BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

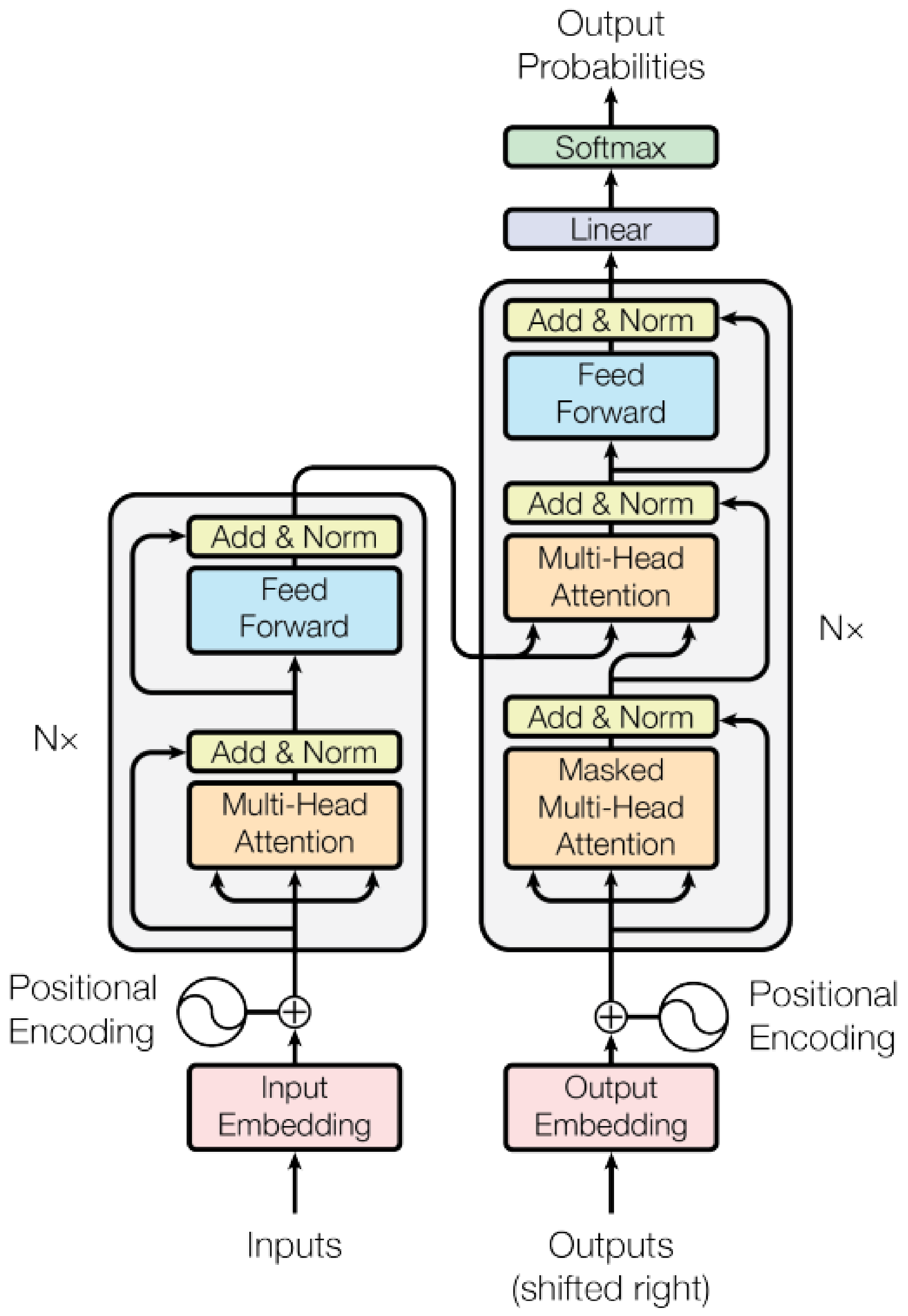
with a quick introduction to transformers

Mateusz Pach

December 14, 2022

Transformer

"Attention Is All You Need" Vaswani et al. 2017

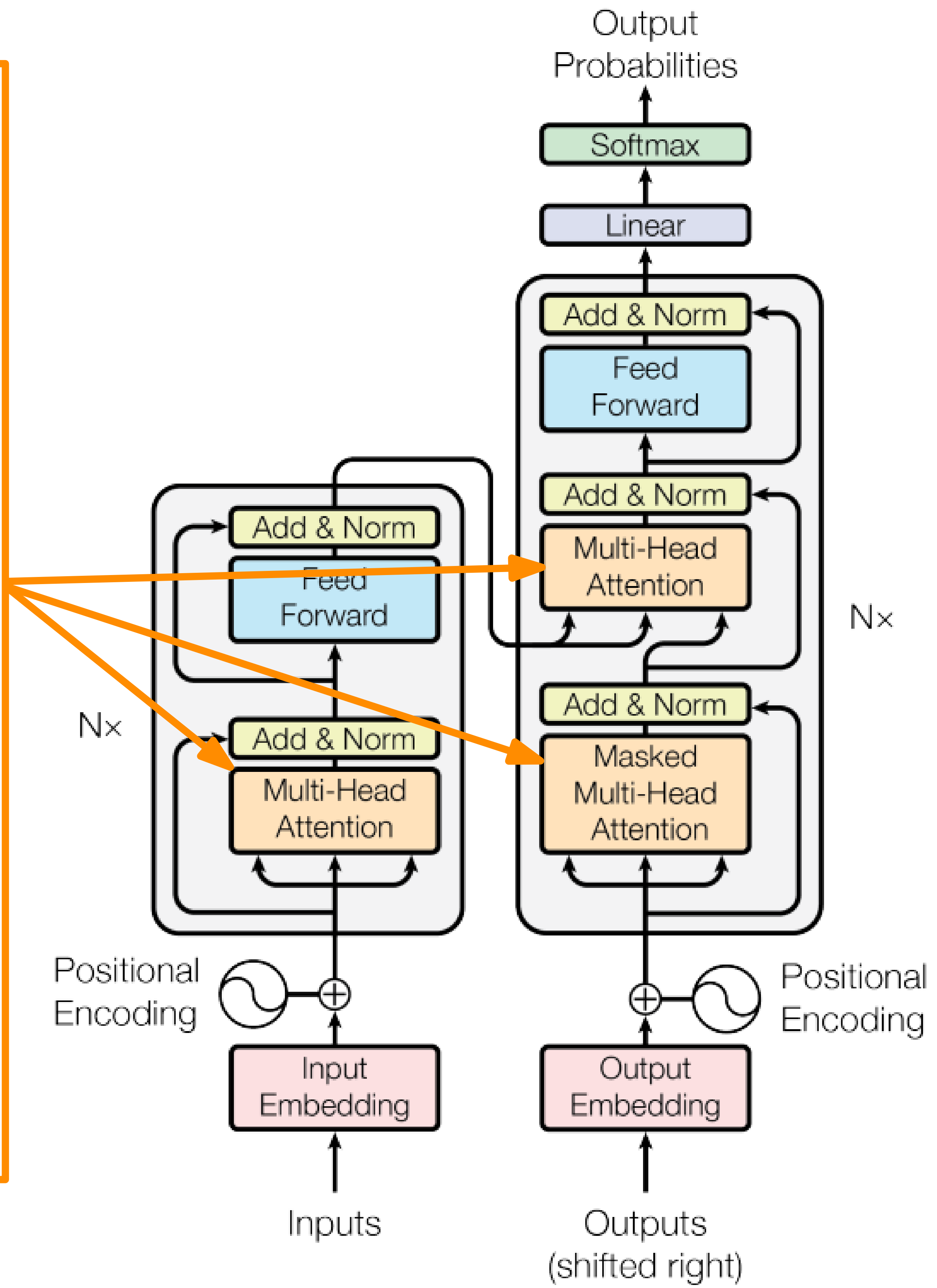


Transformer

"Attention Is All You Need" Vaswani et al. 2017

For each query $q \in Q$ assign sum of values $v \in V$ weighted by similarity of corresponding keys $k \in K$

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



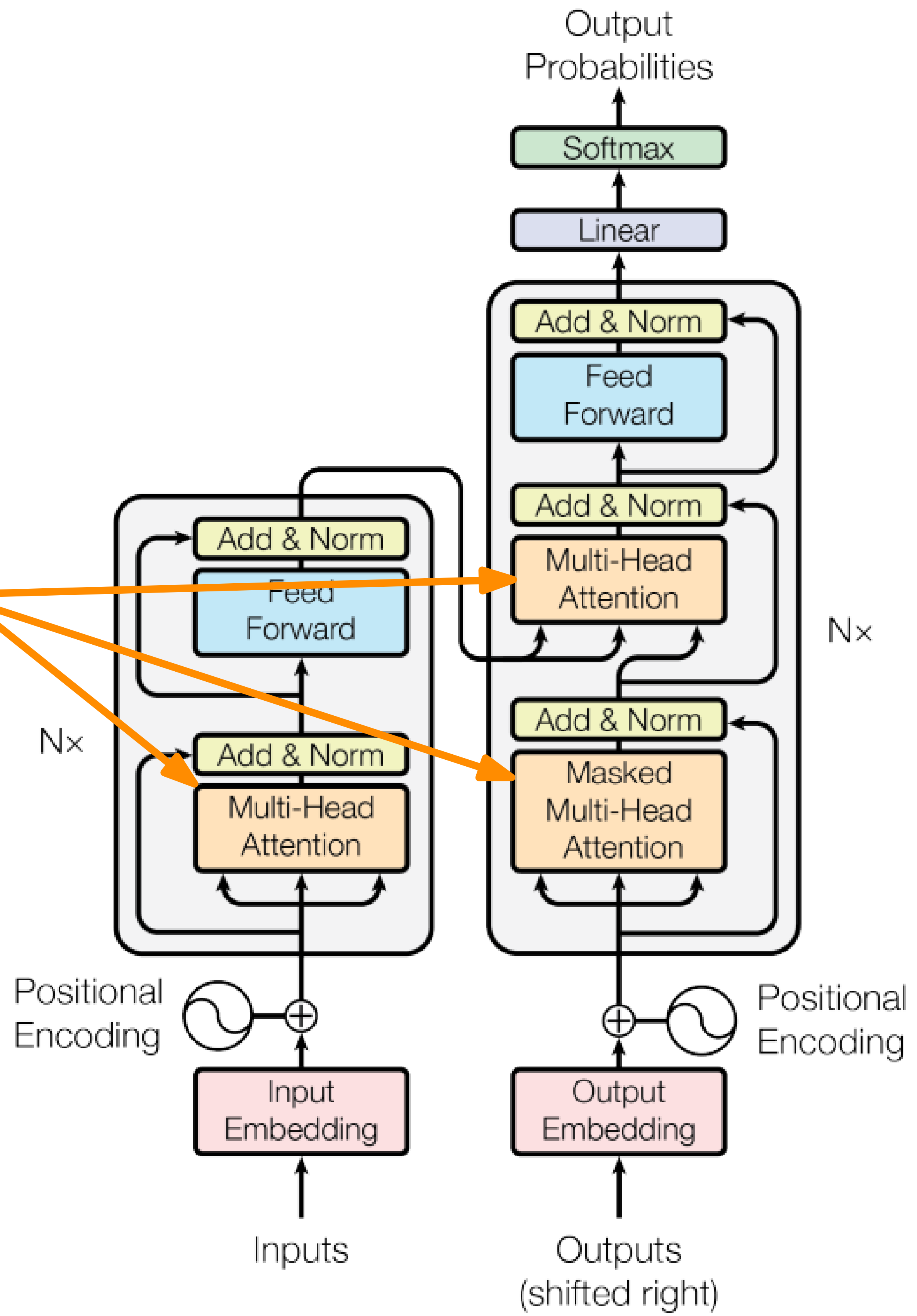
Transformer

"Attention Is All You Need" Vaswani et al. 2017

For each query $q \in Q$ assign sum of values $v \in V$ weighted by similarity of corresponding keys $k \in K$

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

Q	K	V	Attention(Q, K, V)
cat	pies	pies	$0.10 \times$ pies $+ 0.01 \times$ jajko $+ 0.89 \times$ kot
dog	jajko	jajko	$0.89 \times$ pies $+ 0.01 \times$ jajko $+ 0.10 \times$ kot
egg	kot	kot	$0.01 \times$ pies $+ 0.98 \times$ jajko $+ 0.01 \times$ kot



Transformer

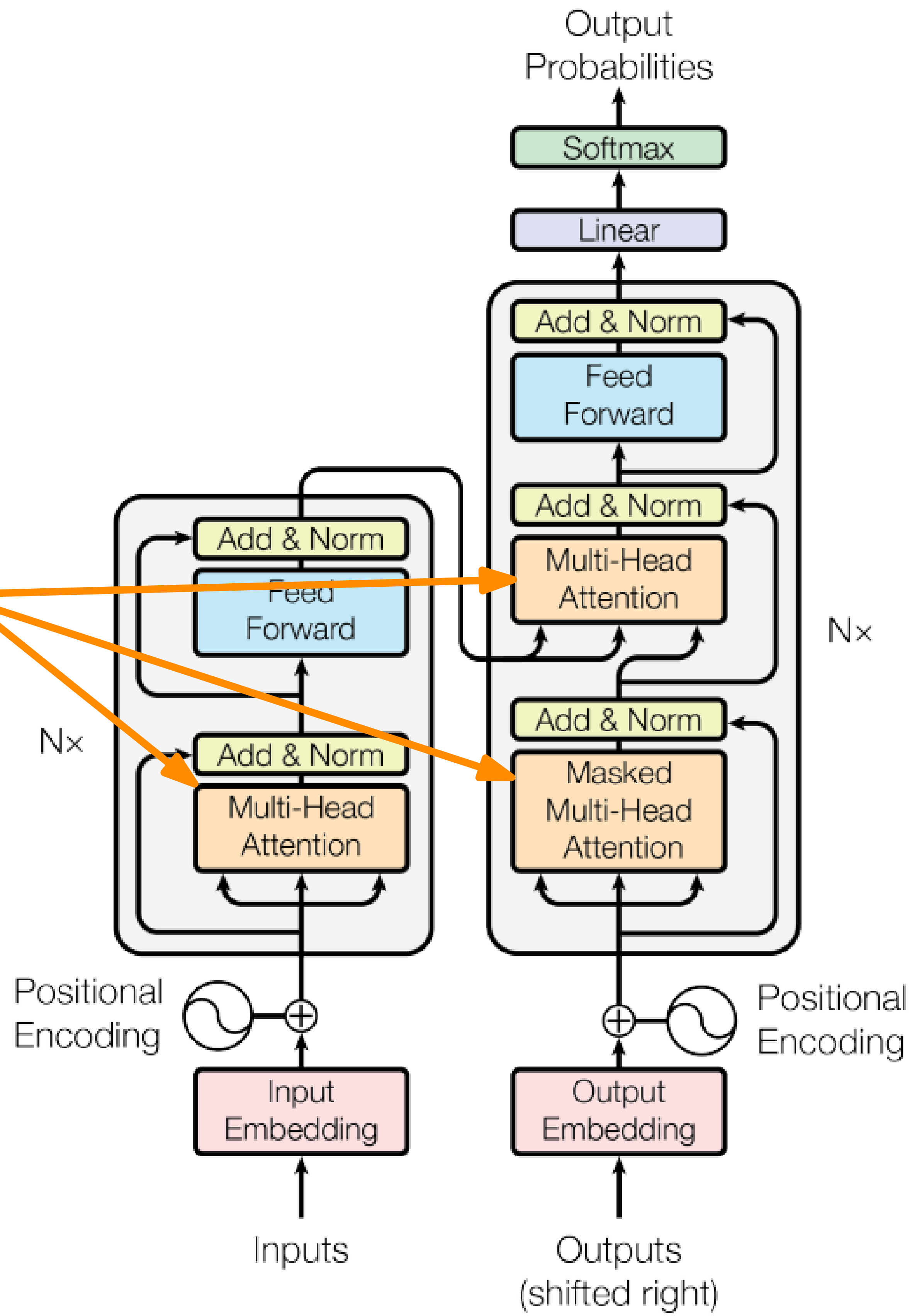
"Attention Is All You Need" Vaswani et al. 2017

For each query $q \in Q$ assign sum of values $v \in V$ weighted by similarity of corresponding keys $k \in K$

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

Q	K	V	Attention(Q, K, V)
cat	pies	pies	$0.10 \times$ pies $+ 0.01 \times$ jajko $+ 0.89 \times$ kot
dog	jajko	jajko	$0.89 \times$ pies $+ 0.01 \times$ jajko $+ 0.10 \times$ kot
egg	kot	kot	$0.01 \times$ pies $+ 0.98 \times$ jajko $+ 0.01 \times$ kot

Q	K	V	Self Attention(Q, K, V)
to	to	to	$0.60 \times$ to $+ 0.10 \times$ jest $+ 0.30 \times$ kot
jest	jest	jest	$0.10 \times$ to $+ 0.70 \times$ jest $+ 0.20 \times$ kot
kot	kot	kot	$0.30 \times$ to $+ 0.10 \times$ jest $+ 0.60 \times$ kot



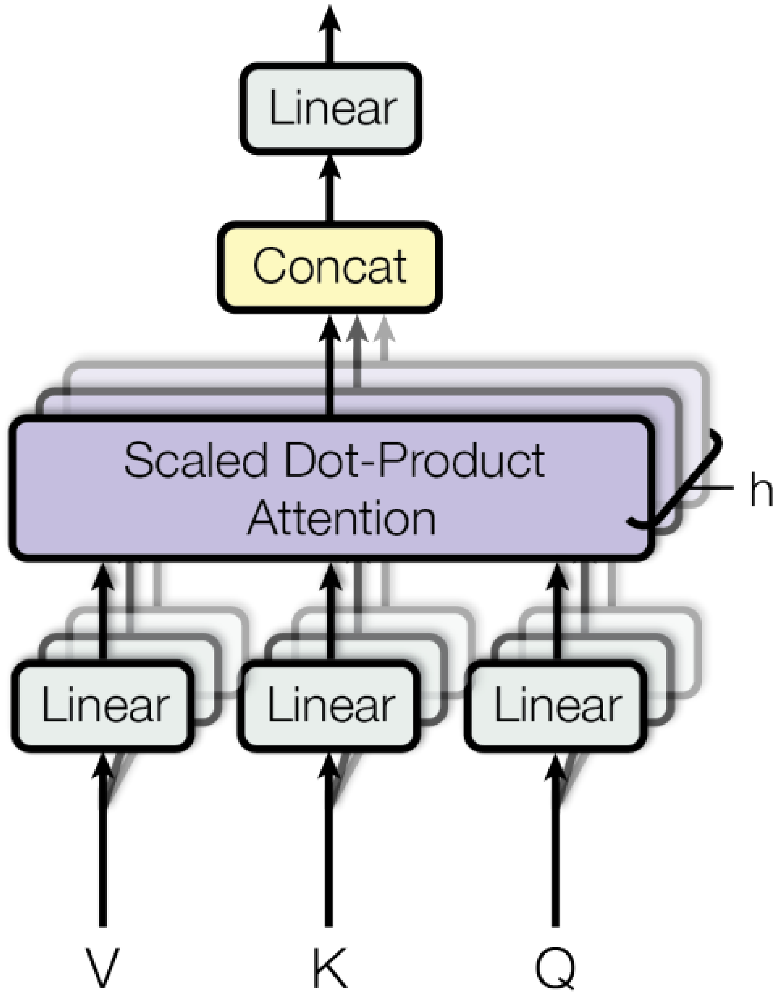
Transformer

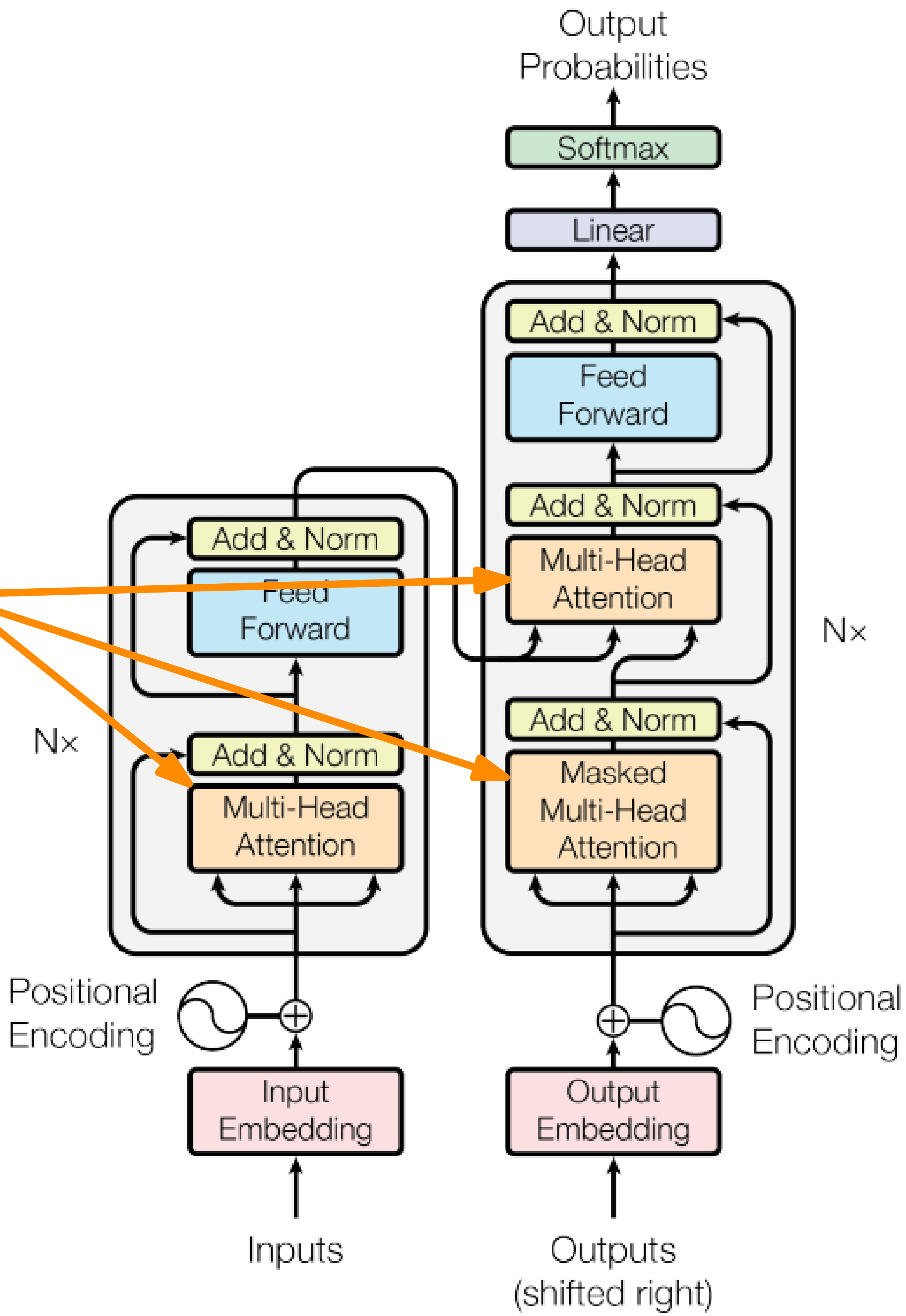
"Attention Is All You Need" Vaswani et al. 2017

Multi-Head as we project to different representations

$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_h)$$

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



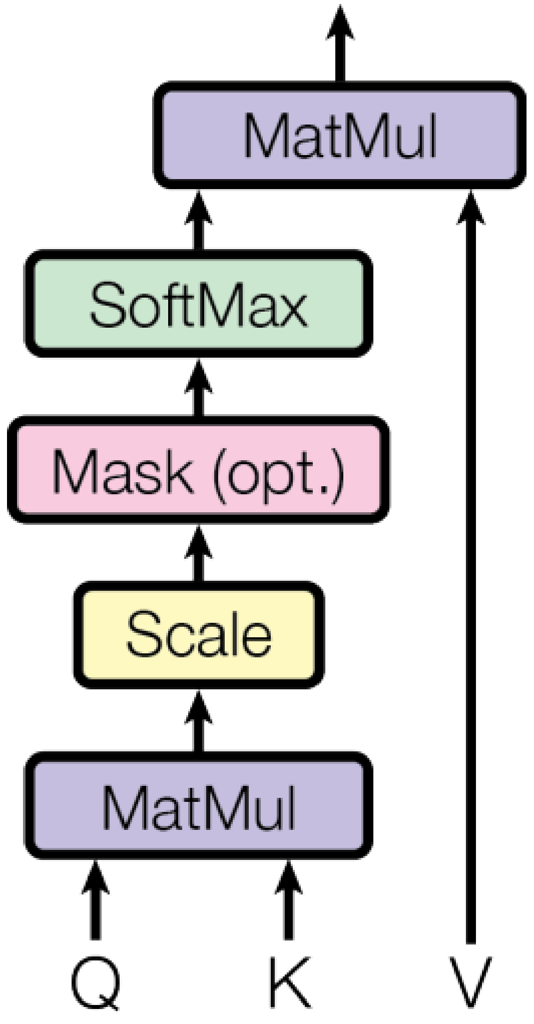


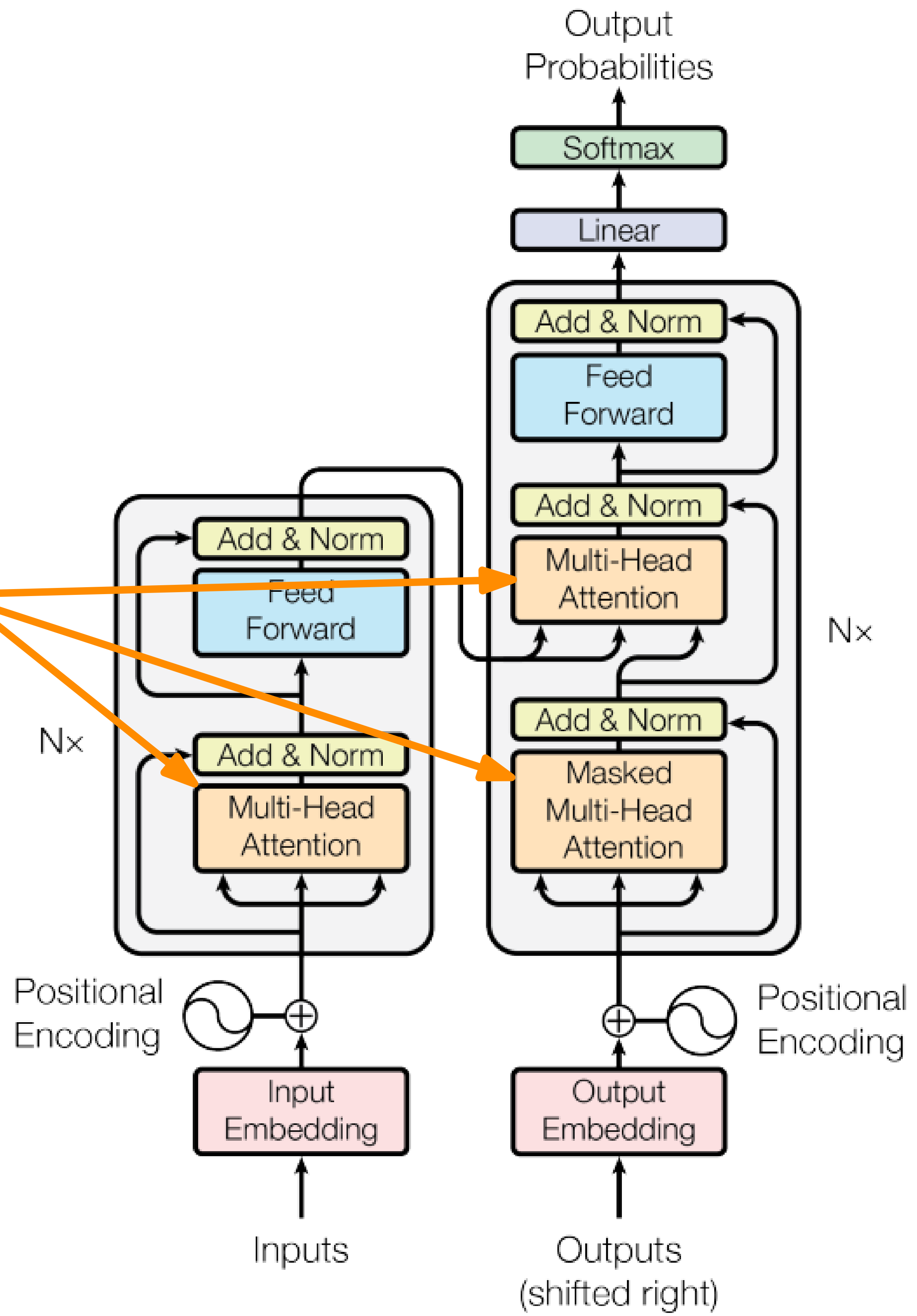
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Masked as we don't want to attend words to the right

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$



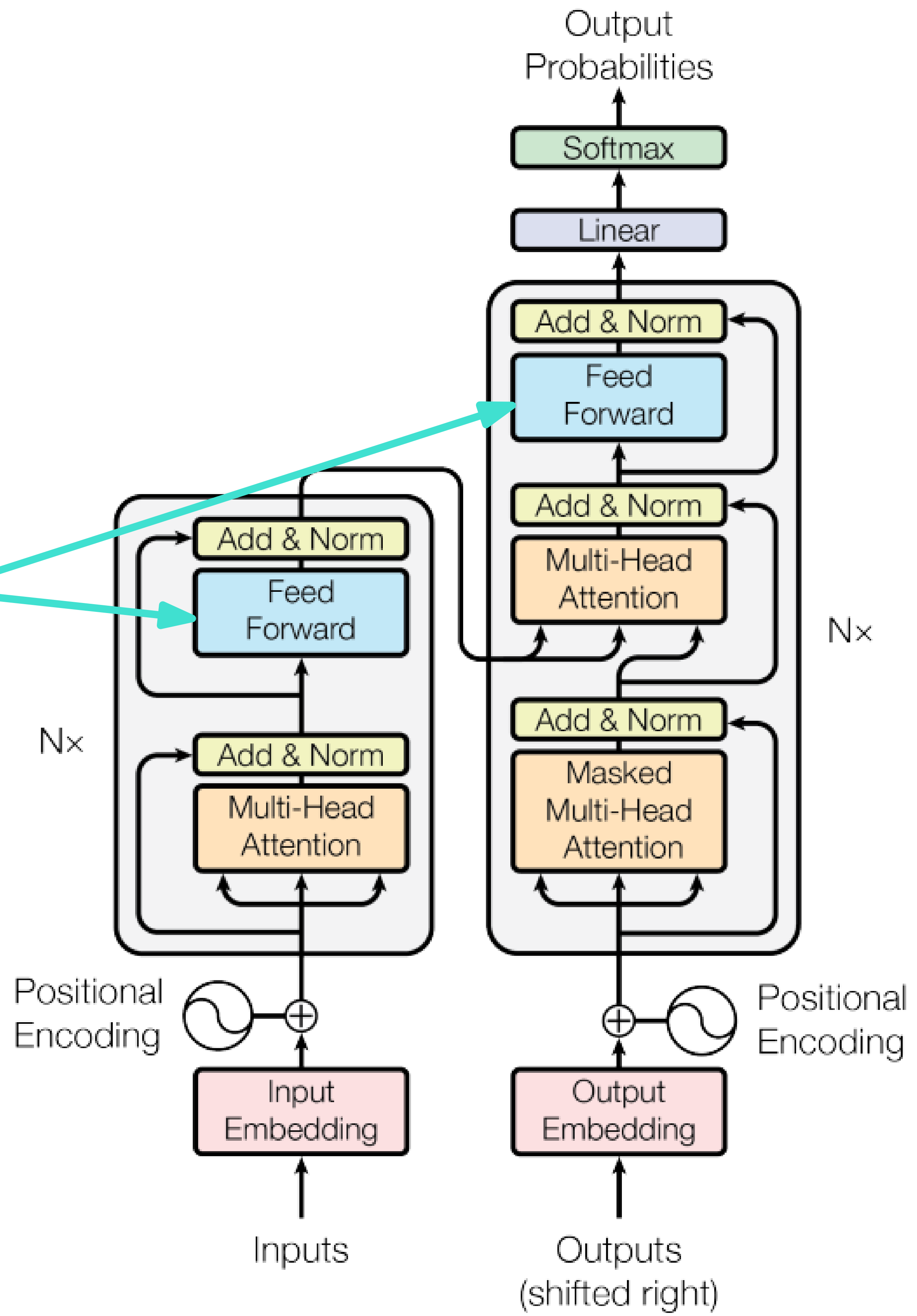


Transformer

"Attention Is All You Need" Vaswani et al. 2017

Feed Forward are two convolutions with kernel size 1

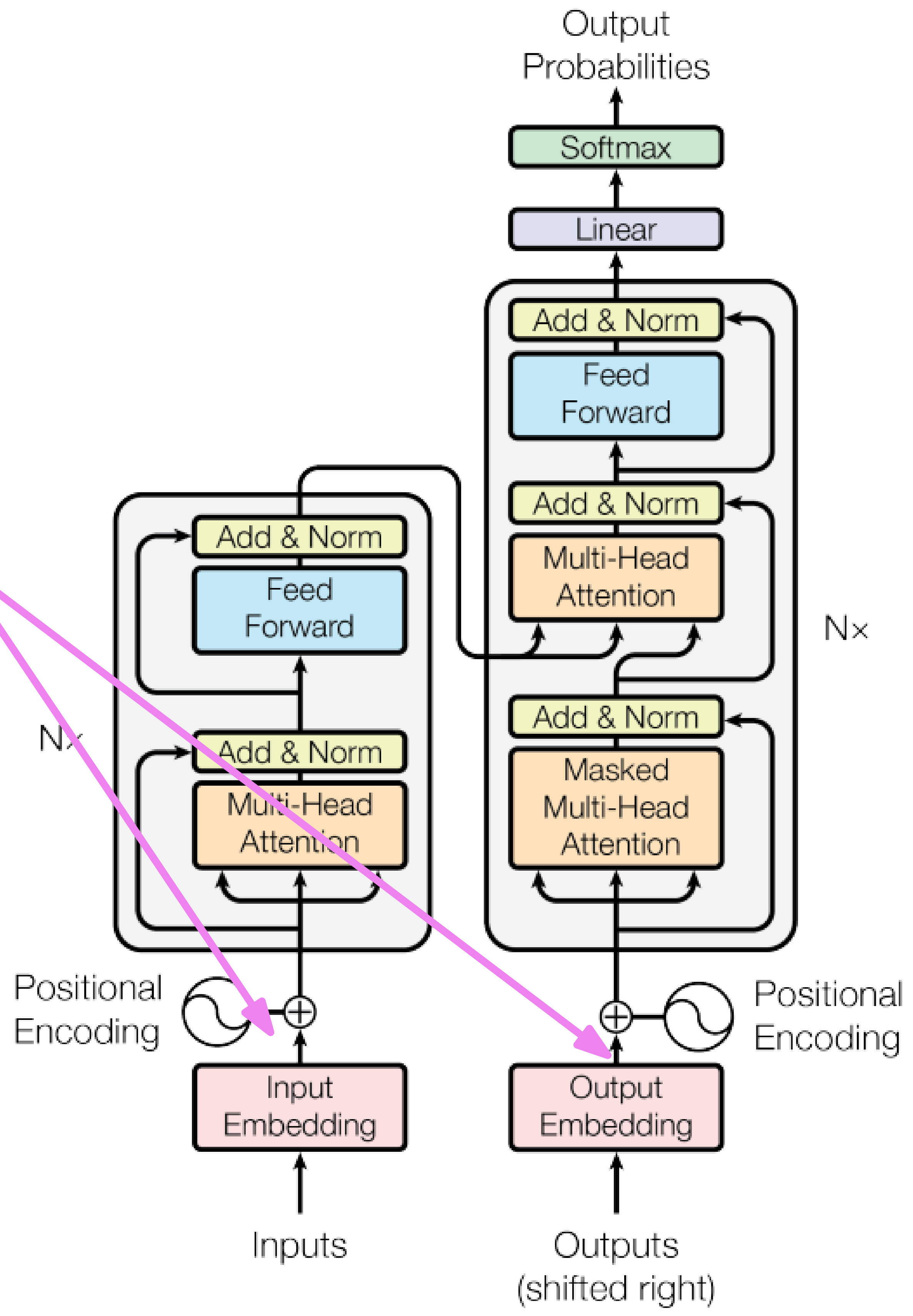
$$\mathsf{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$



Transformer

"Attention Is All You Need" Vaswani et al. 2017

Embedding transforms input tokens into vector representations



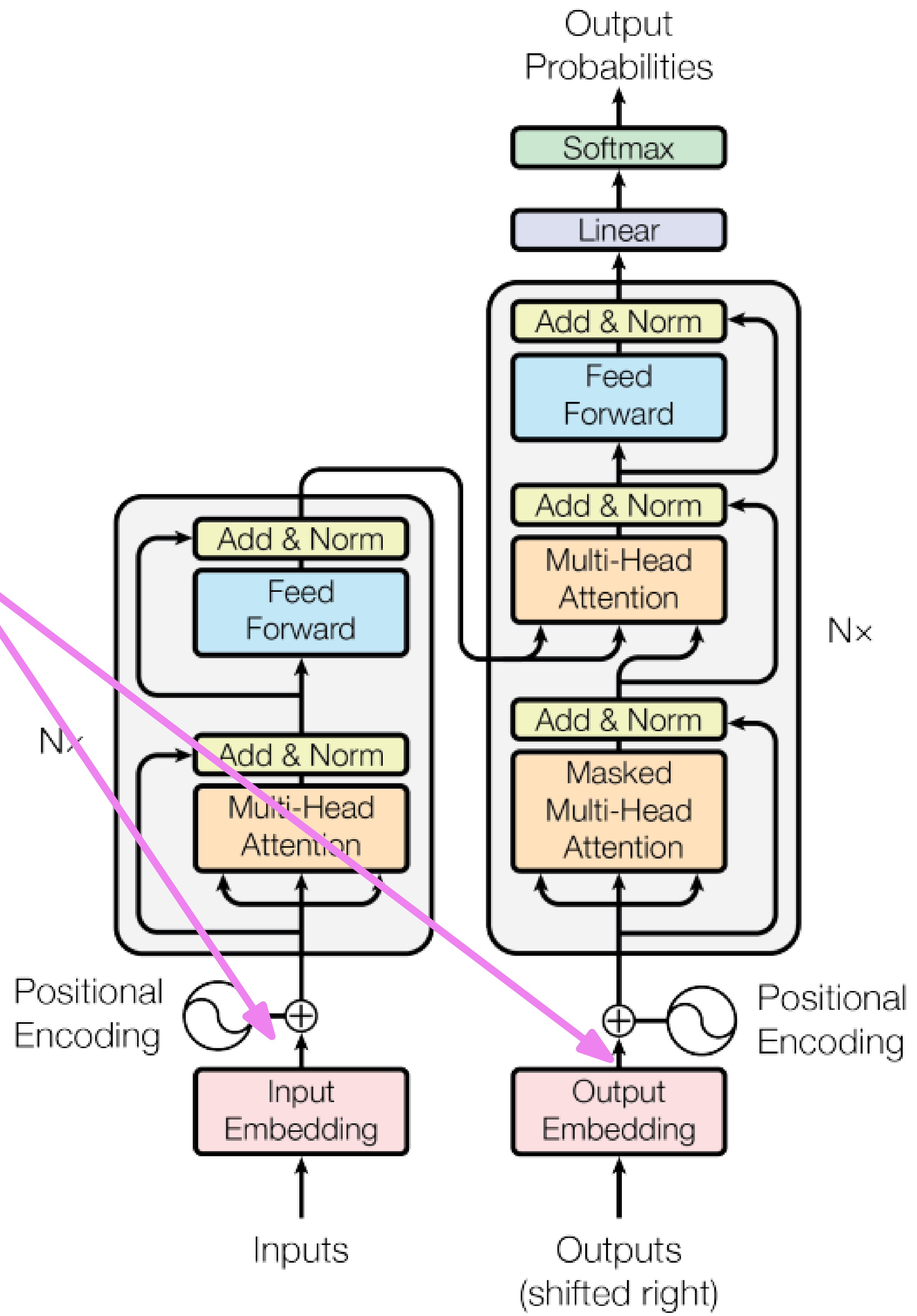
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Embedding transforms input tokens into vector representations

Positional Encoding adds information about position

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



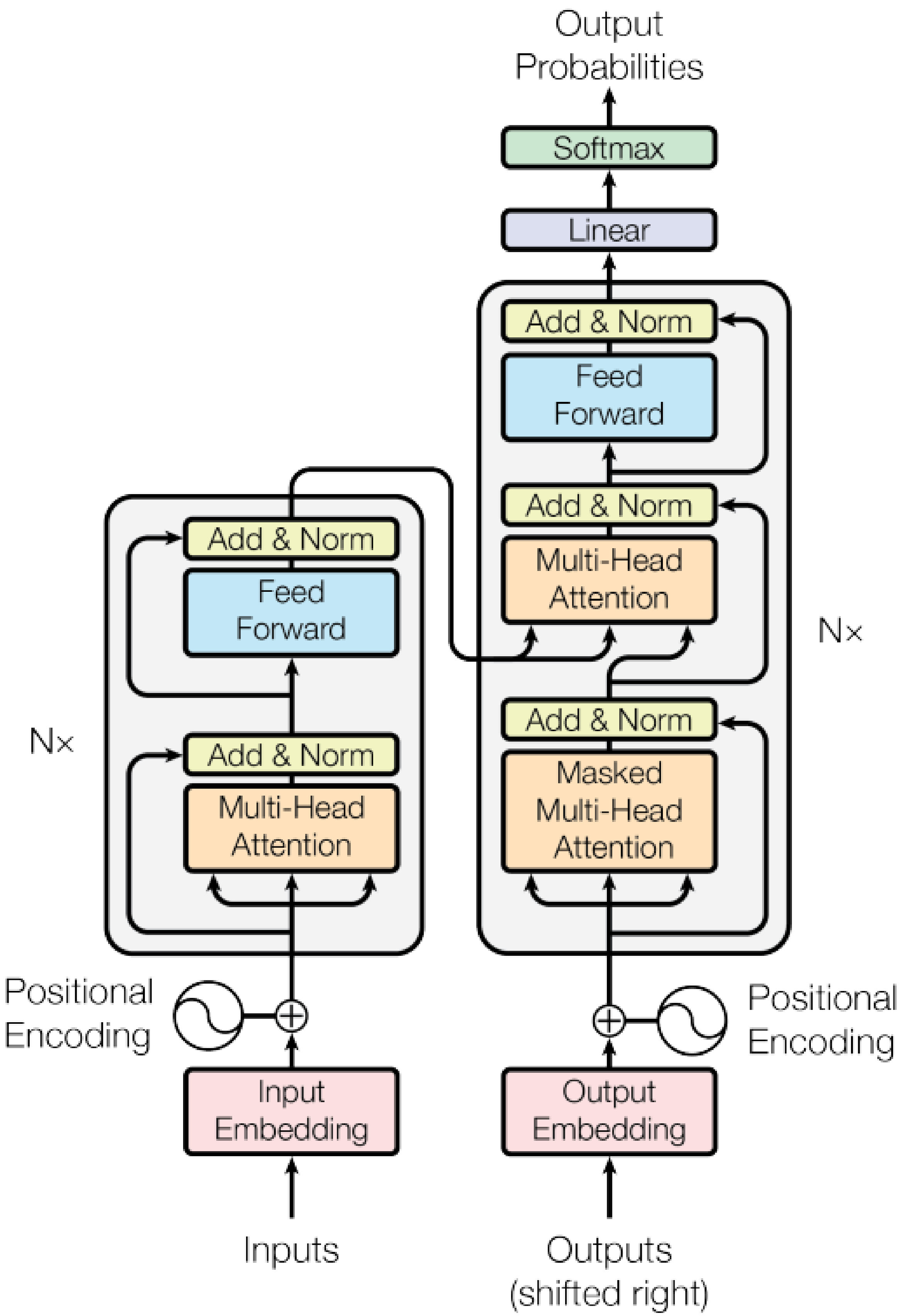
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

1. Embed k input words + encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words + encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l + 1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



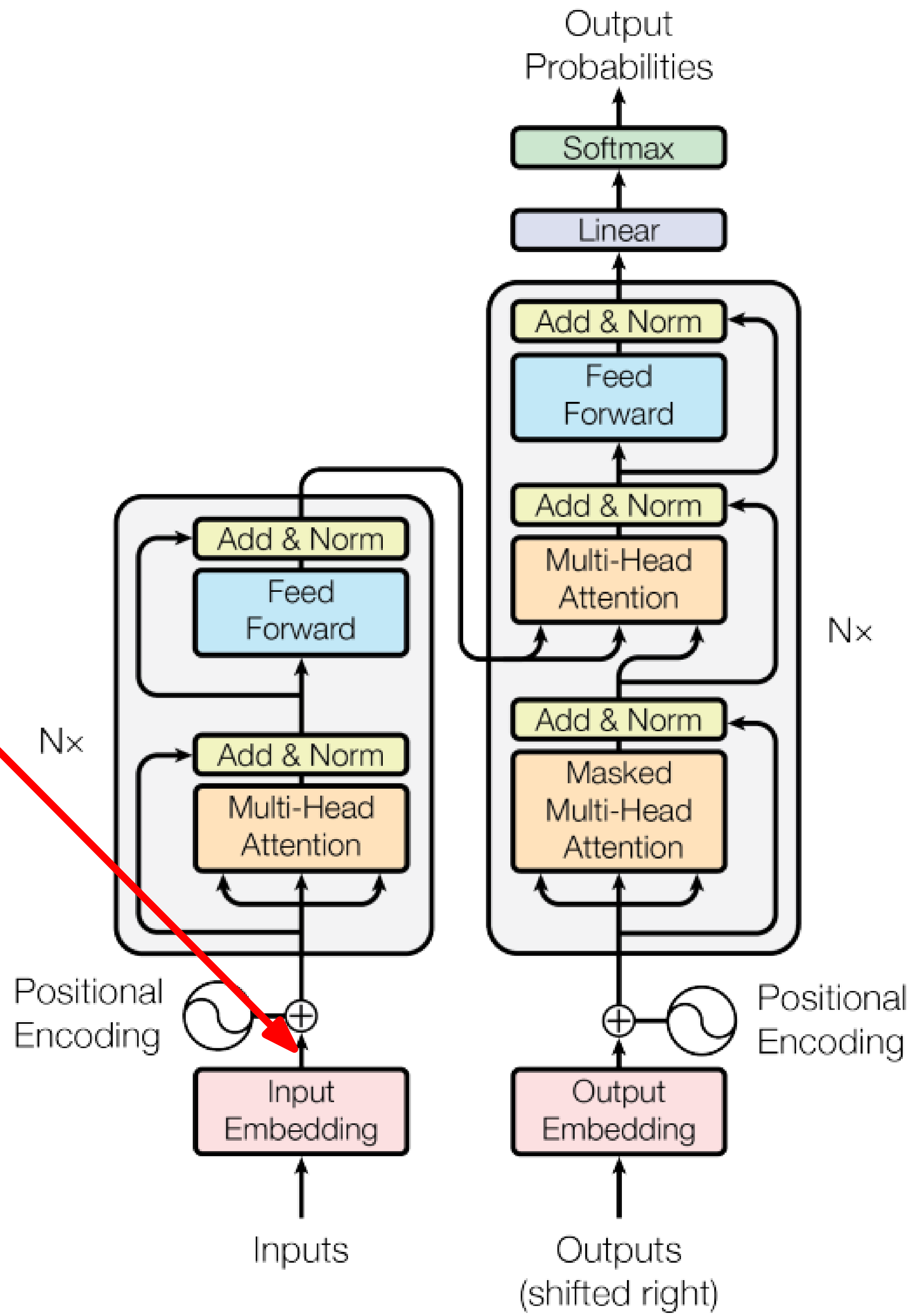
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

1. Embed k input words + encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words + encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l+1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



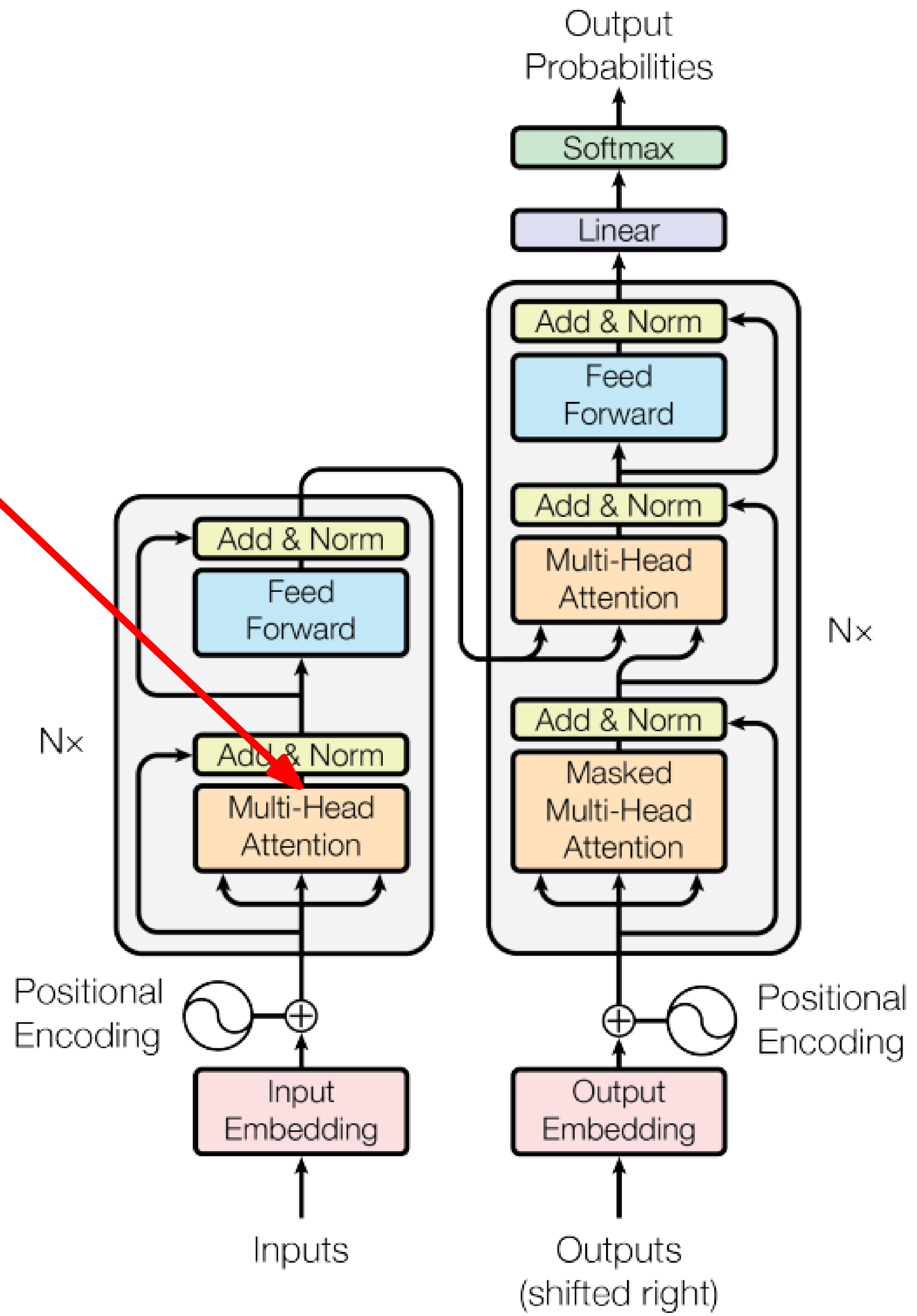
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

1. Embed k input words $+$ encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words $+$ encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l+1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



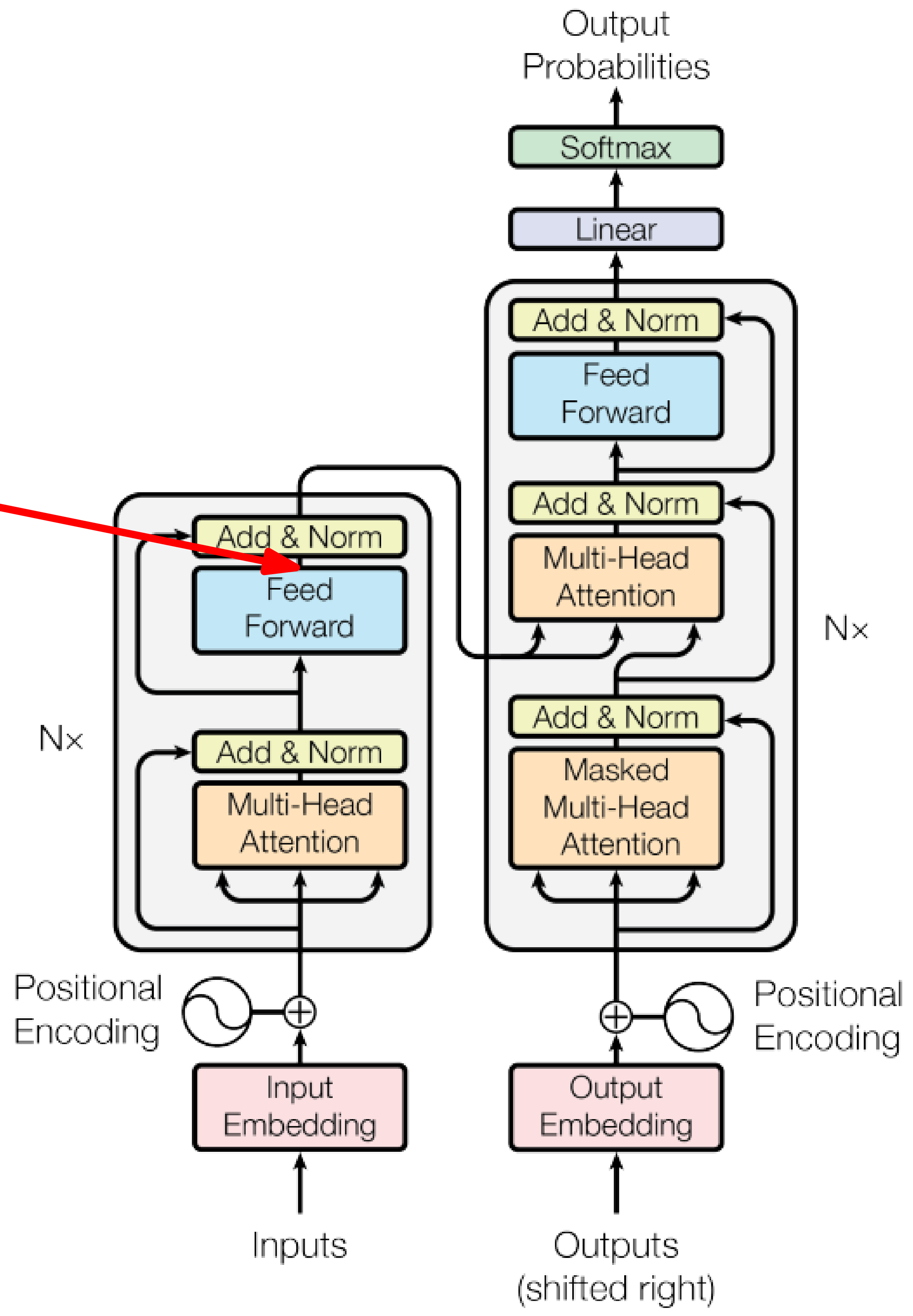
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

1. Embed k input words $+$ encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words $+$ encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l+1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



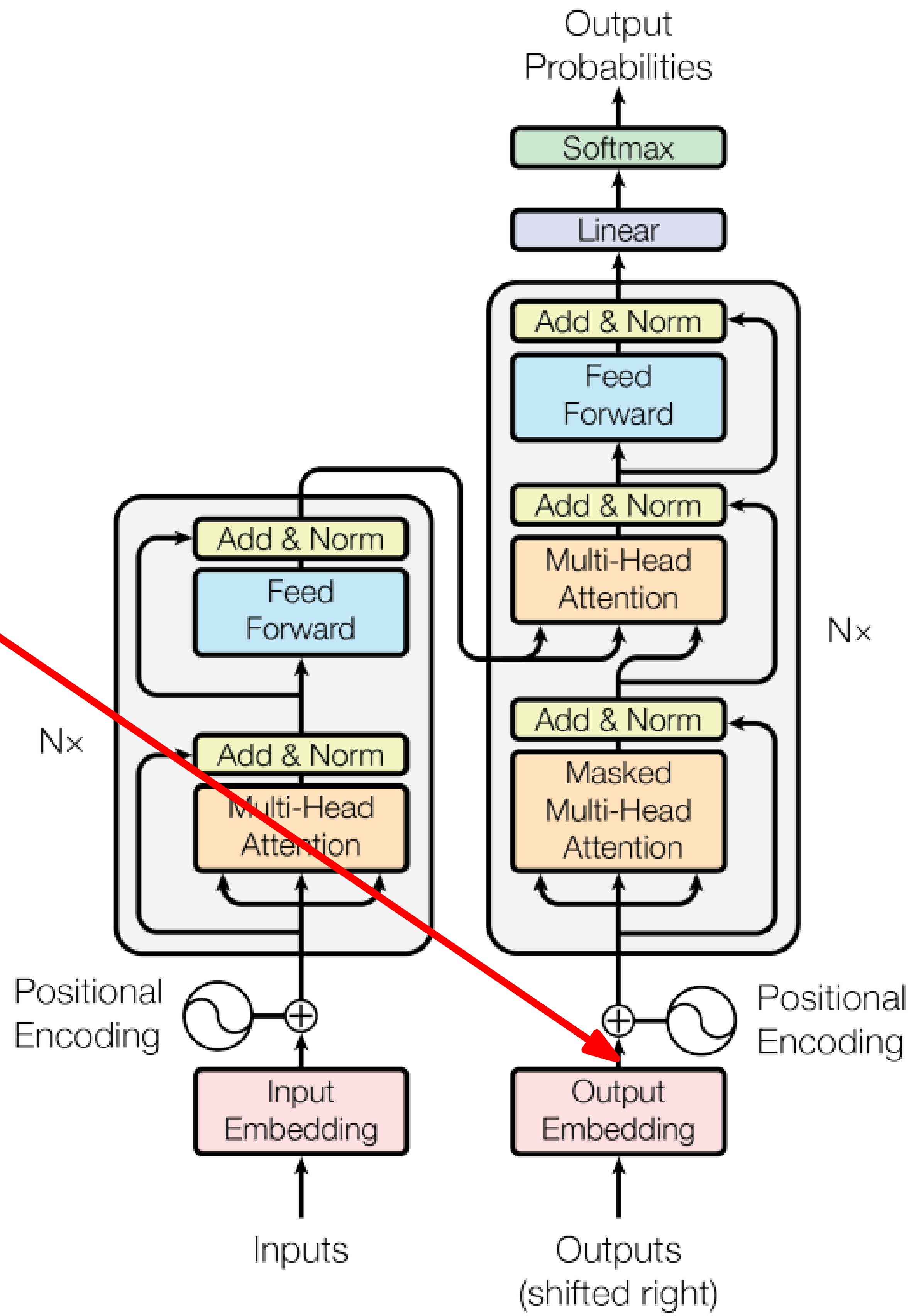
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

1. Embed k input words + encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words + encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l+1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



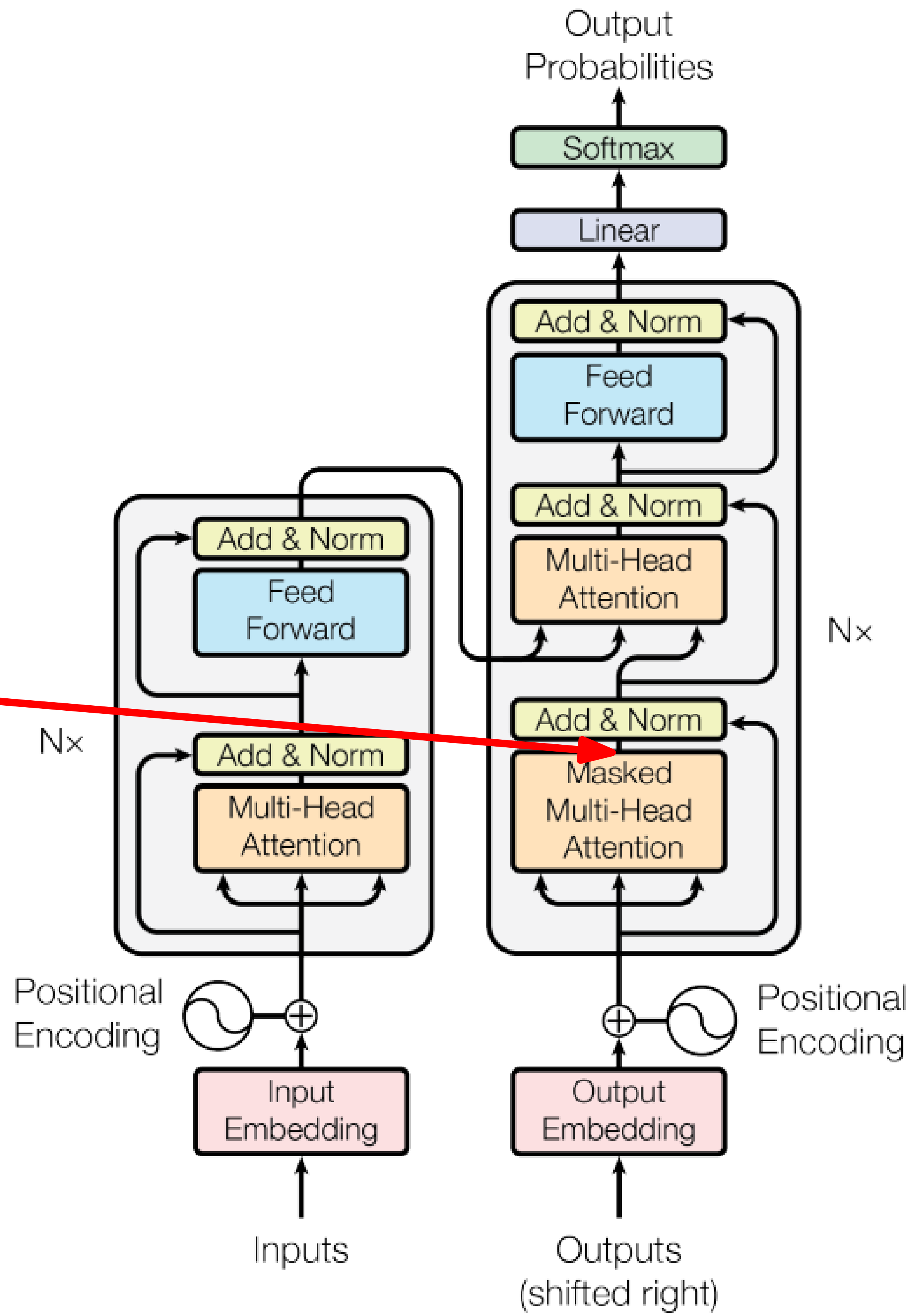
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

1. Embed k input words + encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words + encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l+1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



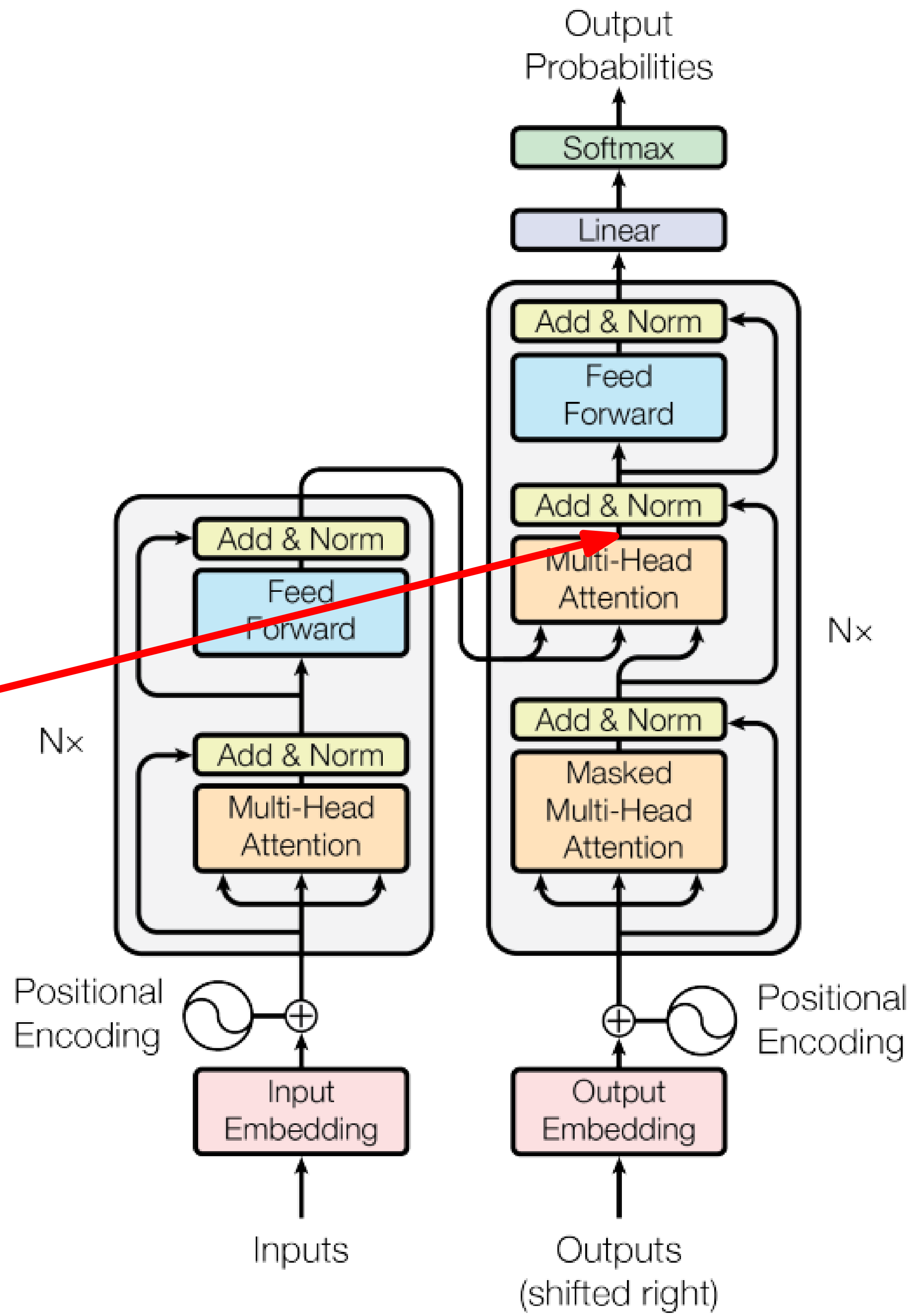
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

1. Embed k input words + encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words + encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l + 1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



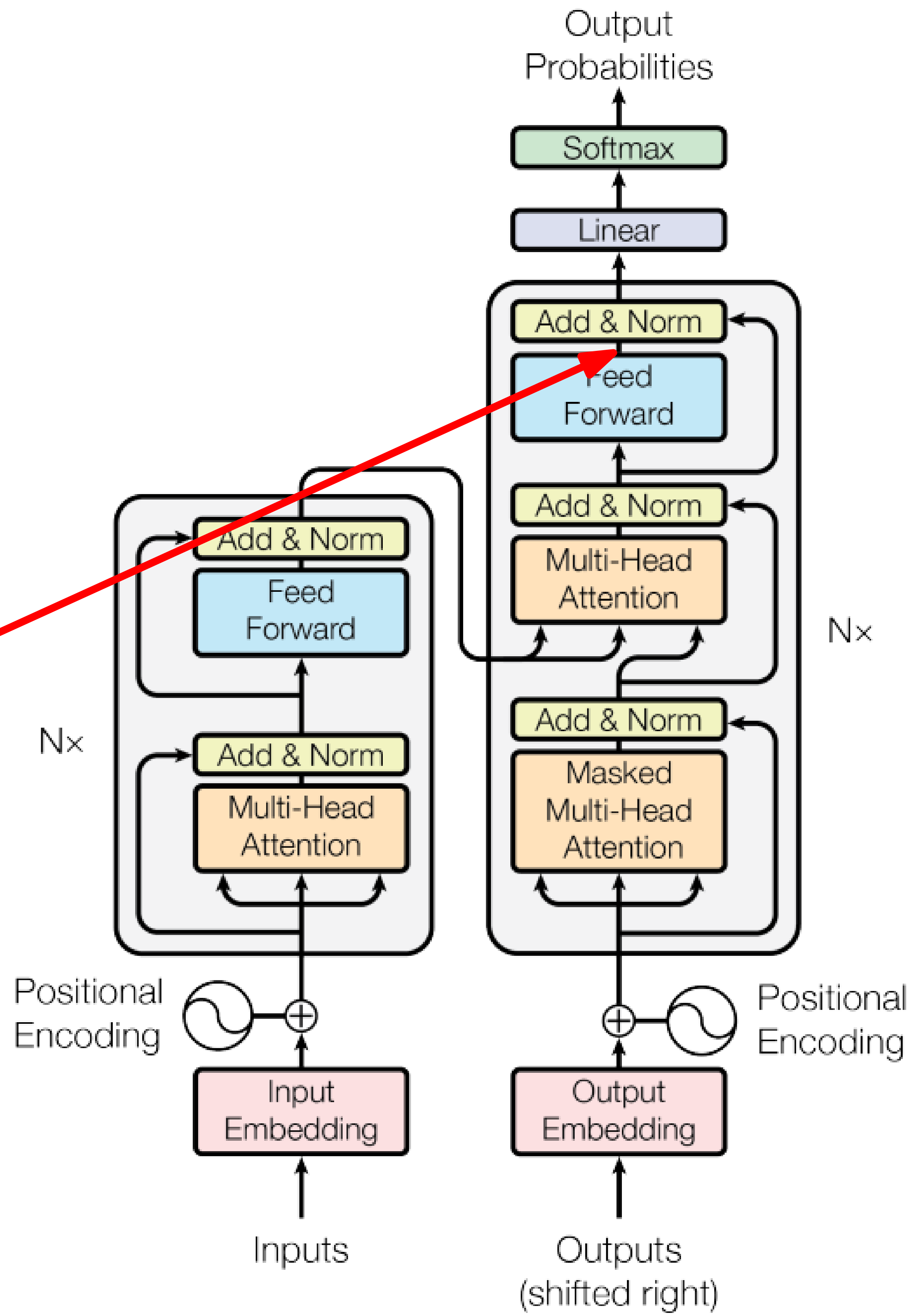
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

1. Embed k input words + encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words + encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l+1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



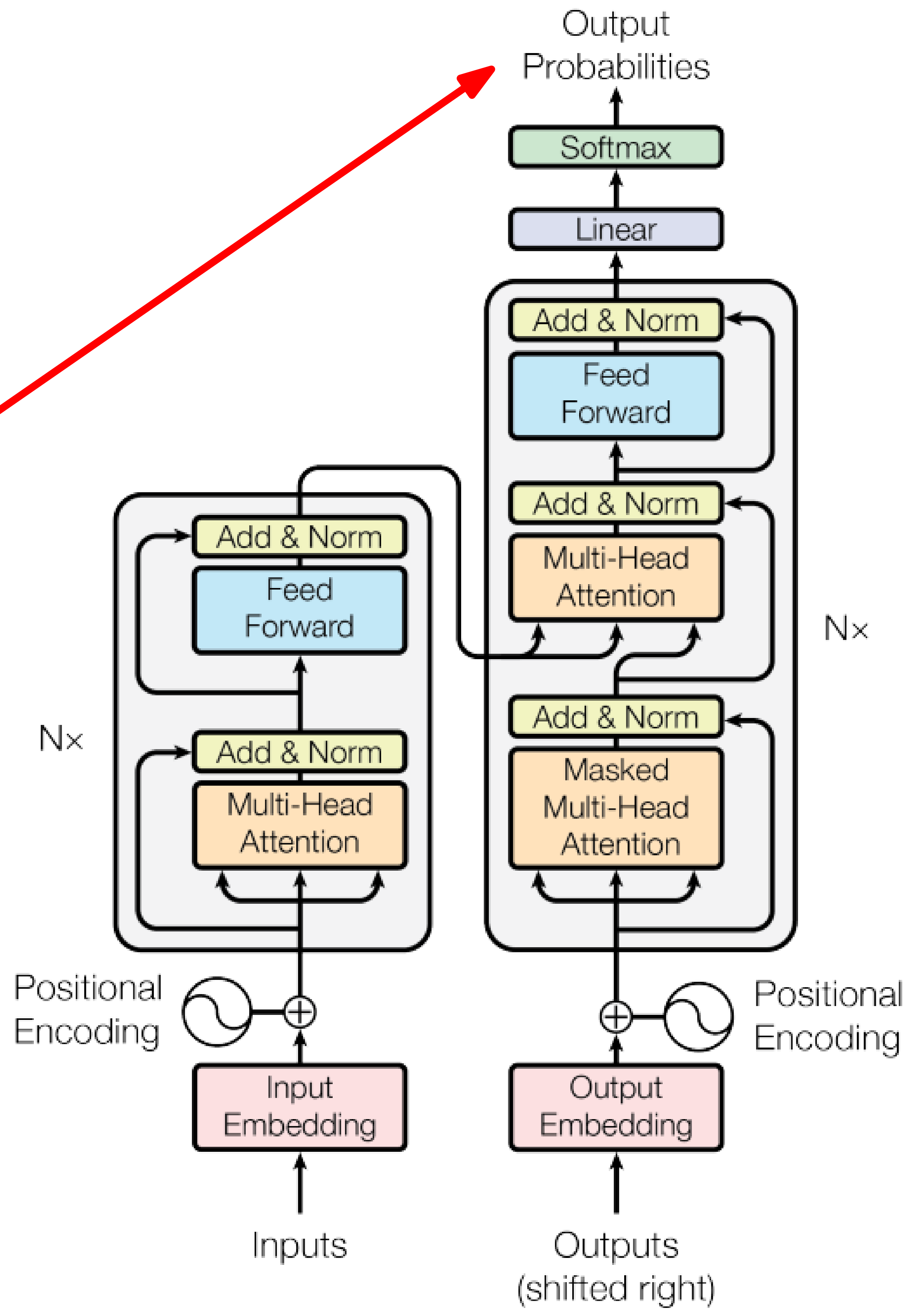
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

1. Embed k input words + encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words + encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l+1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



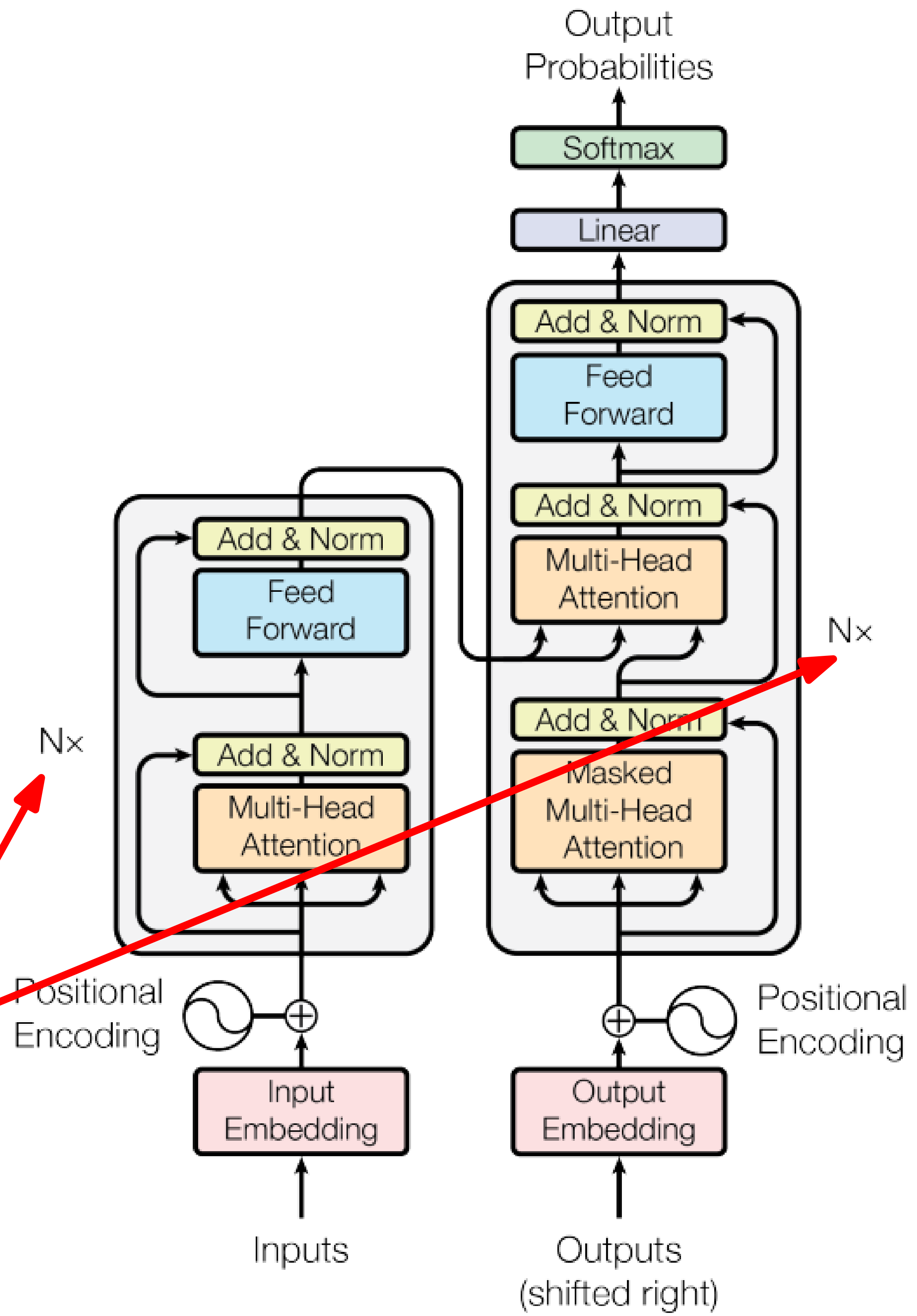
Transformer

"Attention Is All You Need" Vaswani et al. 2017

Intuition behind translation with transformers:

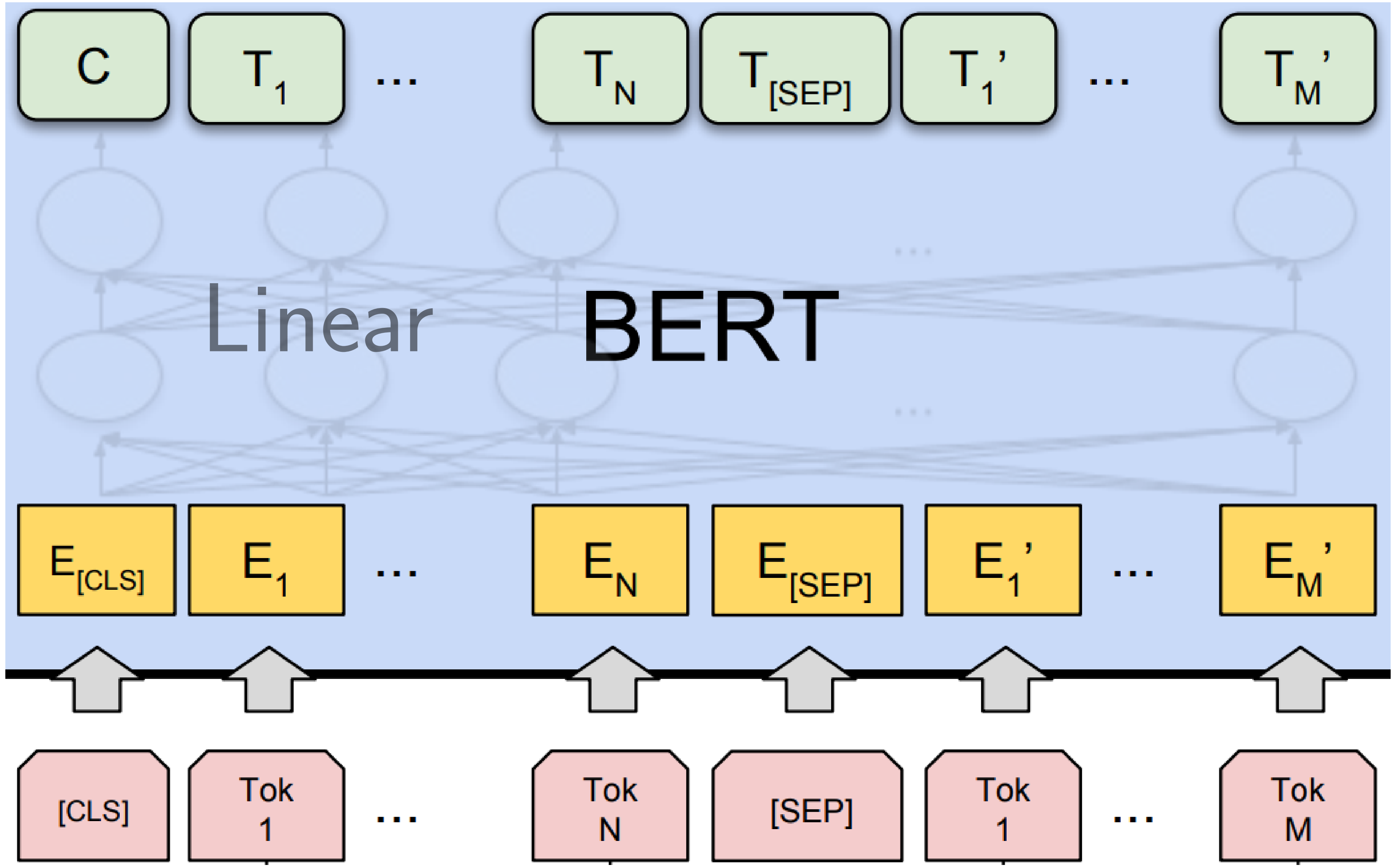
1. Embed k input words + encode position
2. Self-attend input words i.e. enrich them with context
3. Adjust for the next layer in a learned way
4. Embed already translated words + encode position
5. Self-attend l translated words i.e. enrich them with context (from the left)
6. Attend words to translate from translated words i.e enrich with original context
7. Adjust for the next layer in a learned way
8. Output $l + 1$ translated words

Note that steps 2-3 and 6-7 are repeated N times to boost the effect



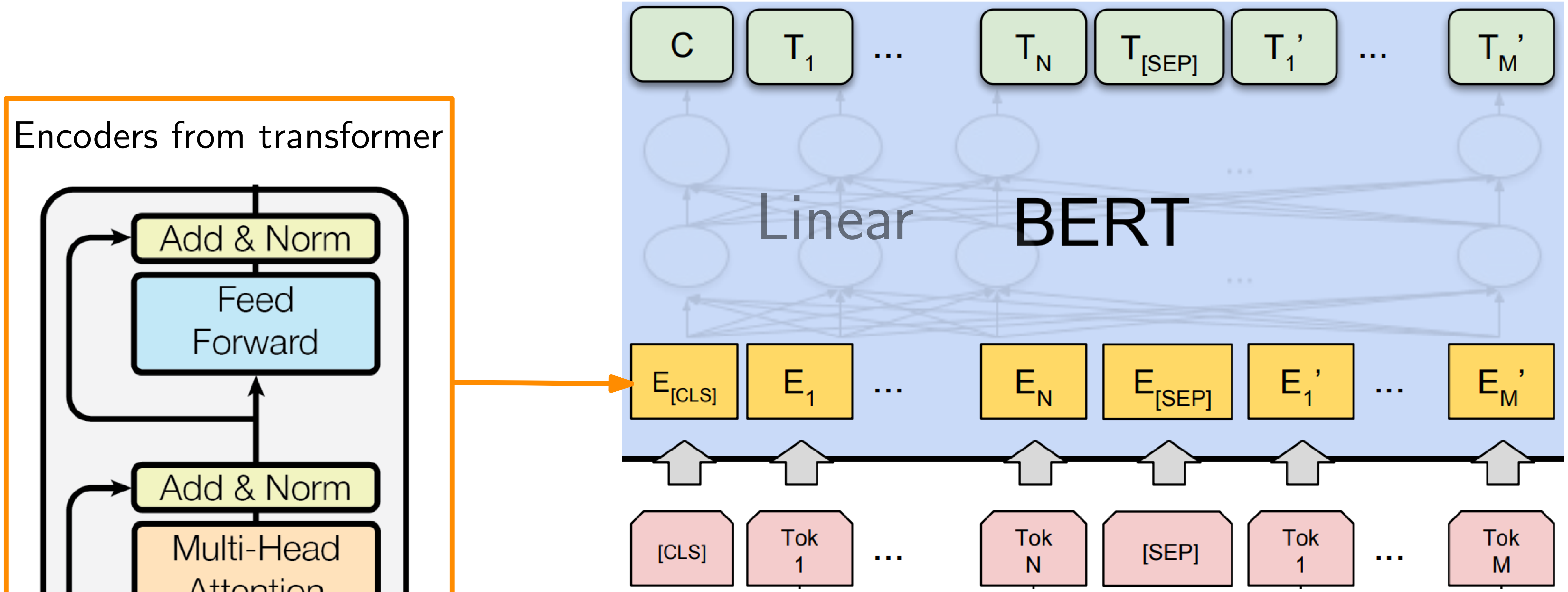
BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018



BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018



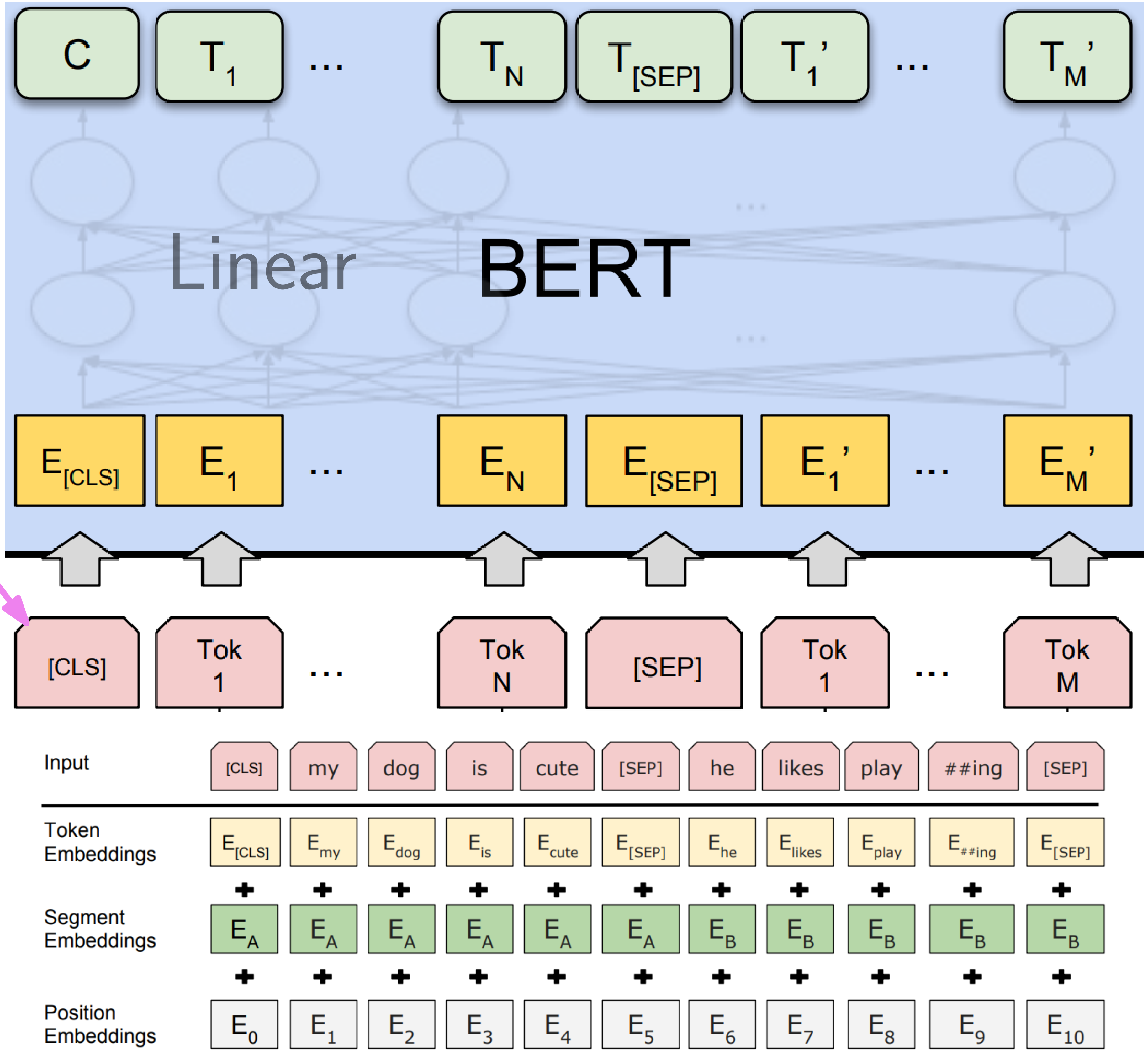
Bidirectional
Encoder
Representations from
Transformers

BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018

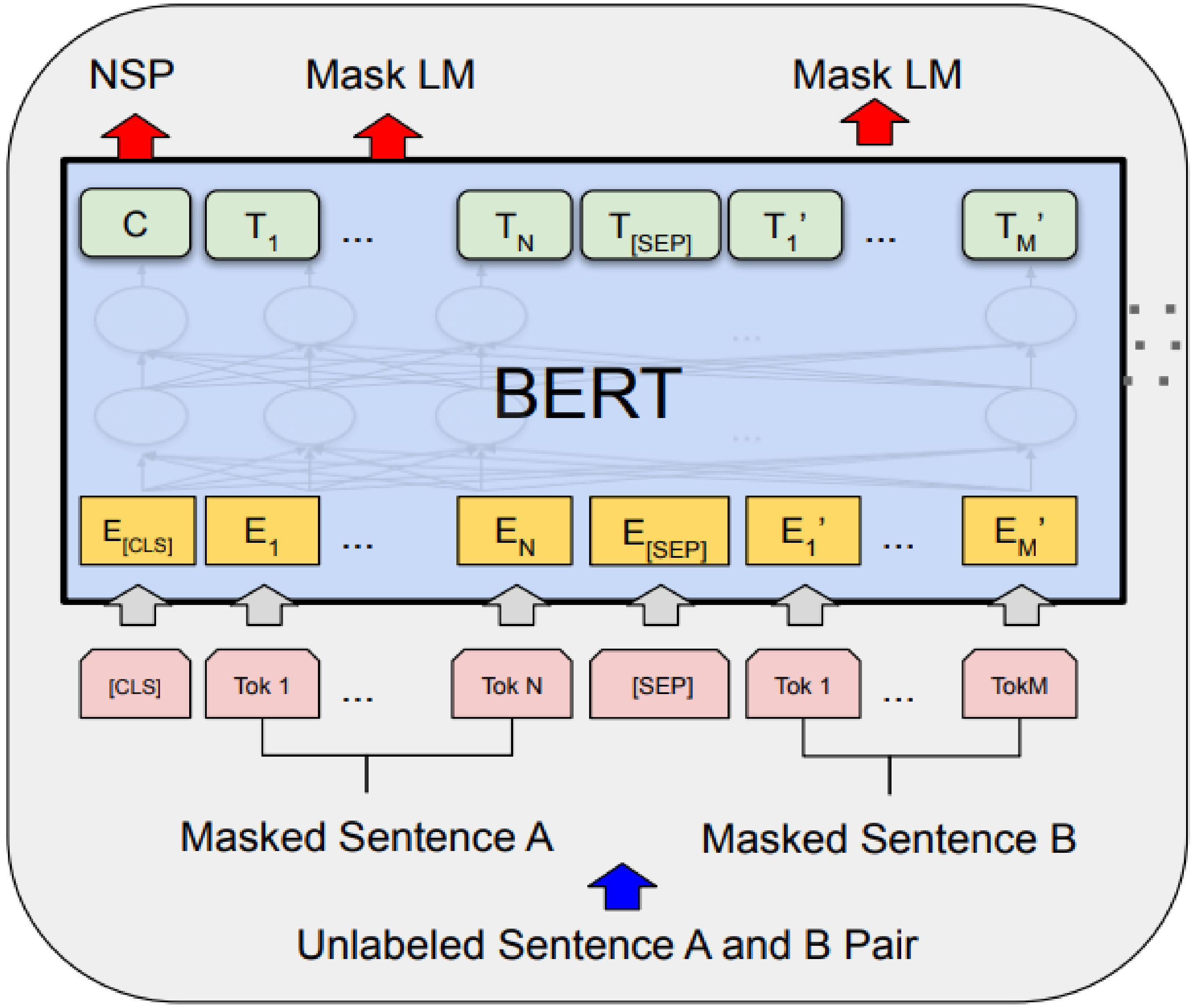
The input embeddings are the sum of the token, segment and position embeddings

There are two special tokens:
[SEP] – it's used to seperate sentences
[CLS] – its corresponding output token is used as agregate representation



BERT

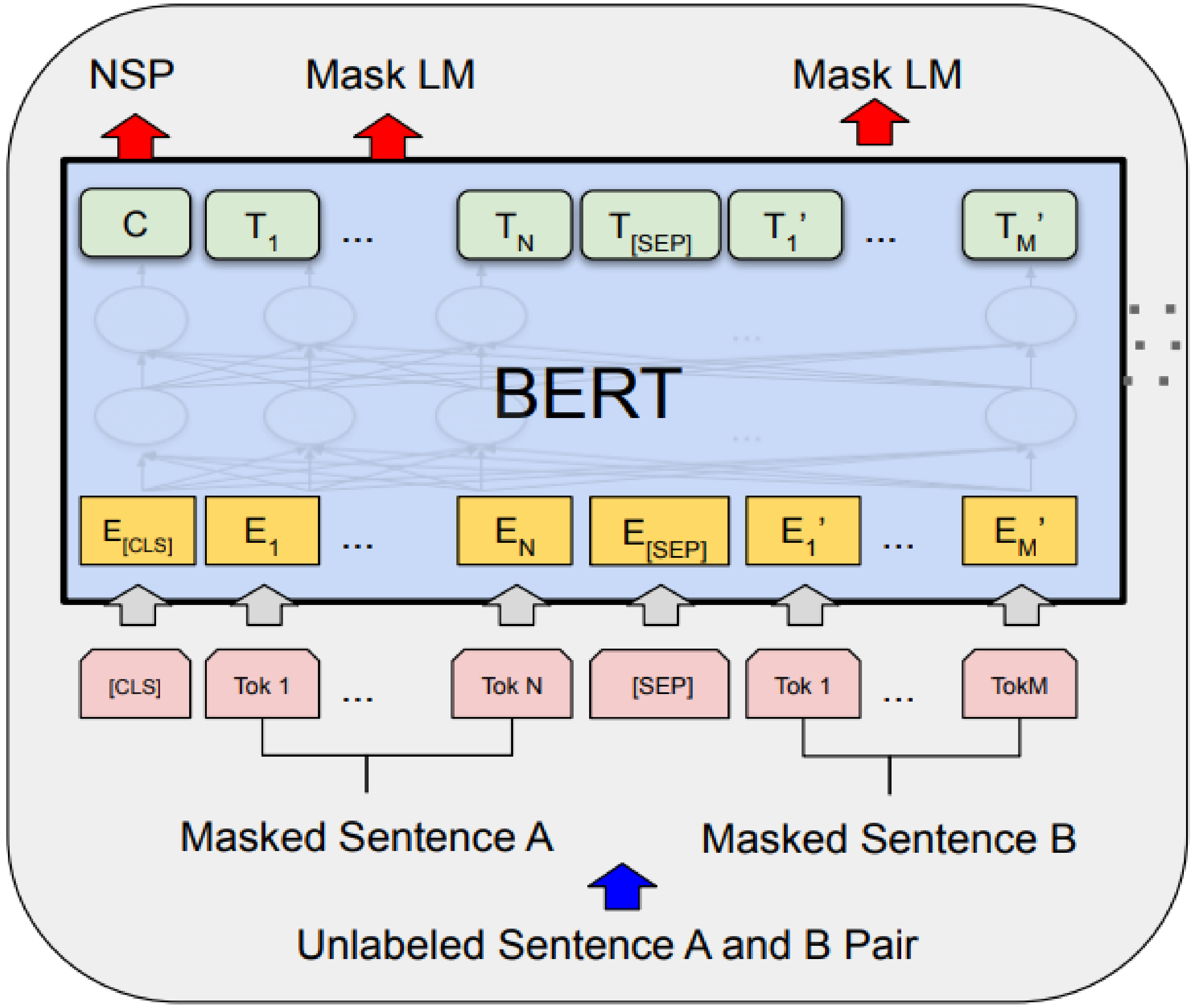
"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018



Pre-training

BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018

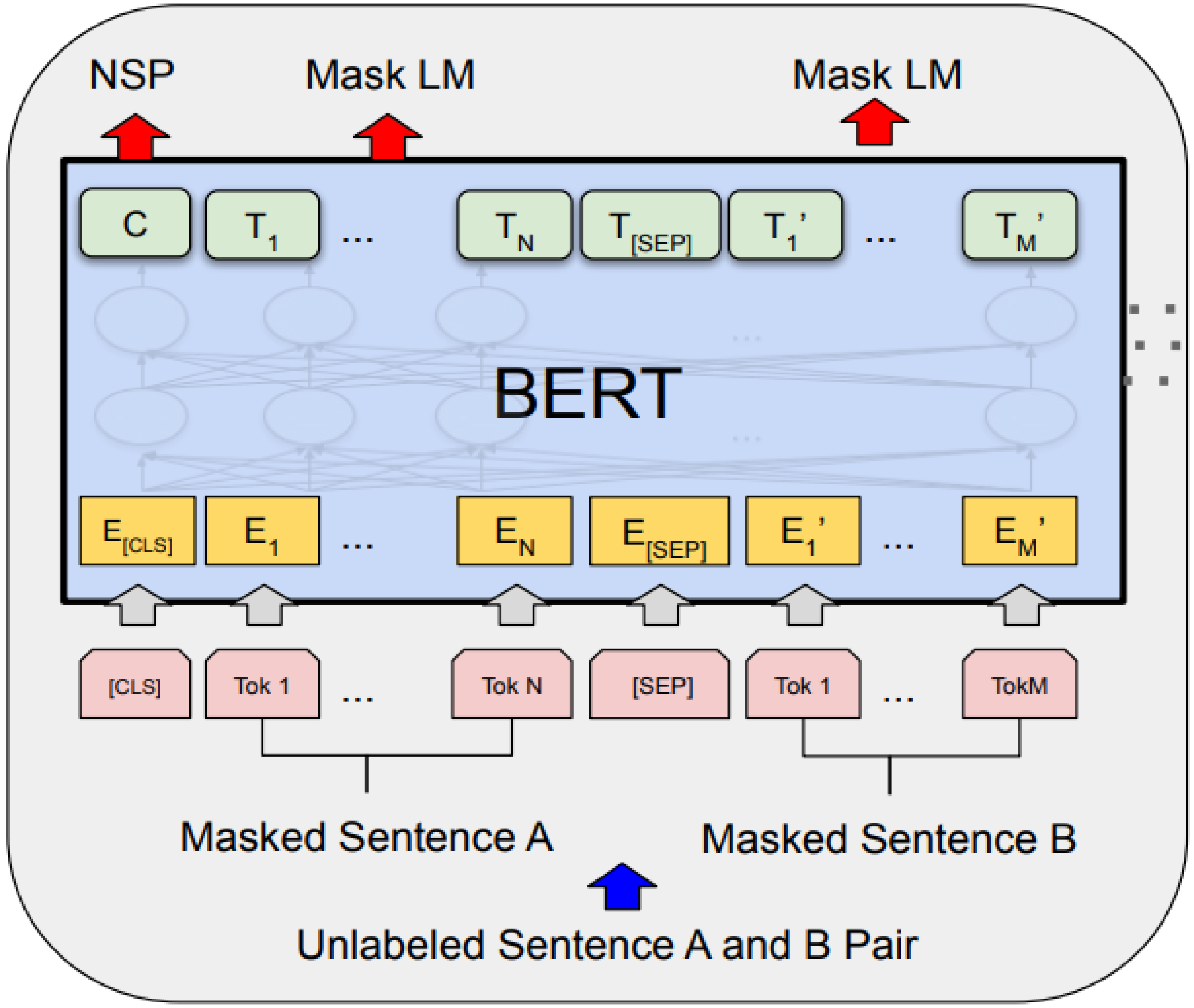


Pre-training

We pre-train the model using two unsupervised tasks.

BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018



Pre-training

We pre-train the model using two unsupervised tasks.

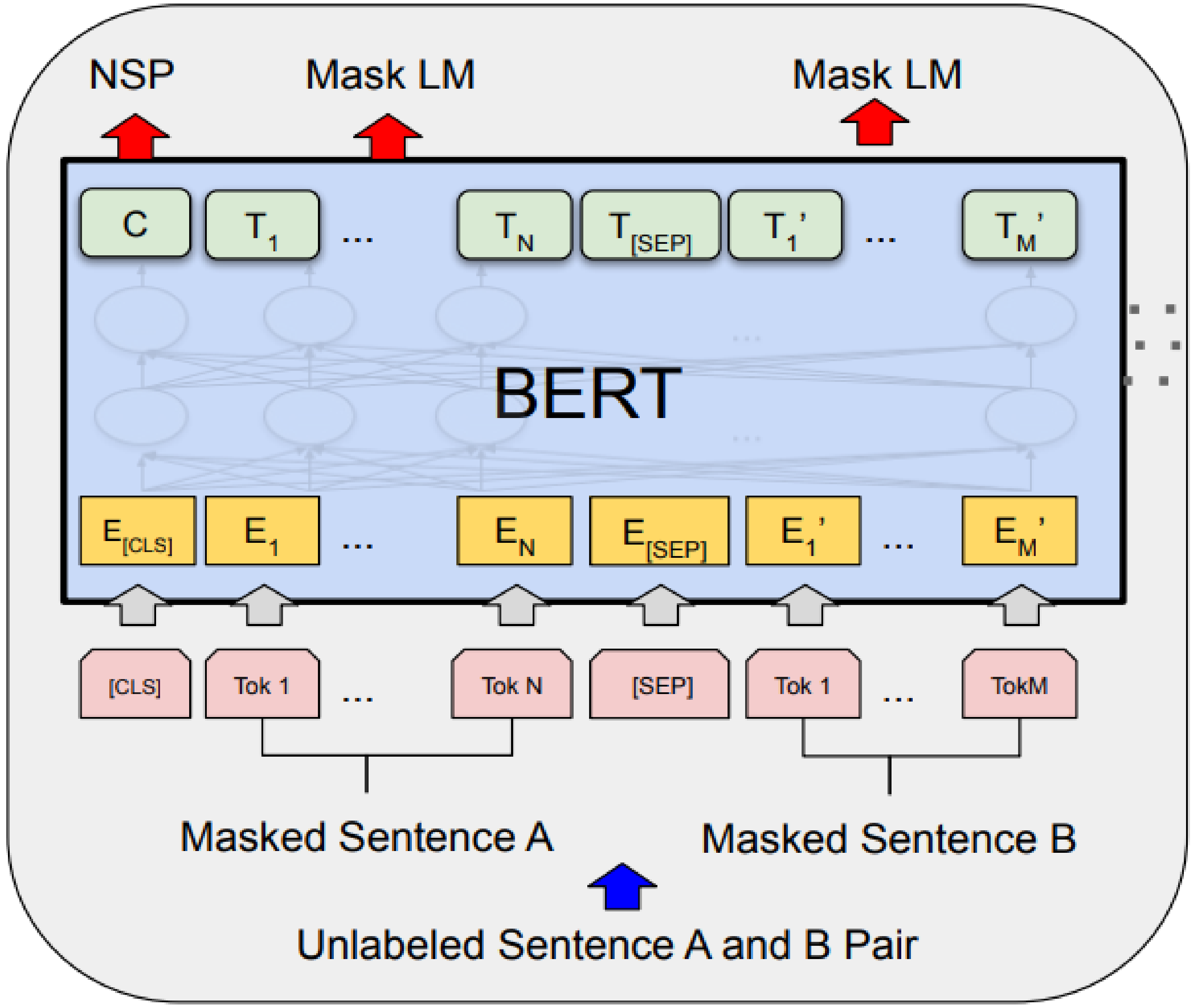
Task 1: Masked LM

Mask some percentage of the input tokens at random, and then predict those masked tokens.

To mitigate mismatch between pre-training and fine-tuning replace some `[MASK]` tokens with random tokens.

BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018



Pre-training

We pre-train the model using two unsupervised tasks.

Task 1: Masked LM

Mask some percentage of the input tokens at random, and then predict those masked tokens.

To mitigate mismatch between pre-training and fine-tuning replace some `[MASK]` tokens with random tokens.

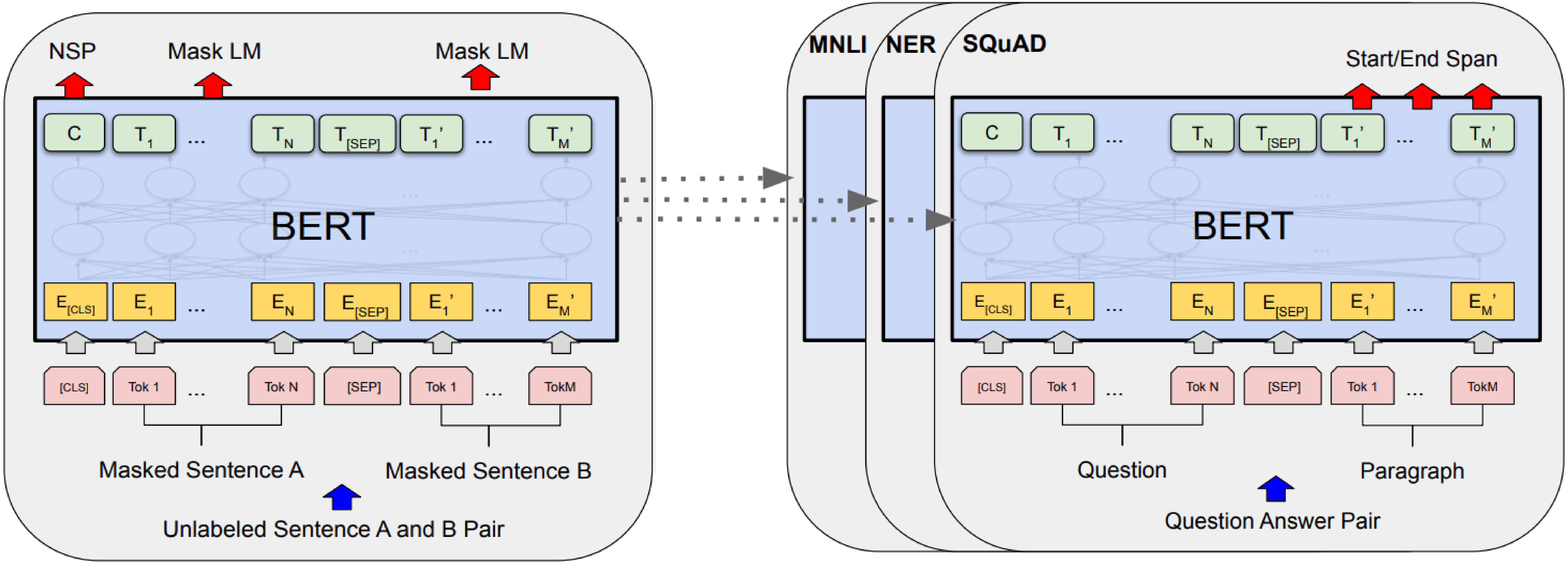
Task 2: Next Sentence Prediction (NSP)

Let the sentence `B` be the actual next sentence that follows `A` 50% of the time and a random sentence from the corpus another 50% of the time.

Use C to output prediction if `B` follows `A`.

BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018

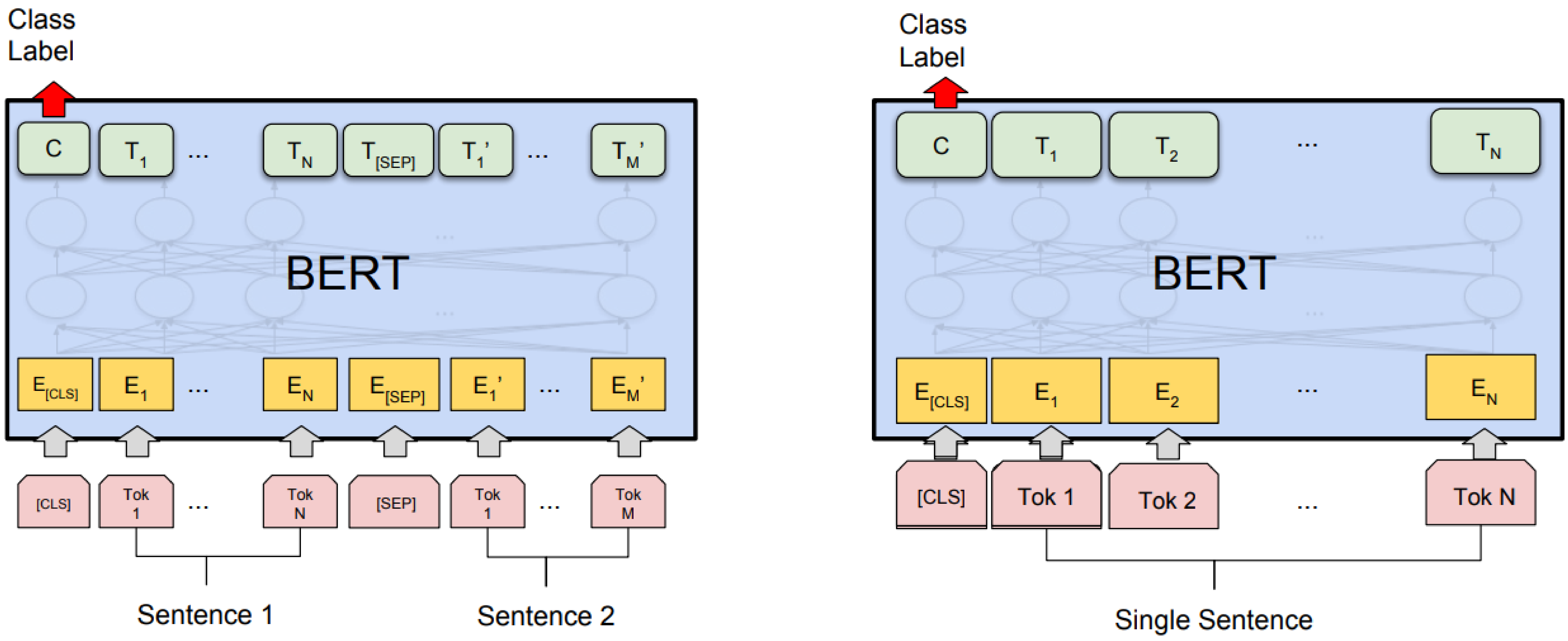


Pre-training

Fine-Tuning

BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018

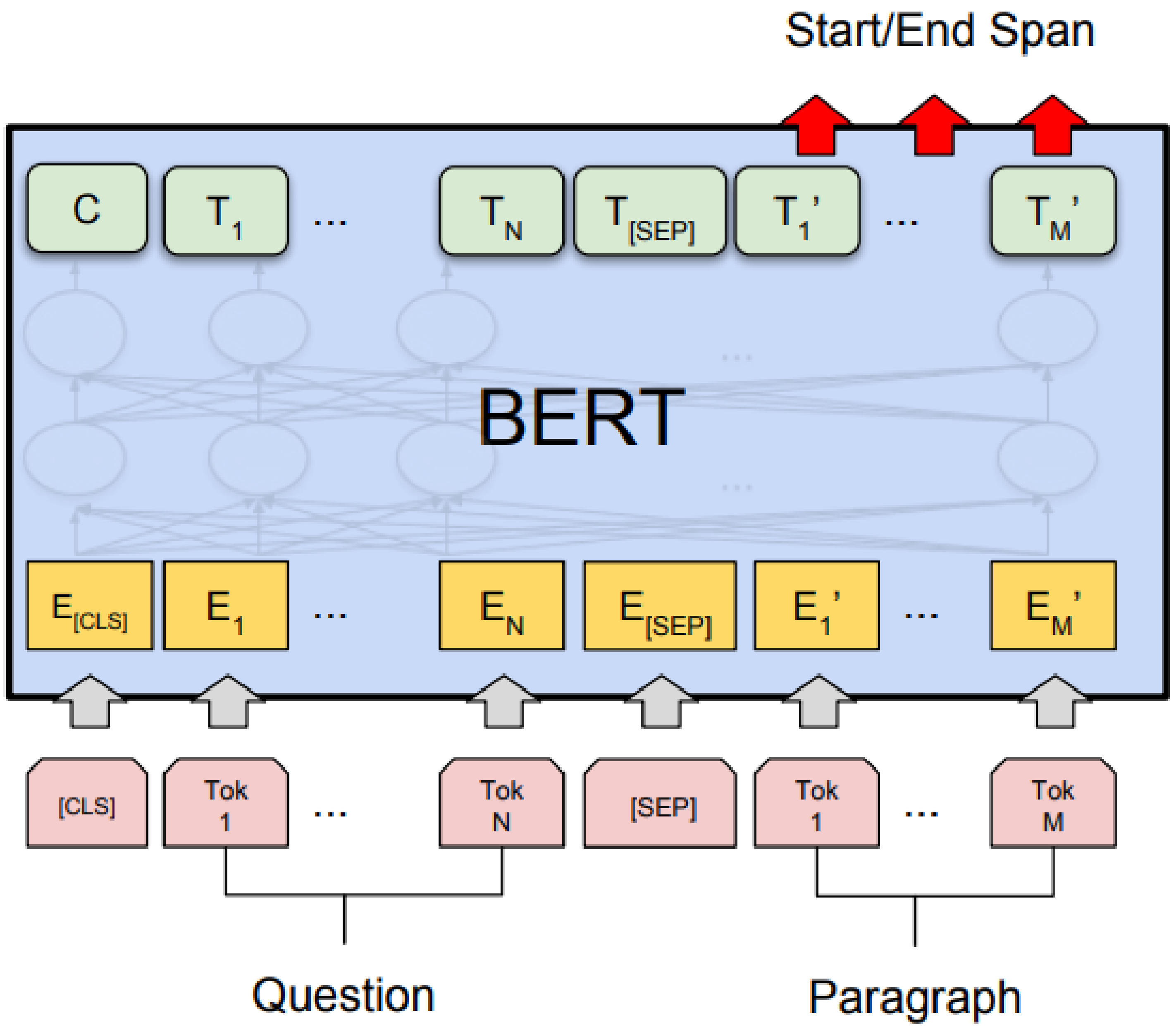


(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

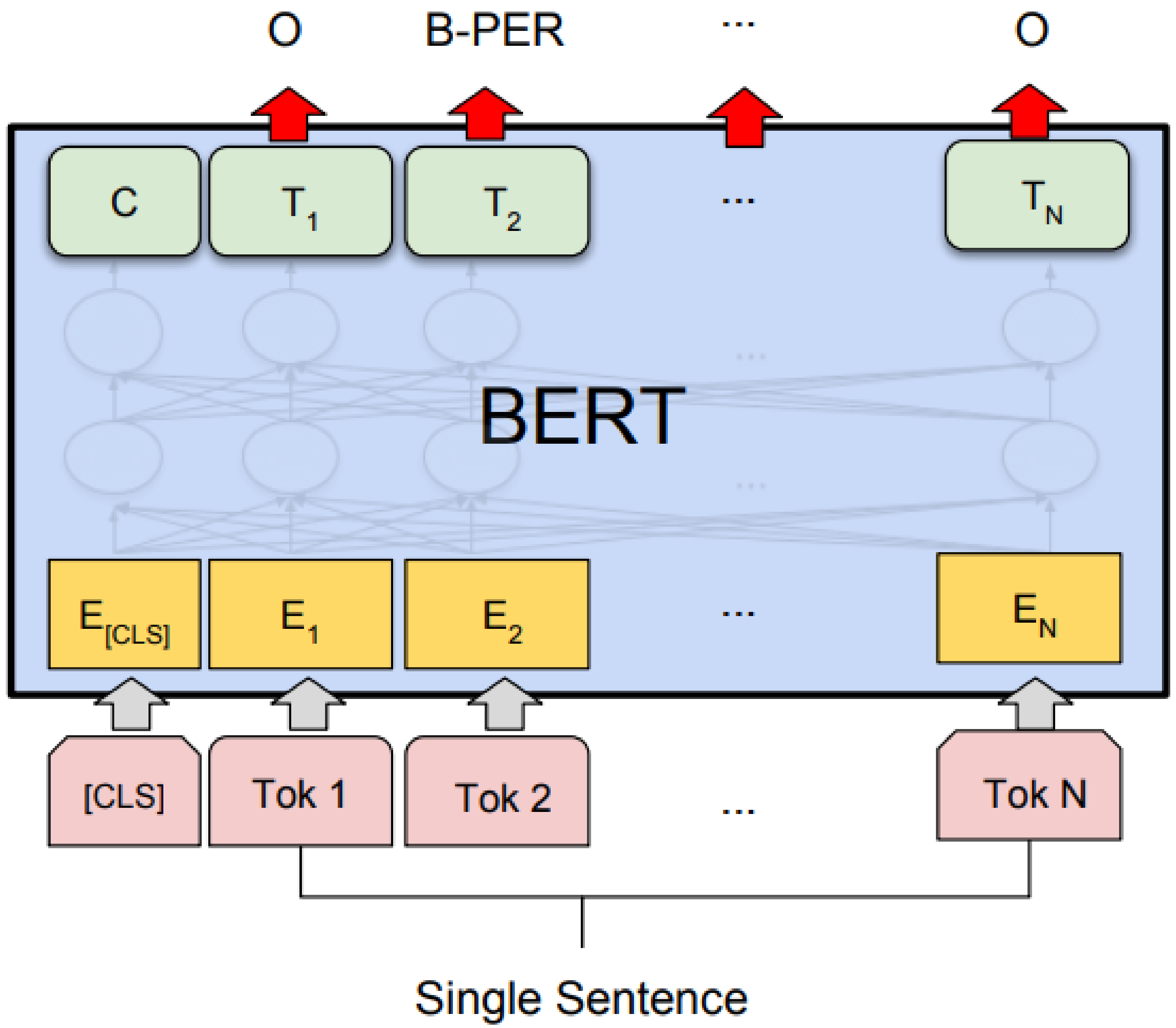
(b) Single Sentence Classification Tasks: SST-2, CoLA

BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018



(c) Question Answering Tasks:
SQuAD v1.1



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

BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018

New state-of-the-art results on 11 NLP tasks

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	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Table 1: GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard). The number below each task denotes the number of training examples. The "Average" column is slightly different

t|
n
a

System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
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 1.1 results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.

System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	86.3	89.0	86.9	89.5
#1 Single - MIR-MRC (F-Net)	-	-	74.8	78.0
#2 Single - nlnet	-	-	74.2	77.1
Published				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (Single)	-		71.4	74.4
Ours				
$\text{BERT}_{\text{LARGE}}$ (Single)	78.7	81.9	80.0	83.1

Table 3: SQuAD 2.0 results. We exclude entries that use BERT as one of their components.

System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
$\text{BERT}_{\text{BASE}}$	81.6	-
$\text{BERT}_{\text{LARGE}}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

: SWAG Dev and Test accuracies. †Human per-
ce is measured with 100 samples, as reported in
the SWAG paper.

BERT

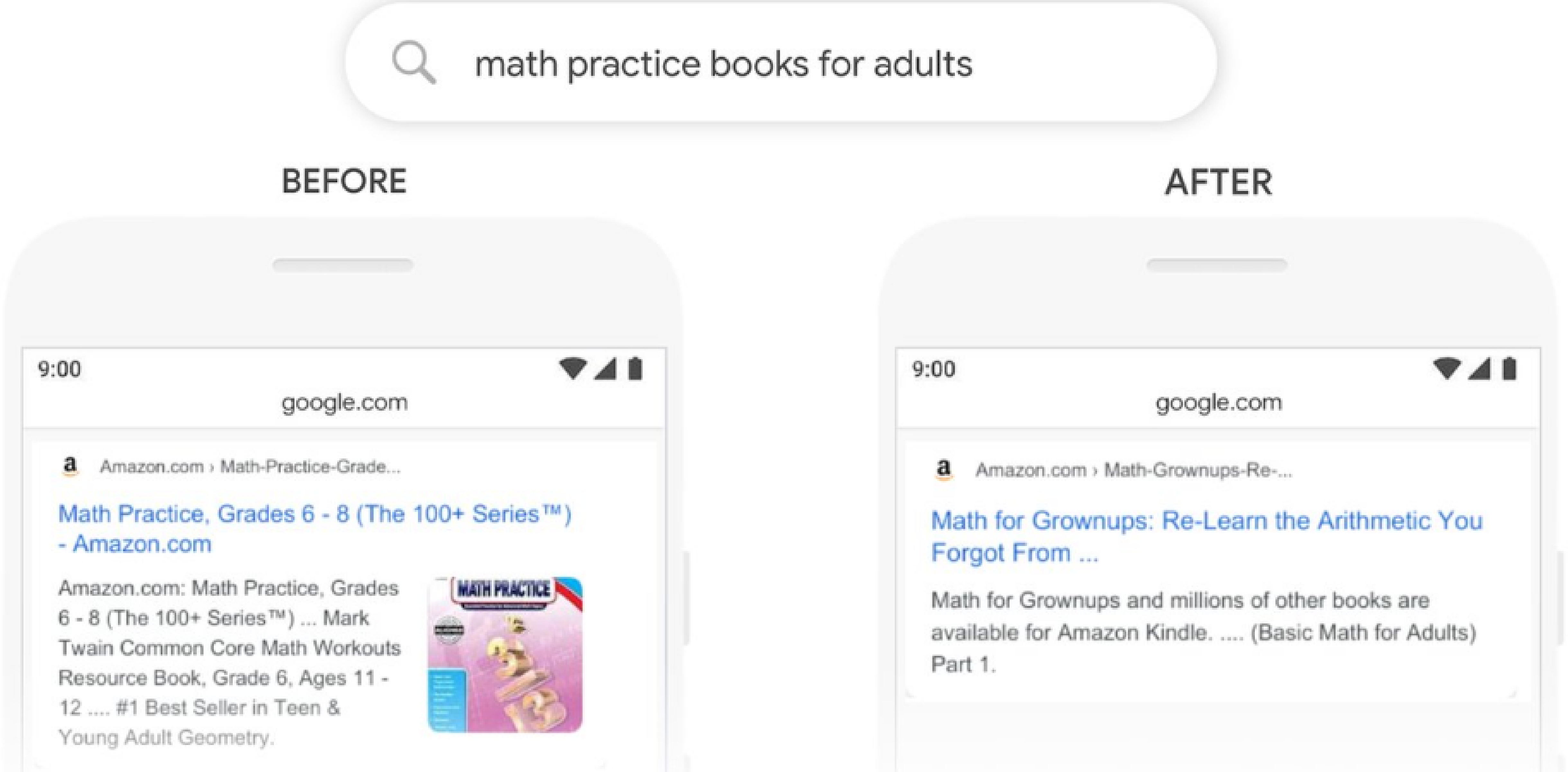
"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018

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.

BERT

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" Devlin et al. 2018



"While the previous results page included a book in the 'Young Adult' category, BERT can better understand that 'adult' is being matched out of context, and pick out a more helpful result."

https://blog.google/products/search/search-language-understanding-bert/

Questions