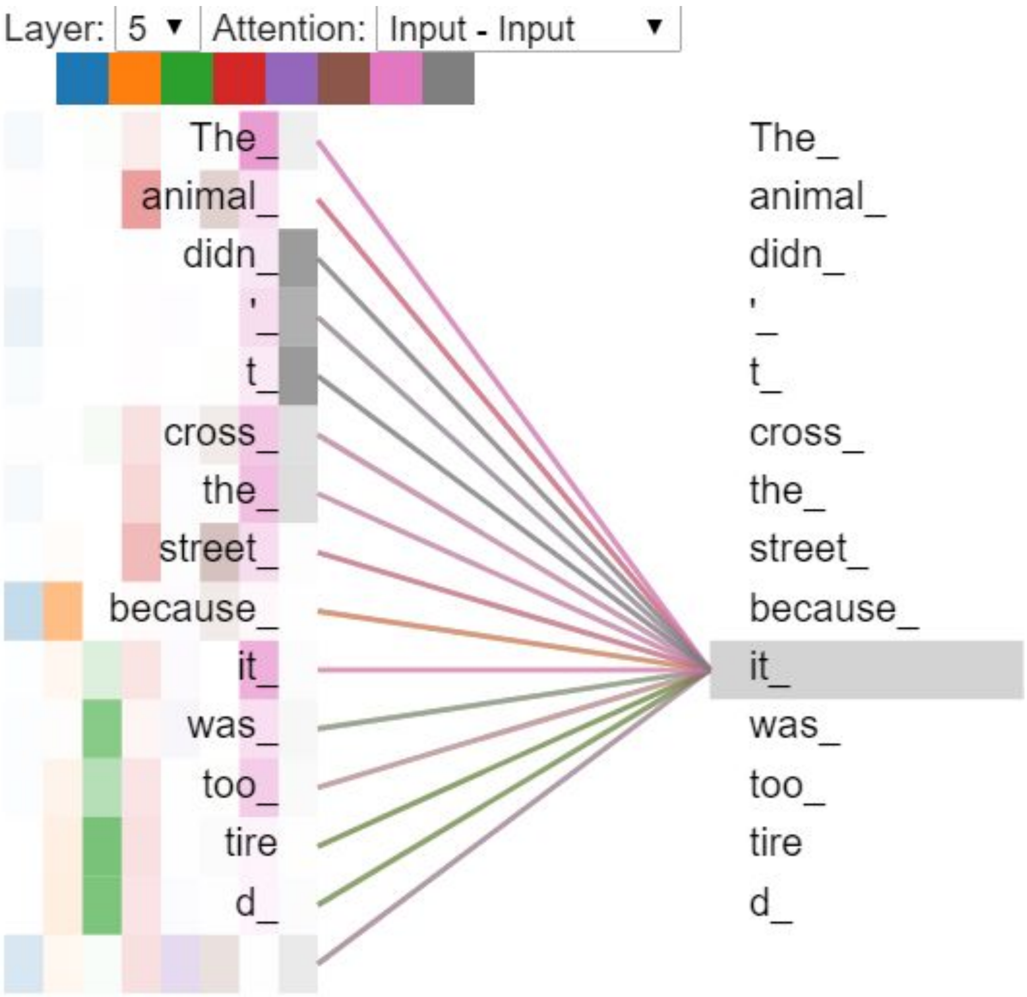
Source: [Google AI Blog]

Transformer: Self-Attention

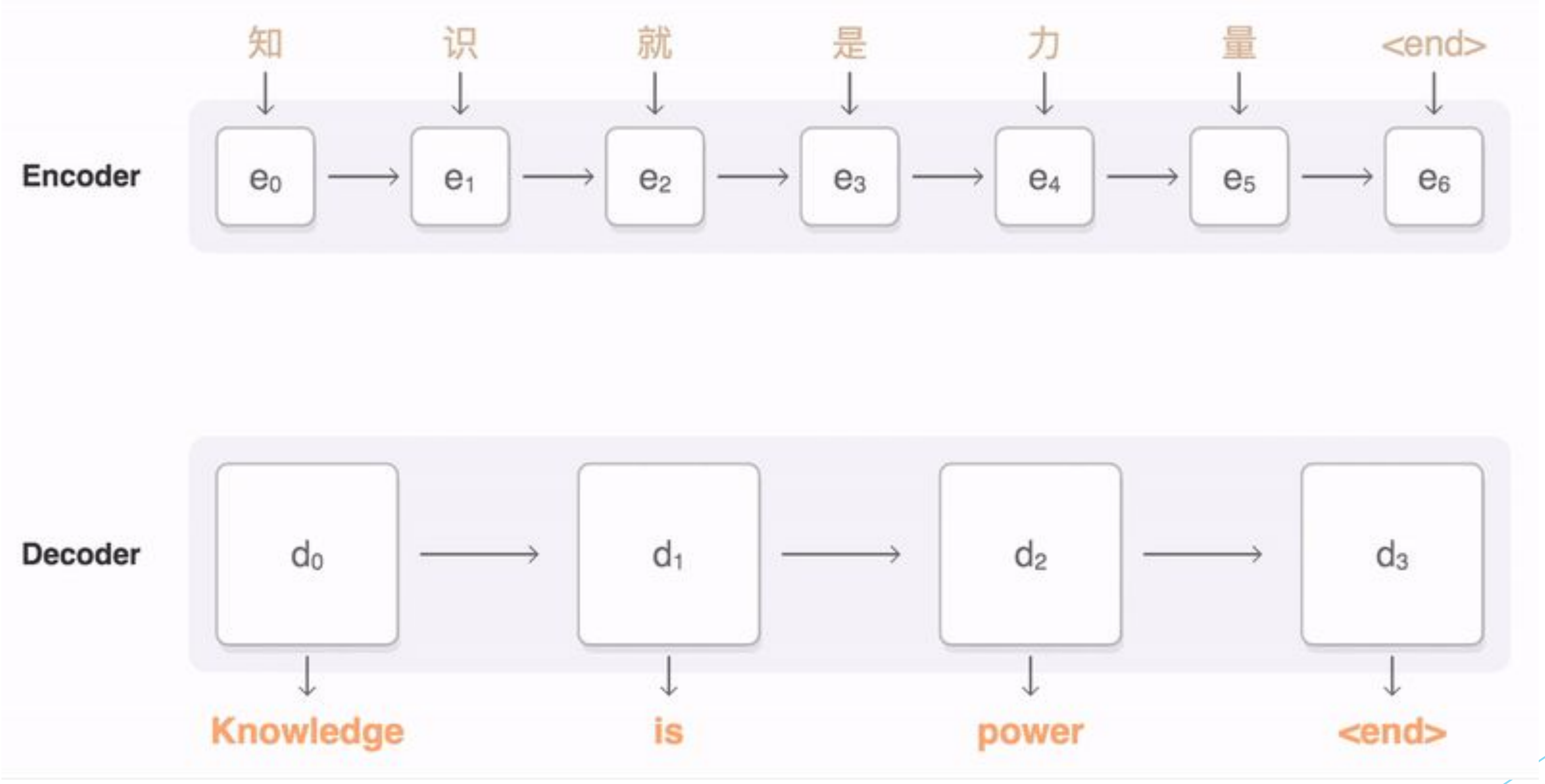


Source: Tensor2Tensor

Transformer: Self-Attention

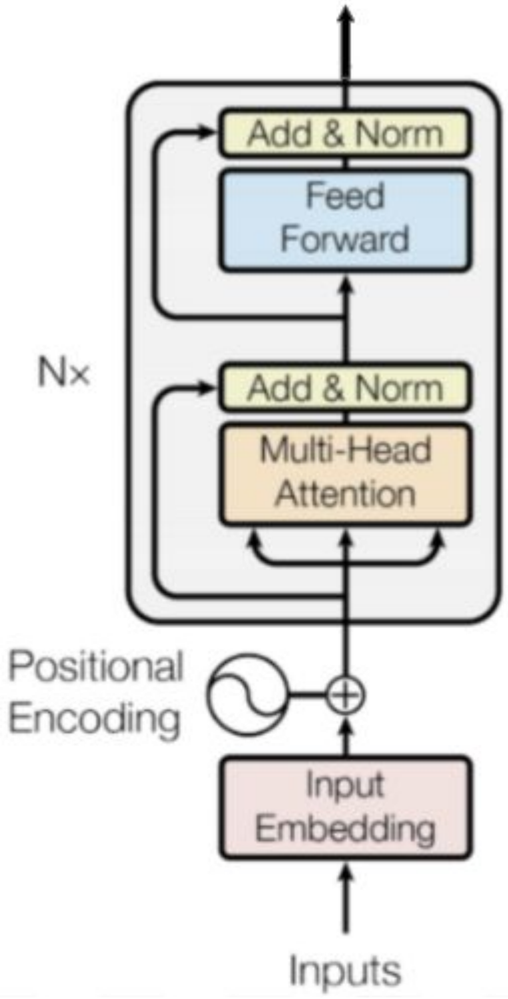


Transformer: Self-Attention

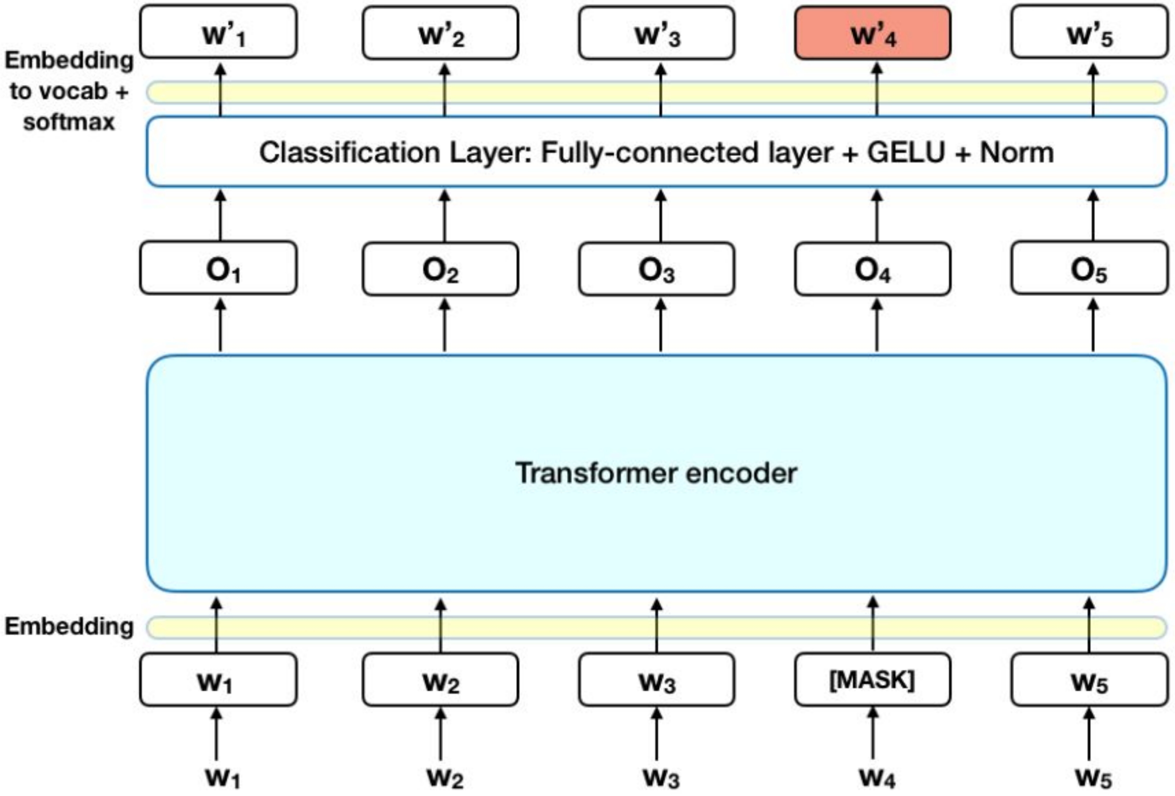


BERT - Architecture

- Transformers - Encoder blocks only
- No weight sharing
- REMEMBER – attention mechanism!
- NOTE – residual connections



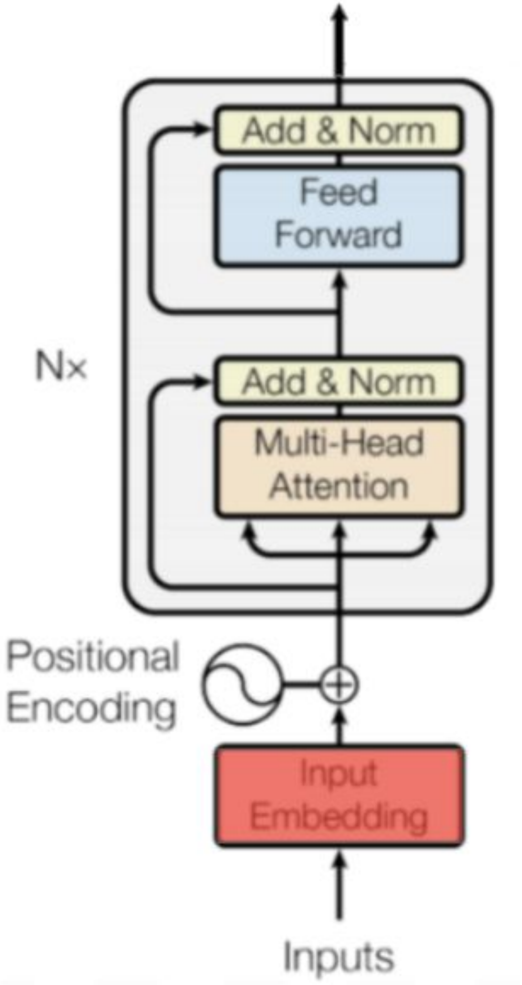
BERT – Architecture



BERT – Input

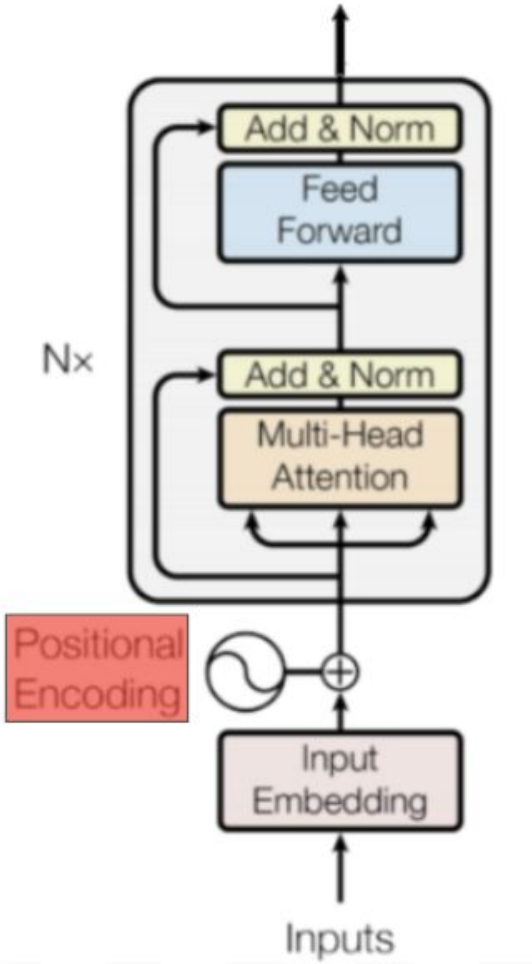
- Input = $(inp\_len \times emb\_dim)$
- Output = $(inp\_len \times emb\_dim)$
- NOTE: padding for equal length of batch

$$
\boldsymbol{Z} = \begin{array}{c} \\ Hello \\ , \\ how \\ are \\ you \\ ? \end{array}
\begin{array}{c}
\begin{array}{ccccc} < & - & d_{emb\_dim} & - & > \end{array} \\
\begin{pmatrix}
123.4 & 0.32 & \cdots & 94 & 32 \\
83 & 34 & \cdots & 77 & 19 \\
0.2 & 50 & \cdots & 33 & 30 \\
289 & 432.98 & \cdots & 150 & 92 \\
80 & 46 & \cdots & 23 & 32 \\
41 & 21 & \cdots & 74 & 33
\end{pmatrix}
\end{array}
$$



BERT – Positional encoding

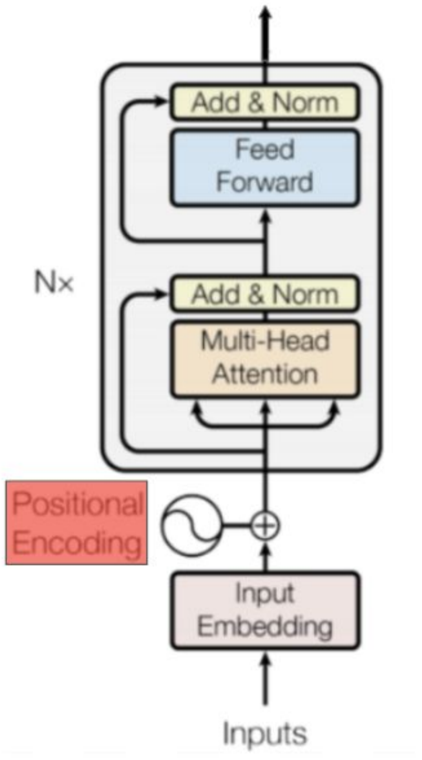
- Non-sequential input
- Hence Positional Encoding
- Non-learned, pre-determined sinusoidal functions [-1,1]
- Thus, same word in different position has different embedding in the SAME sentence.



BERT – Positional encoding

$$p_{i,j} = \begin{cases} \sin\left(\dfrac{i}{10000^{\frac{j}{d_{emb\_dim}}}}\right) & \text{if } j \text{ is even} \\ \cos\left(\dfrac{i}{10000^{\frac{j-1}{d_{emb\_dim}}}}\right) & \text{if } j \text{ is odd} \end{cases}$$

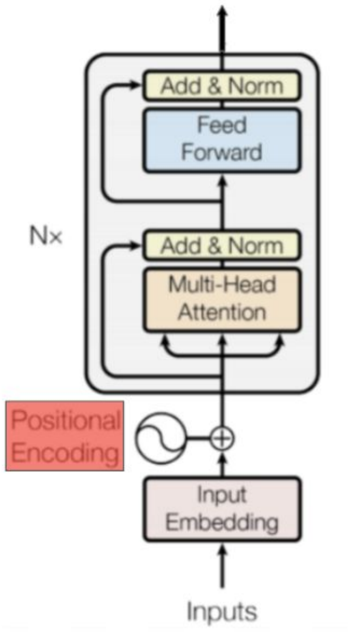
$$\mathbf{P} = \begin{array}{c} \\ Hello \\ , \\ how \\ are \\ you \\ ? \end{array} \begin{array}{c} < \quad - \quad d_{emb\_dim} \quad - \quad > \\ \begin{pmatrix} \sin\left(\frac{0}{10000^{\frac{0}{emb_{dim}}}}\right) & \cos\left(\frac{0}{10000^{\frac{0}{emb_{dim}}}}\right) & \sin\left(\frac{0}{10000^{\frac{2}{emb_{dim}}}}\right) & \cos\left(\frac{0}{10000^{\frac{2}{emb_{dim}}}}\right) & \cdots \\ \sin\left(\frac{1}{10000^{\frac{0}{emb_{dim}}}}\right) & \cos\left(\frac{1}{10000^{\frac{0}{emb_{dim}}}}\right) & \sin\left(\frac{1}{10000^{\frac{2}{emb_{dim}}}}\right) & \cos\left(\frac{1}{10000^{\frac{2}{emb_{dim}}}}\right) & \cdots \\ \sin\left(\frac{2}{10000^{\frac{0}{emb_{dim}}}}\right) & \cos\left(\frac{2}{10000^{\frac{0}{emb_{dim}}}}\right) & \sin\left(\frac{2}{10000^{\frac{2}{emb_{dim}}}}\right) & \cos\left(\frac{2}{10000^{\frac{2}{emb_{dim}}}}\right) & \cdots \\ \sin\left(\frac{3}{10000^{\frac{0}{emb_{dim}}}}\right) & \cos\left(\frac{3}{10000^{\frac{0}{emb_{dim}}}}\right) & \sin\left(\frac{3}{10000^{\frac{2}{emb_{dim}}}}\right) & \cos\left(\frac{3}{10000^{\frac{2}{emb_{dim}}}}\right) & \cdots \\ \sin\left(\frac{4}{10000^{\frac{0}{emb_{dim}}}}\right) & \cos\left(\frac{4}{10000^{\frac{0}{emb_{dim}}}}\right) & \sin\left(\frac{4}{10000^{\frac{2}{emb_{dim}}}}\right) & \cos\left(\frac{4}{10000^{\frac{2}{emb_{dim}}}}\right) & \cdots \\ \sin\left(\frac{5}{10000^{\frac{0}{d_{emb_{dim}}}}}\right) & \cos\left(\frac{5}{10000^{\frac{0}{emb_{dim}}}}\right) & \sin\left(\frac{5}{10000^{\frac{2}{emb_{dim}}}}\right) & \cos\left(\frac{5}{10000^{\frac{2}{emb_{dim}}}}\right) & \cdots \end{pmatrix} \end{array}$$

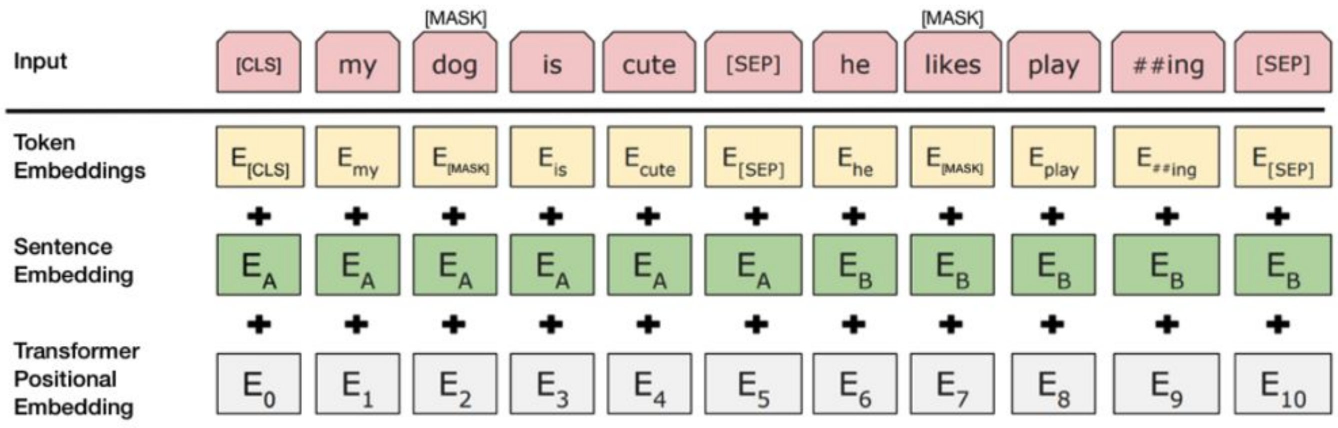


BERT – Model Input

- $X = Z + P$

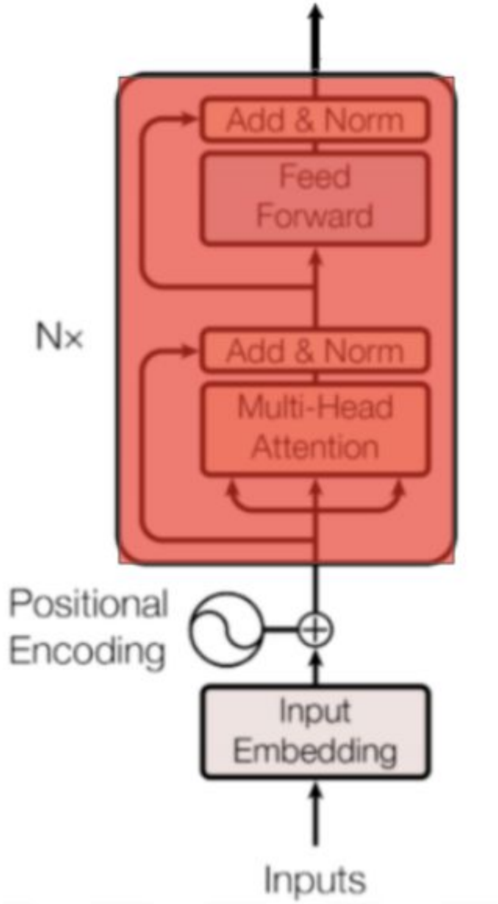
- Input to the encoder block





BERT – Encoder Block

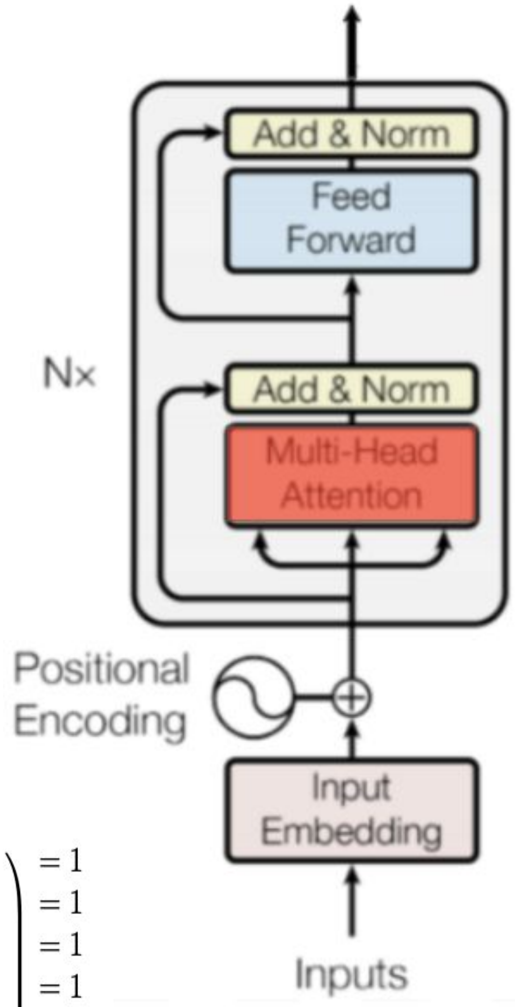
- **Multi-head Attention (h_i)** modules:
  - Multiple times,
  - ..with different weight matrices
  - ..then concat all and pass through linear



BERT – Encoder Block

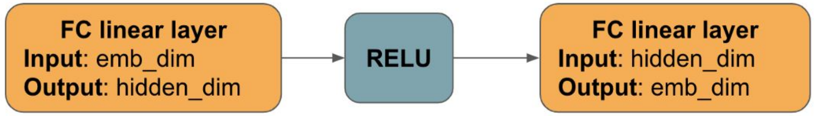
- **Multi-head Attention (h_i)** modules:
  - Multiple times,
  - ..with different weight matrices
  - ..finally concat all and pass through linear

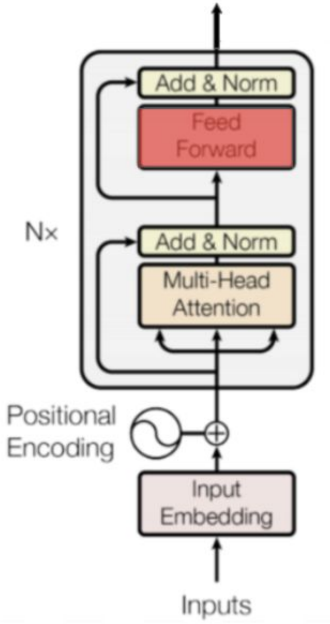
$$
\begin{array}{c}
 \\ Hello \\ , \\ how \\ are \\ you \\ ?
\end{array}
\begin{array}{c}
\begin{array}{cccccc} \quad Hello & , & how & are & you & ? \end{array} \\
\begin{pmatrix}
72.40 * 10^{-06} & 1.23 * 10^{-21} & 6.51 * 10^{-40} & 2.62 * 10^{-22} & 9.99 * 10^{-01} & 4.30 * 10^{-08} \\
1.00 * 10^{+00} & 7.51 * 10^{-30} & 1.54 * 10^{-17} & 9.91 * 10^{-13} & 8.15 * 10^{-69} & 1.09 * 10^{-30} \\
3.12 * 10^{-70} & 2.51 * 10^{-51} & 2.72 * 10^{-21} & 8.03 * 10^{-09} & 1.29 * 10^{-07} & 9.99 * 10^{-01} \\
2.47 * 10^{-72} & 5.54 * 10^{-05} & 9.80 * 10^{-01} & 1.98 * 10^{-02} & 2.77 * 10^{-82} & 2.58 * 10^{-08} \\
2.67 * 10^{-05} & 1.21 * 10^{-09} & 9.75 * 10^{-07} & 3.17 * 10^{-76} & 9.99 * 10^{-01} & 3.64 * 10^{-28} \\
8.59 * 10^{-47} & 1.05 * 10^{-35} & 9.99 * 10^{-01} & 2.38 * 10^{-15} & 4.21 * 10^{-27} & 4.07 * 10^{-06}
\end{pmatrix}
\end{array}
\begin{array}{c}
 \\ = 1 \\ = 1 \\ = 1 \\ = 1 \\ = 1 \\ = 1
\end{array}
$$



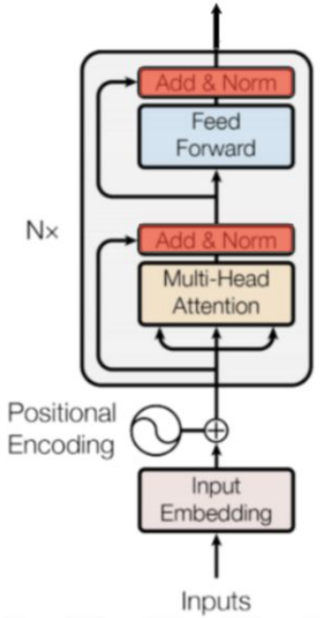
BERT – Encoder Block

- **Feed Forward** module:





- **Add, norm, dropout** layer



BERT – Training Tasks

- **Novel method 1:** Masked Language model
  - 15% input words *masked.*
    - 80% replaced by <MASK>
      - Eg: “My dog is **<MASK>”**
    - 10% replaced with random words
      - Eg: “My dog is **hotdog”**
    - 10% left intact
      - Eg: “My dog is **hairy”**

BERT – Training

- **Novel method 1:** Masked Language model
  - Network trained not to predict all the context words, but only the masked tokens.
  - **Design Decision:**
    - Longer training time than other context predicting models(?)
      - Not really due to performance boost from attention module

BERT – Training

- **Novel method 1:** Masked Language model
  - **Design Decision :**
    - Why replace with random words?
      - Would have only learned a contextual representation of ‘<MASK>’
    - Why not change/mask words for 1.5% of the time?
      - Bias the representation towards actual observed token

BERT – Training

- **Novel method 2:** Next sentence prediction
  - Feed pair of sentences separated by ***[SEP]***
    - ***50%** times the following sentence*
    - ***50%** times random 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

BERT – Training

- **Novel method 2:** Next sentence prediction
  - WHY?
    - *To make representation versatile (eg, sentence level tasks)*
    - *Shows it helps*

BERT – Corpus

- BookCorpus - 800M words
- English Wikipedia - 2,500M words – The ENTIRE WikiPedia!!
- Tokens tokenized using 37,000 WordPiece tokens

BERT – Procedure

- Random samples in batches of two (50% of the time adjacent to each other)
- …such that combined length ≤ 512 tokens
- 15% masked from each sequence

BERT – Procedure

- Batch size = 256 sequences
- Each sequence of length 512 tokens
- Hence $256 \times 512 = 128{,}000$ tokens per batch
- NOTE: sequences can have more than 2 sentences.

BERT – Loss

- The loss was calculated as:

$$\mu_{masked\_LM} + \mu_{sent\_pred\_likelihood}$$

BERT – Experiments

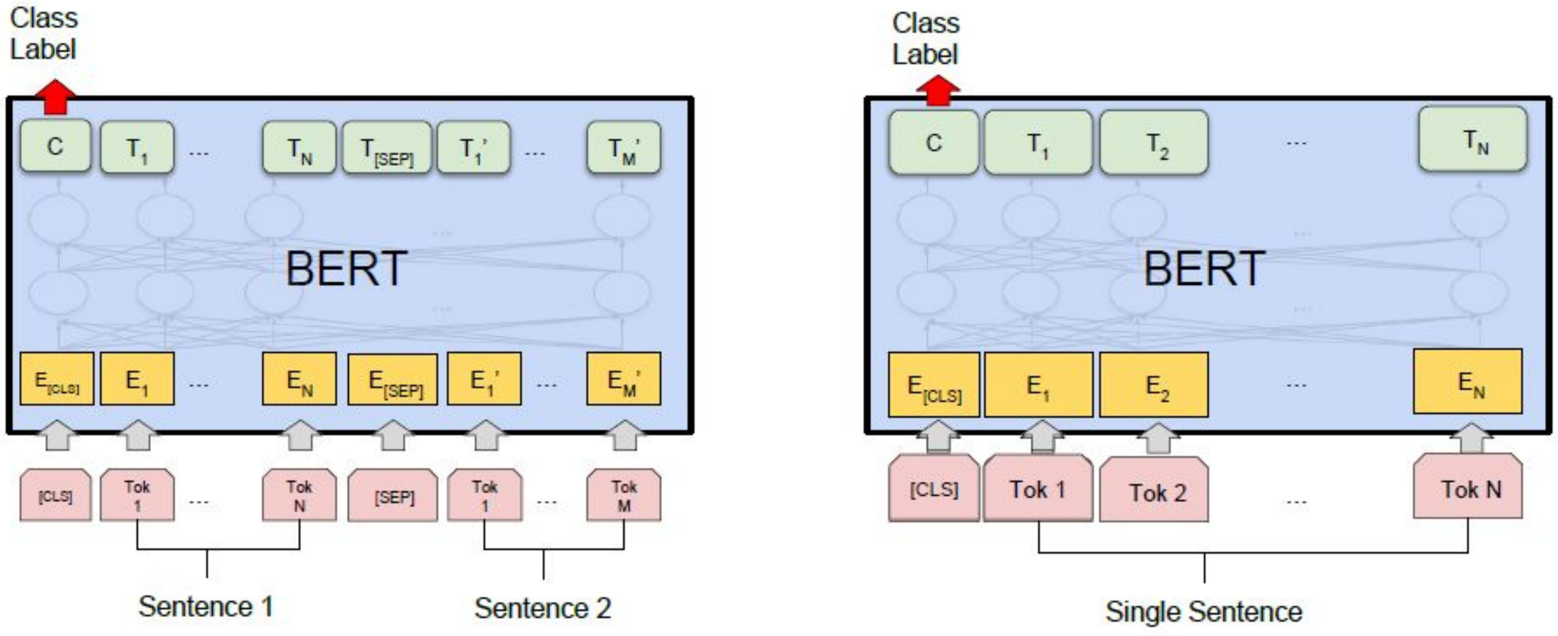
- Hyperparameters:
  - **Batch-size:** 16,32
  - **Learning rate(ADAM):** 5e-5, 3e-5, 2e-5
  - **Number of epochs:** 3,4
  - GELU activation

BERT – Experiments (GLUE)

- Evaluation tasks:
  - MNLI
  - QQP
  - QNLI
  - MRPC
  - WNLI
  - SST-2
  - CoLA
  - STS-B
  - RTE

BERT – Experiments (GLUE)

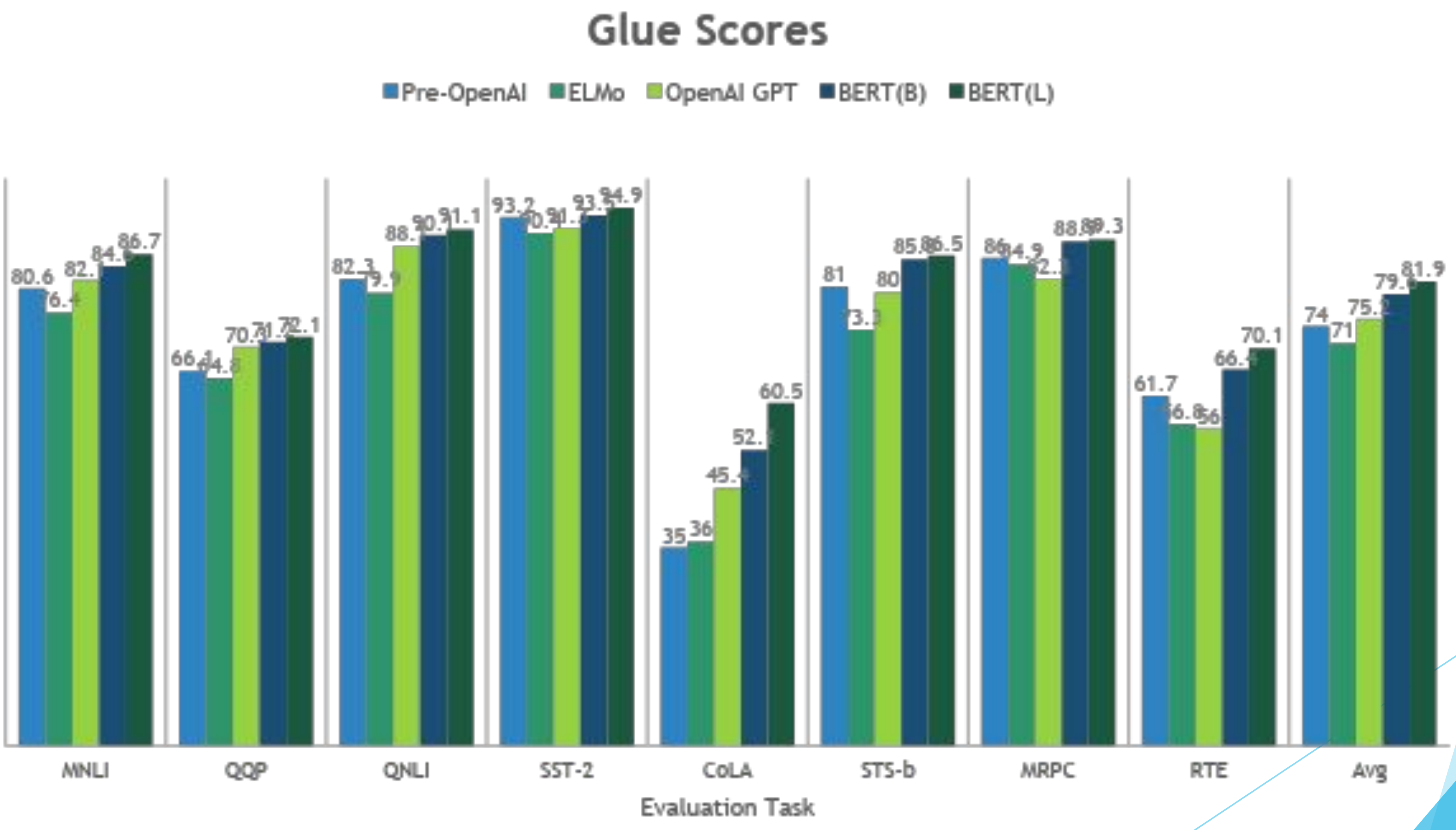
- Model used:



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

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

BERT – Results(GLUE)



BERT – Experiments (SQuAD)

- Input Question:

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

- Input Paragraph:

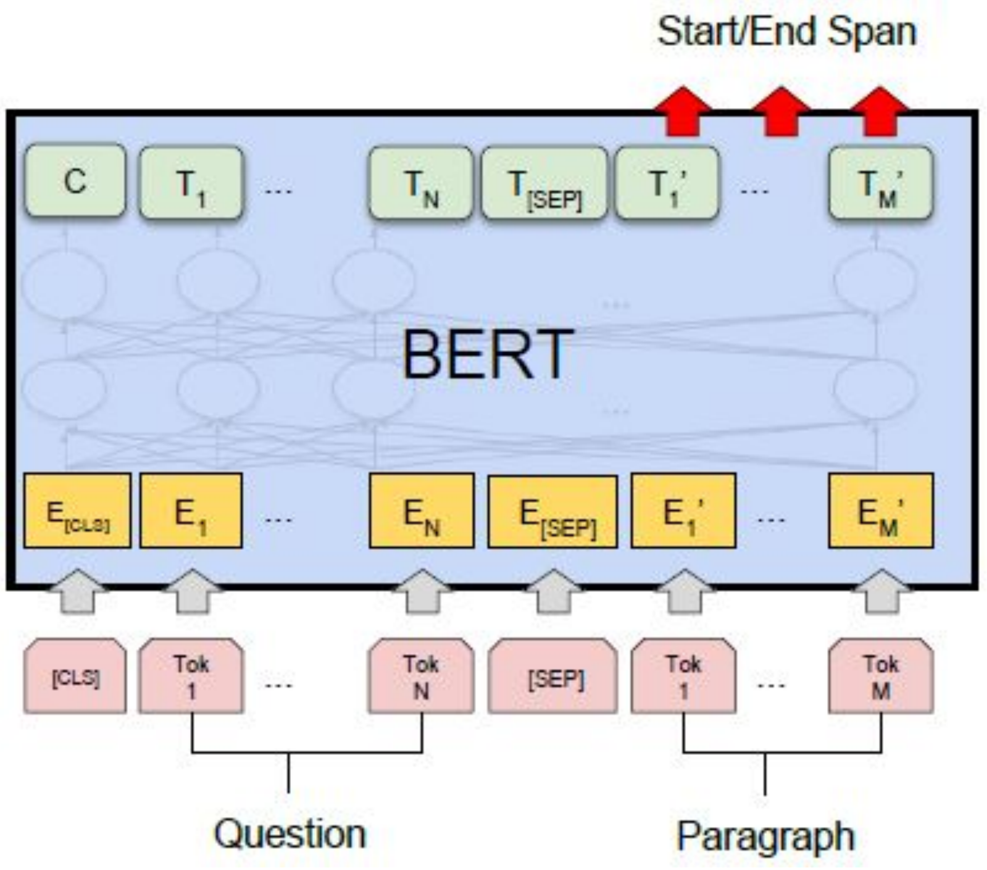
  ... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. ...

- Output Answer:

  within a cloud

BERT – Experiments (SQuAD)

- Model used:



BERT – Experiments (SQuAD)

- S : start vector of answer span
- T_i: Output at step 'i'

$$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$$

- Same for stop

BERT – Results(SQuAD)

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.

BERT – Results(Named Entity Recognition)

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**

BERT – Results(SWAG)

A girl is going across a set of monkey bars. She

(i) jumps up across the monkey bars.

(ii) struggles onto the bars to grab her head.

(iii) gets to the end and stands on a wooden plank.

(iv) jumps up and does a back flip.

BERT – Results(SWAG)

System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
$\text{BERT}_{\text{BASE}}$	81.6	-
$\text{BERT}_{\text{LARGE}}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

Table 4: SWAG Dev and Test accuracies. Test results were scored against the hidden labels by the SWAG authors. †Human performance is measure with 100 samples, as reported in the SWAG paper.

BERT – Ablation Studies

- **Pre-training tasks**
  - MLM and no NSP (effect of NSP)
  - LTR and no NSP (effect of MLM)
    - i.e, OpenAI GPT architecture

BERT – Ablation Studies

- **Pre-training tasks**
  - MLM and no NSP (effect of NSP)

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

BERT – Ablation Studies

- **Pre-training tasks**
  - Effect of MLM

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$BERT_{BASE}$	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

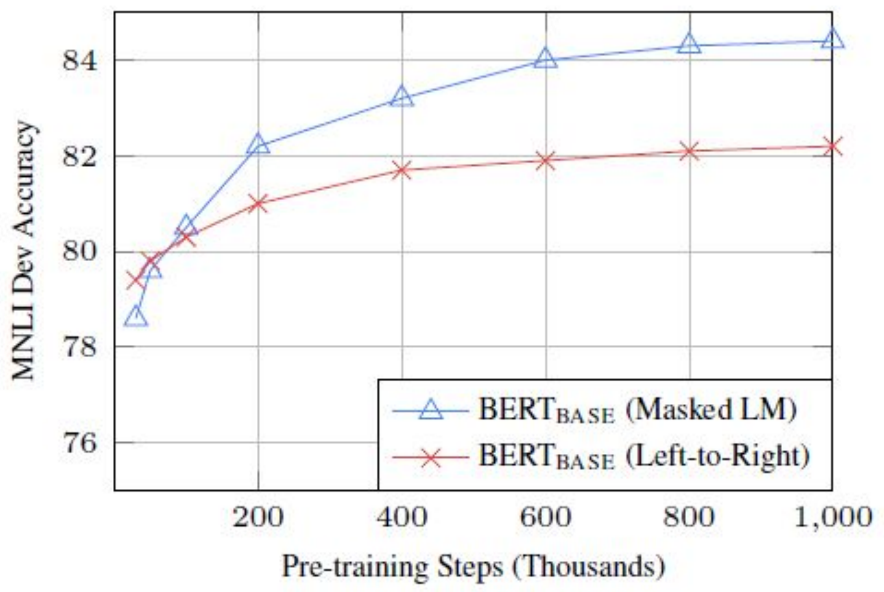
BERT – Ablation Studies

- **Model Size**
  - Less data, Huge model – still works

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

BERT – Ablation Studies

- **Training Steps**
  - YES, more steps = higher performance
  - Converges slower, but performance HIGHER



BERT – Final Remarks and Salient Points

- ✔ Bi-Directional learning
- ✔ 2 Novel pre-training tasks (PROVED to be better)
- ✔ New Input representation
- ✔ Very comprehensive and FAIR comparisons/experiments

Extending BERT

- ✔ Multi-Task Learning - natural extension of BERT
- ✔ Studying what do different layers learn? (cue: ELMo)
- ✔ What if syntactic information added?

Thank you and have a Happy Easter!

Q and A