Pre-Training

NLP: Fall 2023

Anoop Sarkar

Preliminaries

Word structure and subword models

- NLP used to model the vocabulary in simplistic ways based on English
- Tokenize based on spaces into a sequence of "words"
- All novel words at test time were mapped to [UNK] (unknown token)

cs224n-2023-lecture9-pretraining.pdf

Byte Pair Encoding algorithm

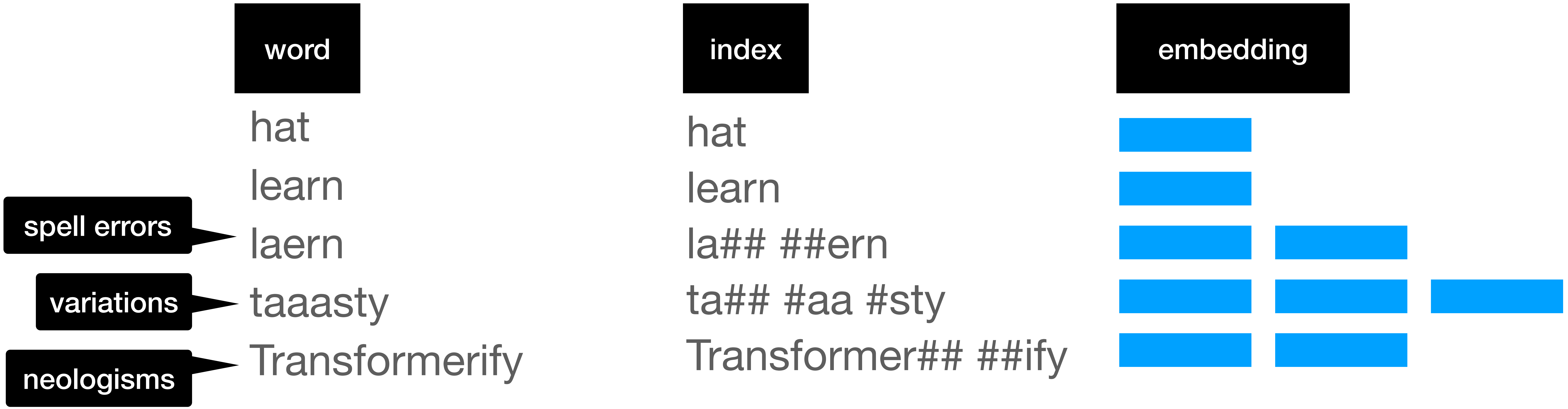
- Learn a vocabulary of parts of words (subwords)
- Vocabulary of subwords is produced before training a model on the training dataset (larger the better)
- At training and test time the vocabulary is split up into a sequence of known subwords
- Byte Pair Encoding (BPE) algorithm (takes max merges as input)
  - Init subwords with individual characters/bytes and "end of word" token.
  - Using the training data find most common adjacent subwords, merge and add to list of subwords
  - Replace all pairs of characters with new subword token; iterate until max merges

See bpe.ipynb

https://arxiv.org/abs/1508.07909

Word structure and subword models

- Common words are kept as part of the vocabulary (ignore morphology)
- Rarer words are split up into subword tokens
- In the worst case, words are split up into characters (or bytes)



Pre-training Transformers

Representation Learning

Brief History of Pre-training

1960 to 2015

- Singular Value Decomposition (1960s):
  - Take matrix $M \in |V| \times |V|$ of word co-occurrence counts
  - Use SVD to map $M = USV^T$ truncate to $|V| \times k$ initial singular values
  - Use truncated U use as word embeddings.
- Word2Vec/GloVe (2010):
  - Continuous Bag of Words (CBOW) - context words predict target word
  - Skip-gram - target word predicts each context word

Semi-supervised Sequence Learning

Andrew M. Dai
Google Inc.
adai@google.com

Quoc V. Le
Google Inc.
qvl@google.com

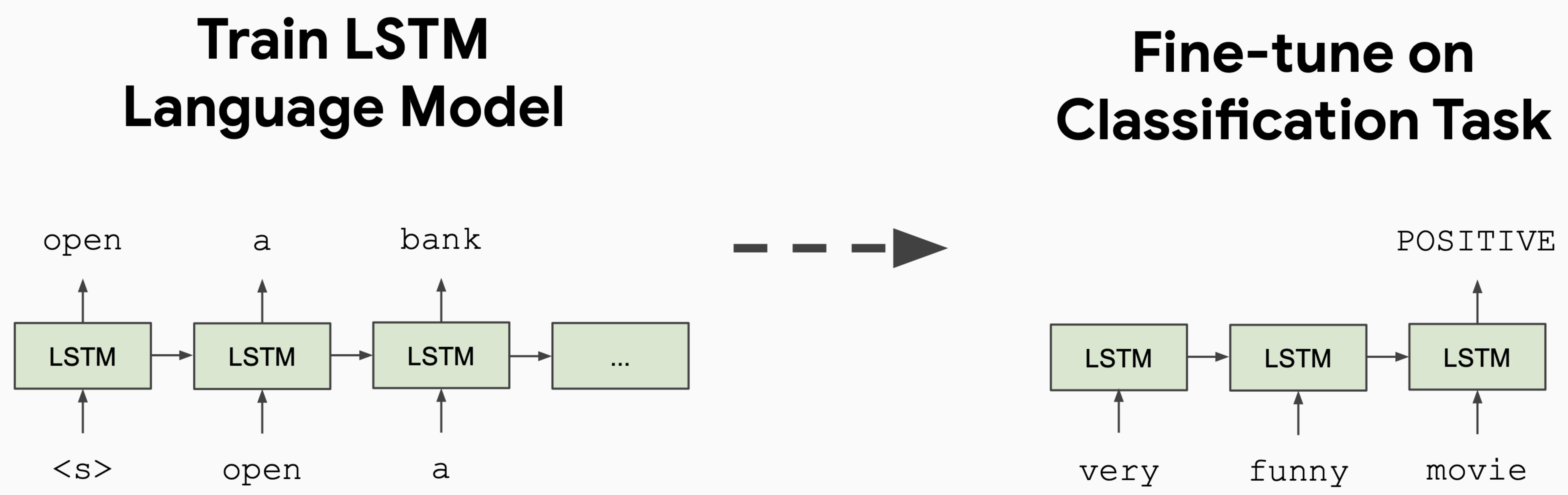


Fig from J. Devlin BERT slides https://arxiv.org/abs/1511.01432 **Nov 2015**

Deep contextualized word representations

ELMO

Matthew E. Peters[†], **Mark Neumann**[†], **Mohit Iyyer**[†], **Matt Gardner**[†],
{matthewp,markn,mohiti,mattg}@allenai.org

Christopher Clark[*], **Kenton Lee**[*], **Luke Zettlemoyer**[†*]
{csquared,kentonl,lsz}@cs.washington.edu

[†]Allen Institute for Artificial Intelligence
[*]Paul G. Allen School of Computer Science & Engineering, University of Washington

https://arxiv.org/abs/1802.05365 **Oct 2017**

ELMO

Fig from J. Devlin BERT slides

https://arxiv.org/abs/1802.05365

Improving Language Understanding by Generative Pre-Training

GPT1

Alec Radford OpenAI alec@openai.com

Karthik Narasimhan OpenAI karthikn@openai.com

Tim Salimans OpenAI tim@openai.com

Ilya Sutskever OpenAI ilyasu@openai.com

https://openai.com/research/language-unsupervised

Jun 2018

GPT1

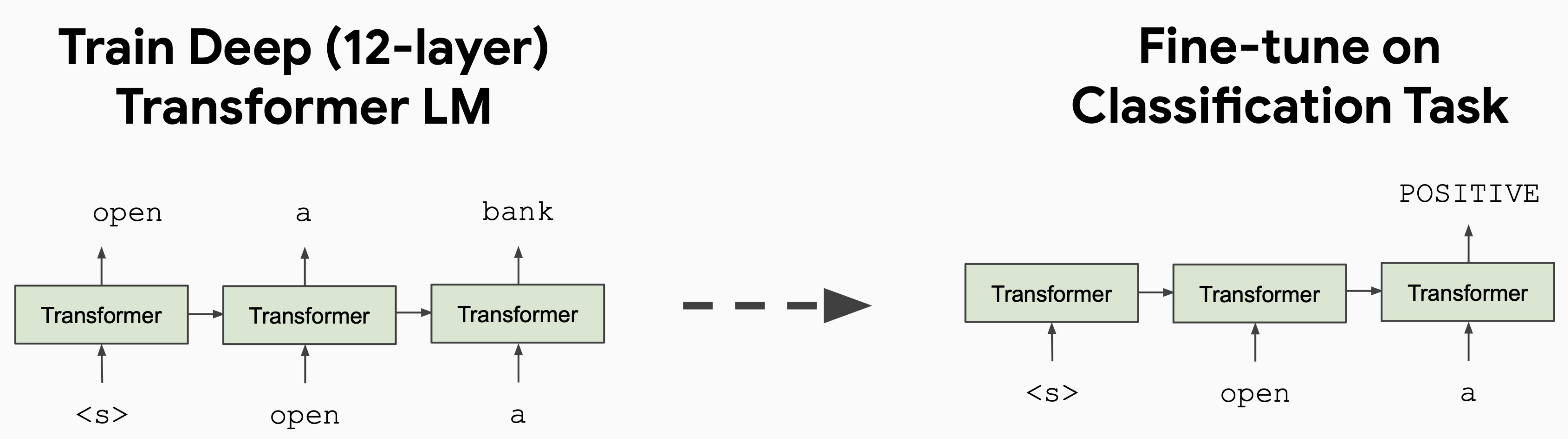


Fig from J. Devlin BERT slides

See also ULMFit: https://arxiv.org/abs/1801.06146

GPT1

Pre-training an autoregressive language model

BooksCorpus: 7K unpublished books (1B words)

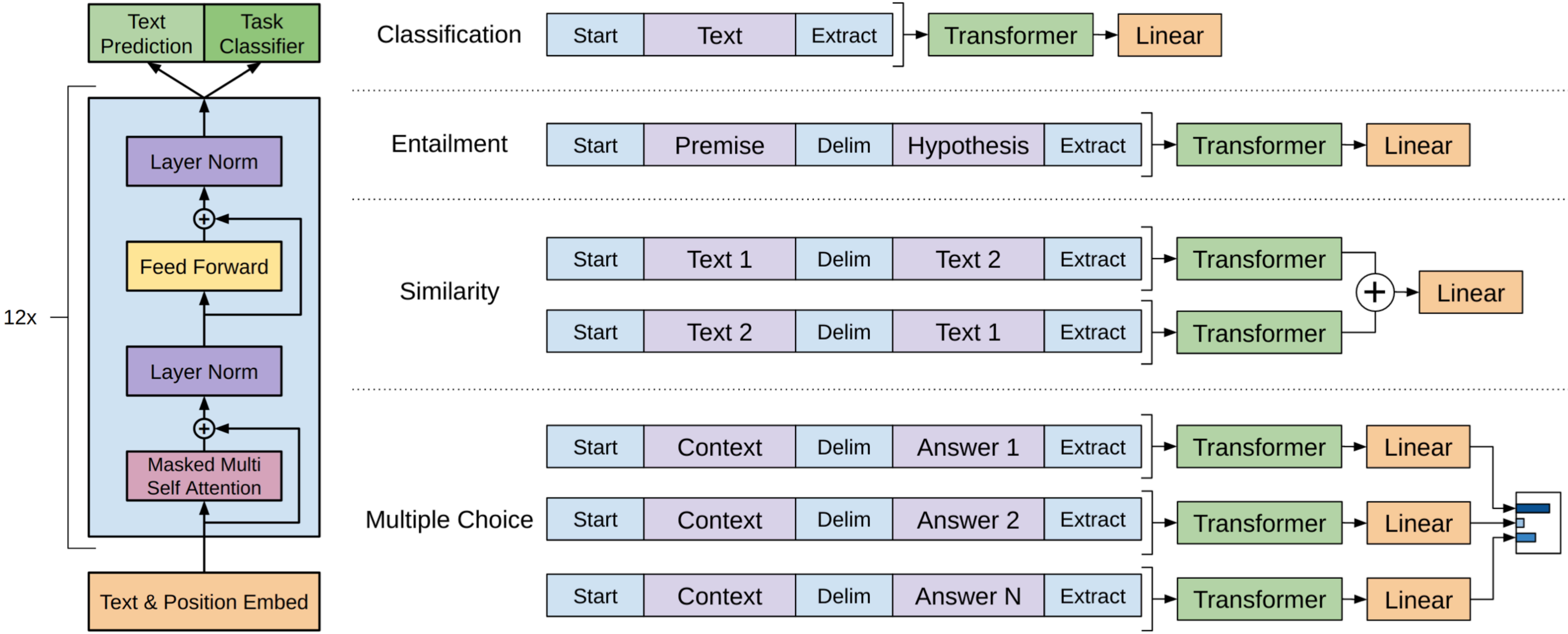
- Start with a large amount of unlabeled data $\mathcal{U} = \{u_1, \ldots, u_n\}$
- Pre-training objective: Maximize the likelihood of predicting the next token
  - $L_i(\mathcal{U}) = \sum_i \log P(u_i \mid u_{i-k}, \ldots, u_{i-1}; \Theta)$
- This is equivalent to training a Transformer decoder
  - $h_0 = U\boxed{W_e} + W_p$
  - $h_\ell = \text{transformer\_block}(h_{\ell-1}) \forall \ell \in [1,n]$
  - $P(u) = \text{softmax}(h_n \boxed{W_e^T})$
- Directionality is needed to generate a well-formed probability distribution

$U = (u_{-k}, \ldots, u_{-1})$ is the context vector of tokens

n is the number of Transformer layers

W_e is the token embedding matrix

W_p is the position embedding matrix


Text Prediction
Task Classifier
Layer Norm
Feed Forward
Layer Norm
Masked Multi Self Attention
12x
Text & Position Embed
Classification
Start
Text
Extract
Transformer
Linear
Entailment
Start
Premise
Delim
Hypothesis
Extract
Transformer
Linear
Similarity
Start
Text 1
Delim
Text 2
Extract
Transformer
Start
Text 2
Delim
Text 1
Extract
Transformer
Linear
Multiple Choice
Start
Context
Delim
Answer 1
Extract
Transformer
Linear
Start
Context
Delim
Answer 2
Extract
Transformer
Linear
Start
Context
Delim
Answer N
Extract
Transformer
Linear

Dataset	Task	SOTA	GPT1
SNLI	Textual entailment	89.3	89.9
MNLI matched	Textual entailment	80.6	82.1
MNLI mismatched	Textual entailment	80.1	81.4
SciTail	Textual entailment	83.3	88.3
QNLI	Textual entailment	82.3	88.1
RTE	Textual entailment	61.7	56.0
STS-B	Semantic similarity	81.0	82.0
QQP	Semantic similarity	66.1	70.3
MRPC	Semantic similarity	86.0	82.3
RACE	Reading comprehension	53.3	59.0
ROCStories	Commonsense reasoning	77.6	86.5
COPA	Commonsense reasoning	71.2	78.6
SST-2	Sentiment analysis	93.2	91.3
CoLA	Linguistic acceptability	35.0	45.4
GLUE	Multi task benchmark	68.9	72.8

https://openai.com/research/language-unsupervised

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

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Fig from J. Devlin BERT slides

Transformer encoder
BERT model architecture

- Multi-headed self attention (models context)
- Feed-forward layers (non-linear hierarchical feature representation learning)
- LayerNorm and residuals (allows training of deep networks)
- Positional embeddings (allows model to learn relative position representation)

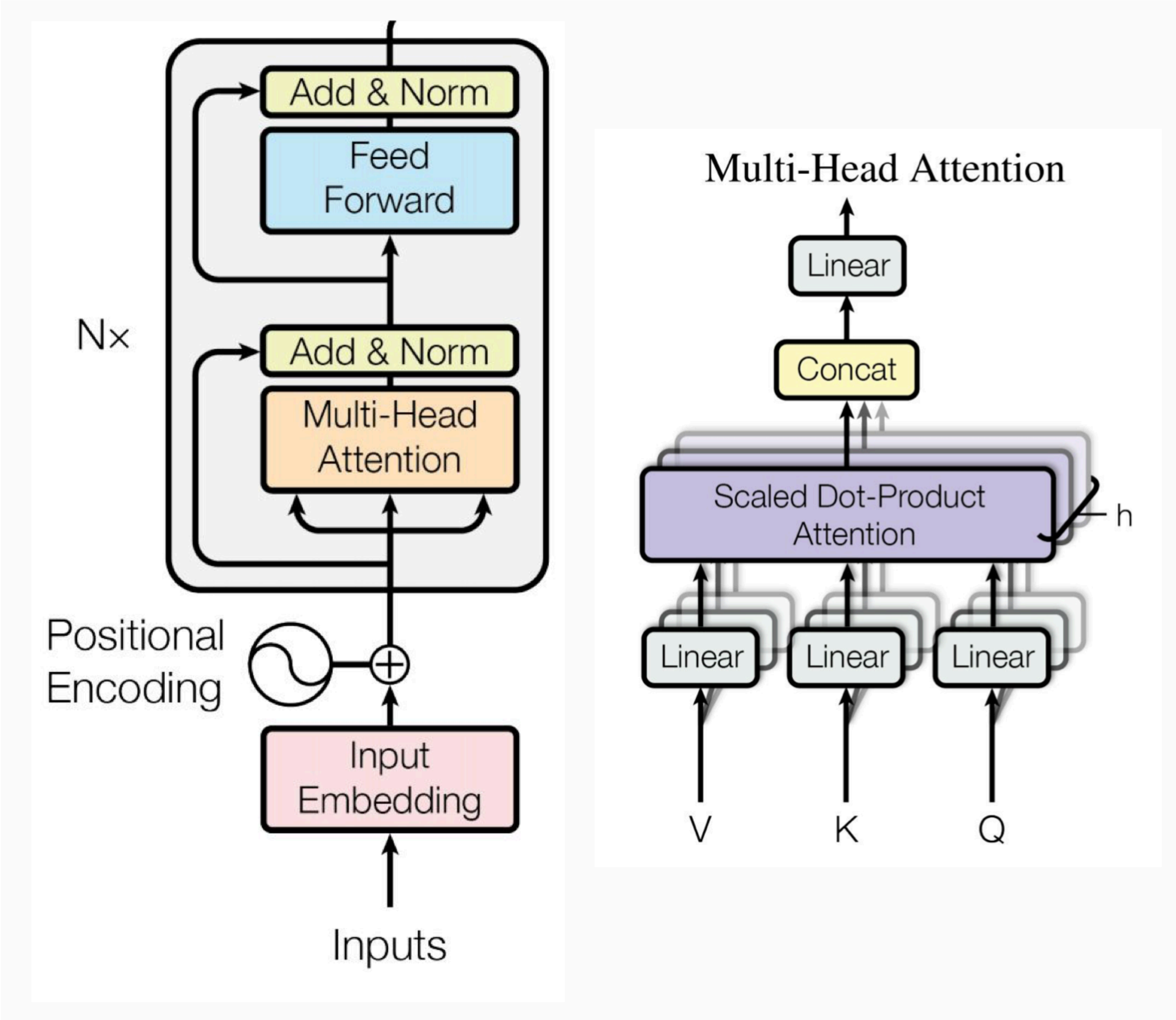
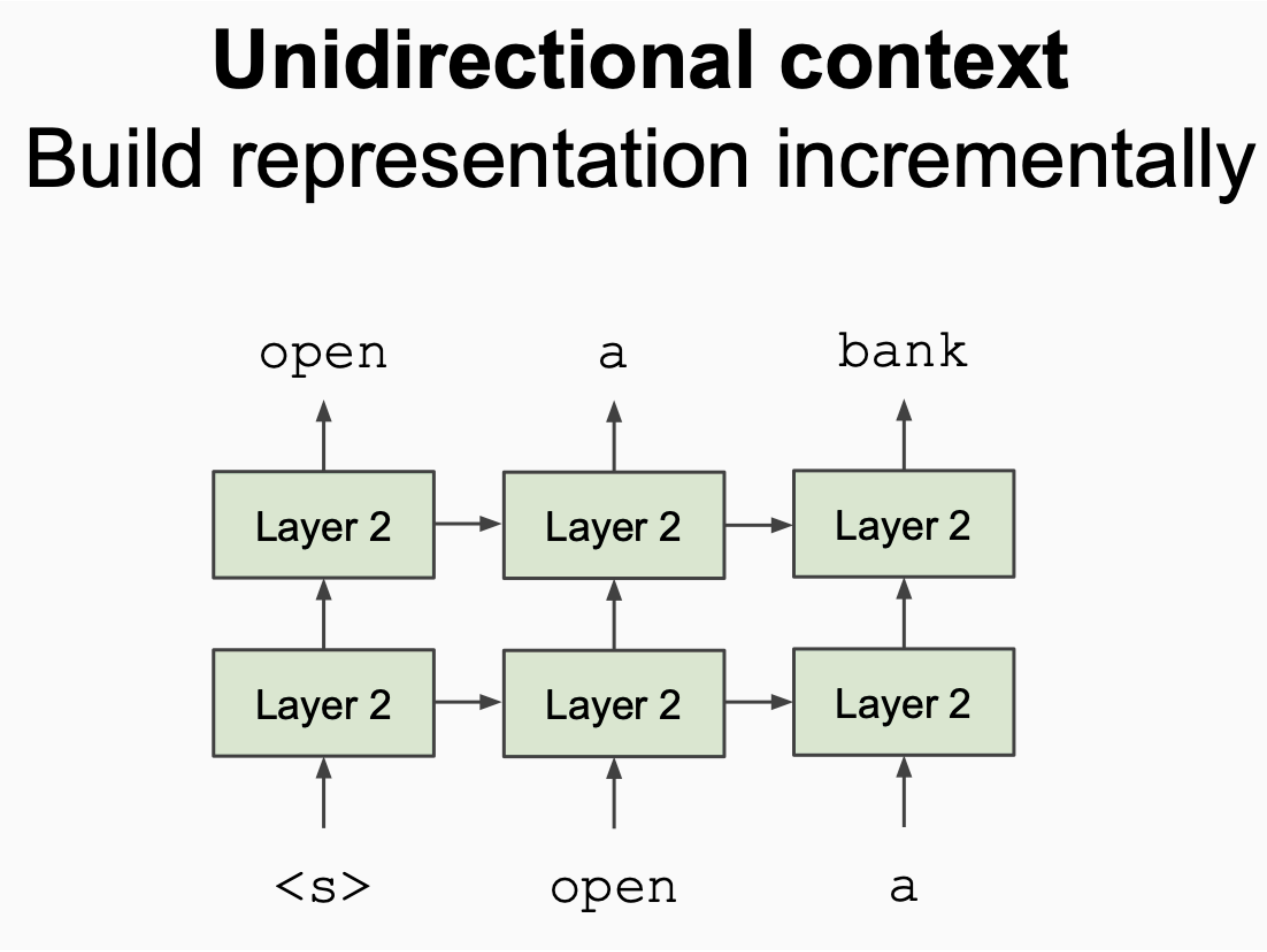


Fig from J. Devlin BERT slides

Directionality

Unidirectional context (GPT) vs Bidirectional context (ELMO)



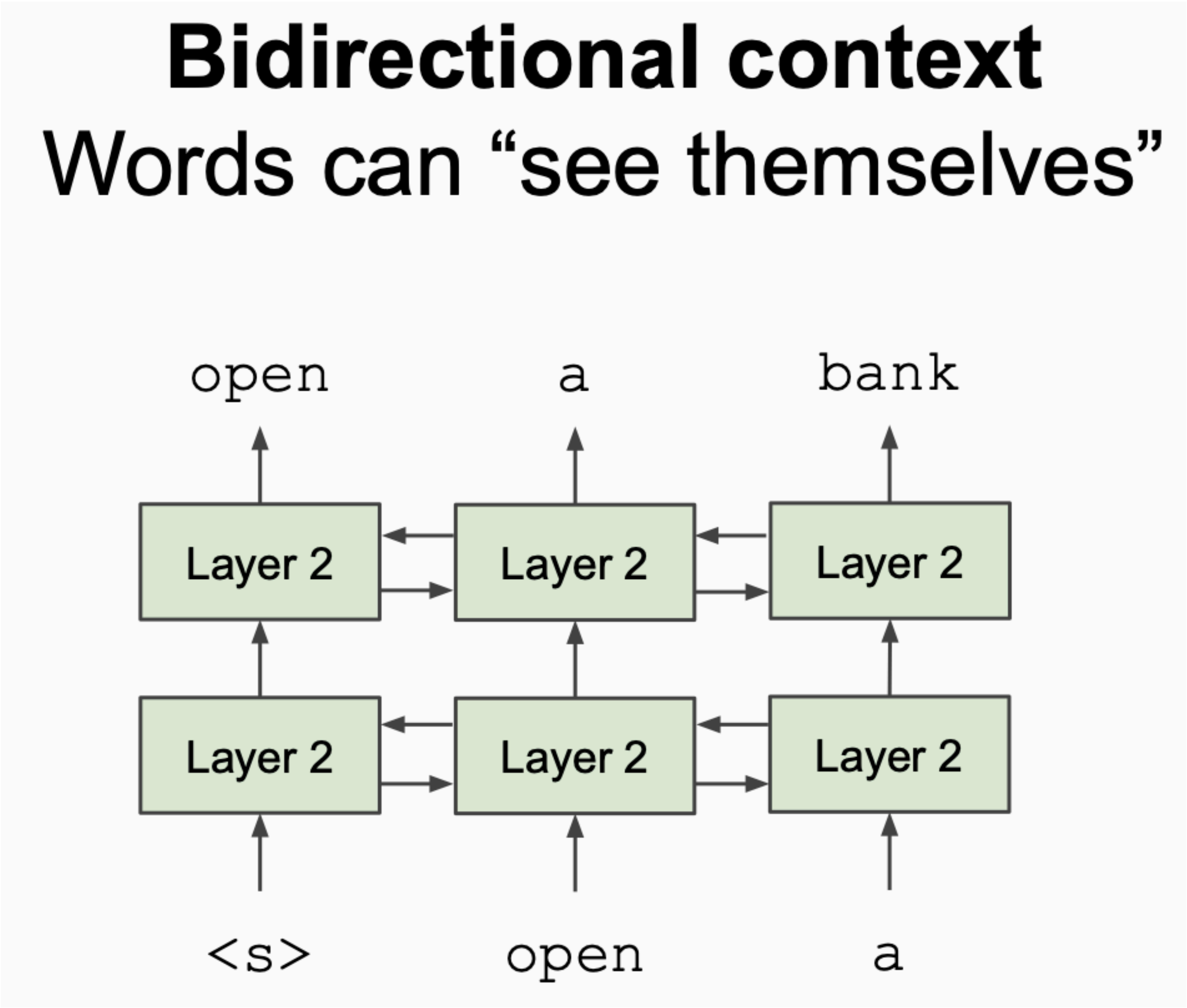


Fig from J. Devlin BERT slides

Bidirectional representation learning

without probabilities

- Use the entire sentence context
- Don't worry about probabilities just solve a task and learn parameters
- Solution: use two loss functions

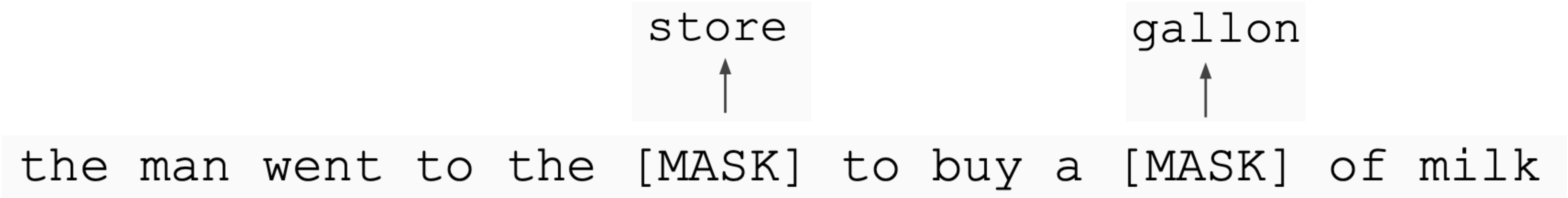
1. Language model but masking a single arbitrary token at a time.
    - Called the cloze task (Taylor 1953) aka Masked language modeling
2. Next sentence prediction (based on the Skip-Thought Vectors paper)

https://psycnet.apa.org/record/1955-00850-001

https://arxiv.org/abs/1506.06726

Masked LM

- Loss function to train a Transformer
- Keep most of the sentences intact. Mask out k% of the input tokens (k=15)
- Predict the masked tokens
- Too little masking: too many epochs needed to train a good representation
- Too much masking: not enough context to predict the token



Masked LM

Problem: Mask token is never used for any fine-tuning task

- Predict 15% of the tokens but do not replace tokens with [MASK] 100% of the time (for those 15% of tokens)

- Instead:

  1. 80% of the time replace with [MASK] and predict the right token

  2. 10% of the time replace with a random word and predict the right token

  3. 10% of the time keep the token unchanged and predict

Next Sentence Prediction (NSP)

Learning sentence representations

- BERT is always provided with two sentences at a time during training separated by a [SEP] token: `[CLS] Sentence A [SEP] Sentence B`
- Replace 50% of Sentence B with a random sentence
- Otherwise use the Sentence B that follows Sentence A
- Loss function: Predict if Sentence B follows Sentence A or not

Sentence A = `The man went to the store.`
Sentence B = `He bought a gallon of milk.`
Label = `IsNextSentence`

Sentence A = `The man went to the store.`
Sentence B = `Penguins are flightless.`
Label = `NotNextSentence`

30K subword vocabulary

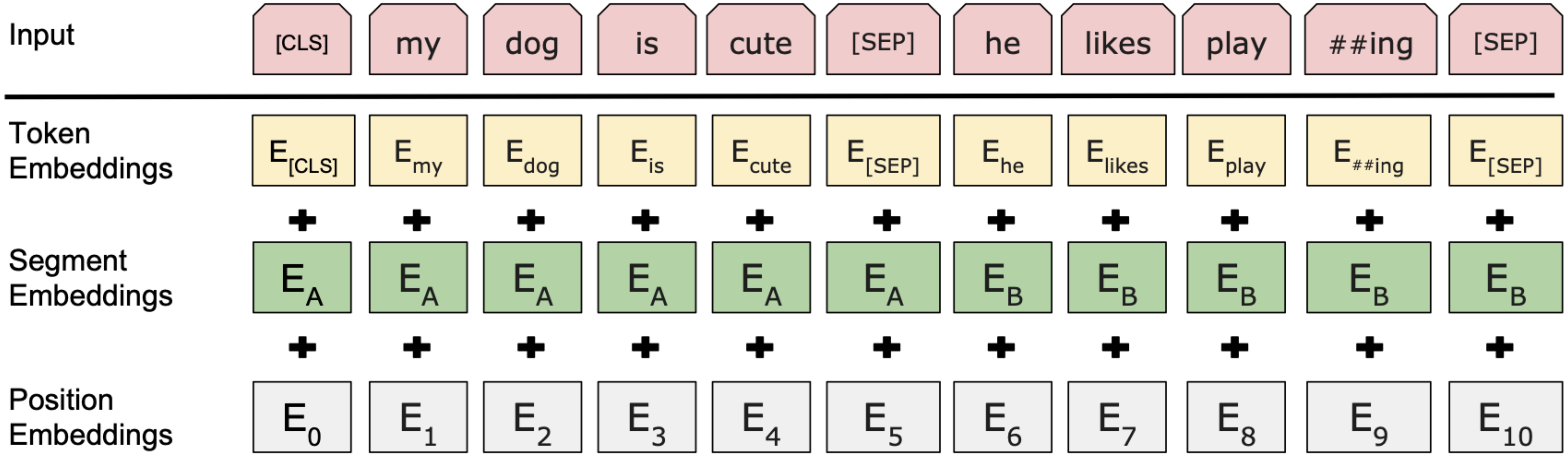
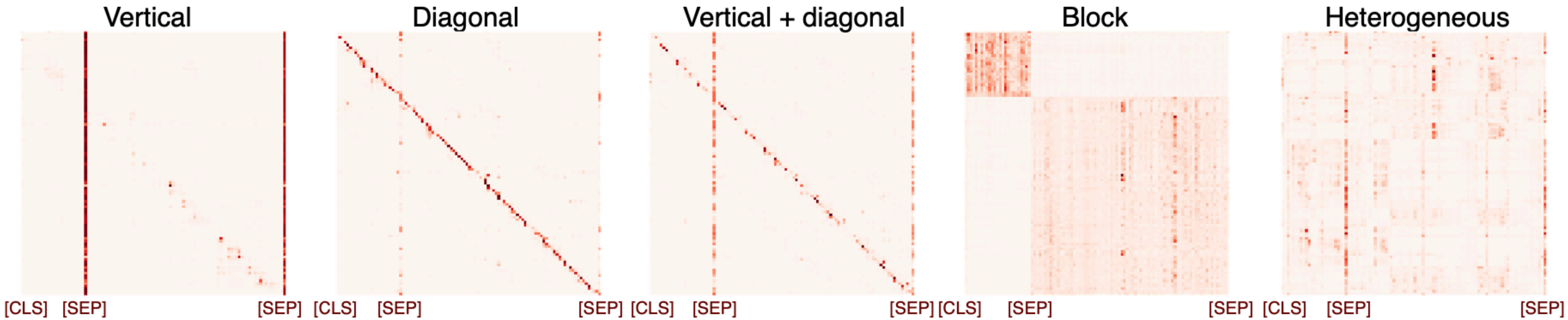


Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

Fig from J. Devlin BERT slides



(a) Reference: typical BERT self-attention patterns by Kovaleva et al. (2019).

Model and Training

- **Data**: Wikipedia (2.5B tokens) + BooksCorpus (800M tokens)
- **Batch size**: 131,072 tokens
  - 1024 sequences $\times$ 128 length
  - 256 sequences $\times$ 512 length
- **Training time**: 1M steps (~40 epochs)
- **Optimizer**: AdamW, 1e-4 learning rate, linear decay
- **BERT-base**: 12 layer, 768 hidden, 12 attention heads. 110M parameters (=GPT1)
- **BERT-large**: 24 layer, 1024 hidden, 16 attention heads. 340M parameters

Fine-tuning procedure

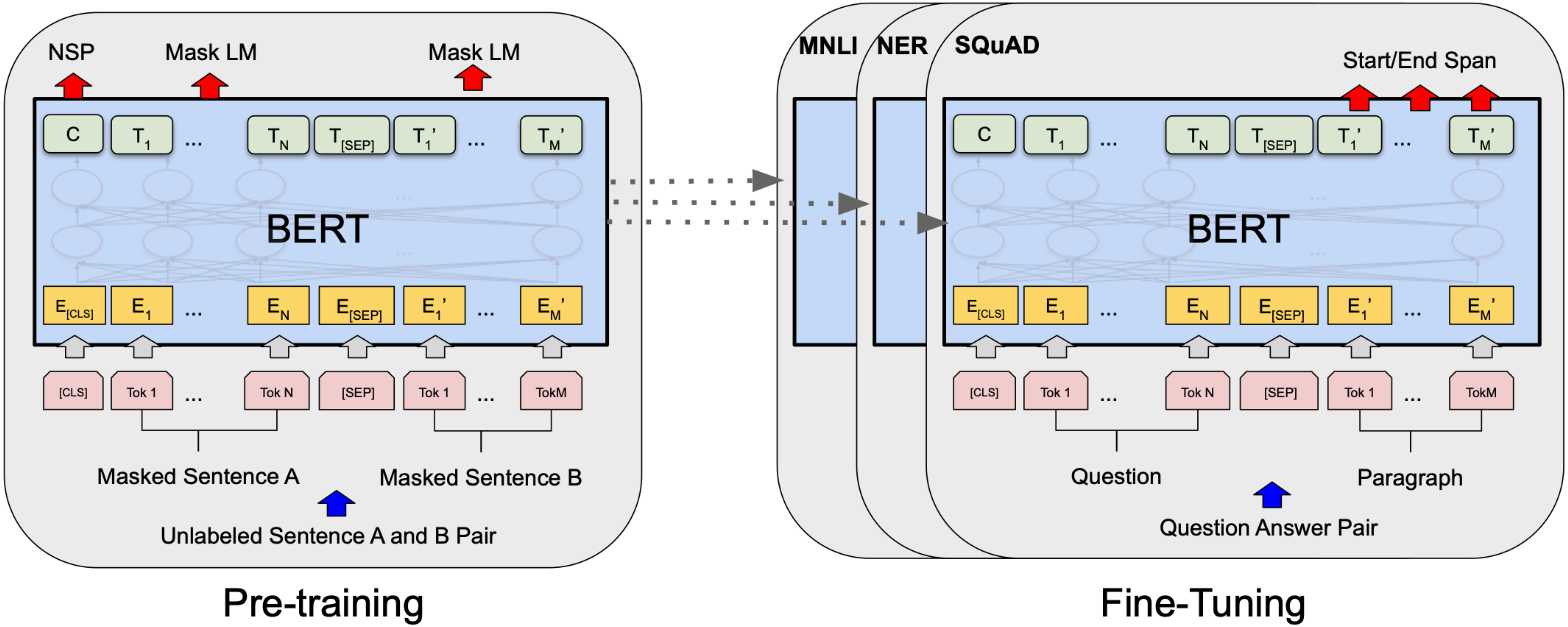
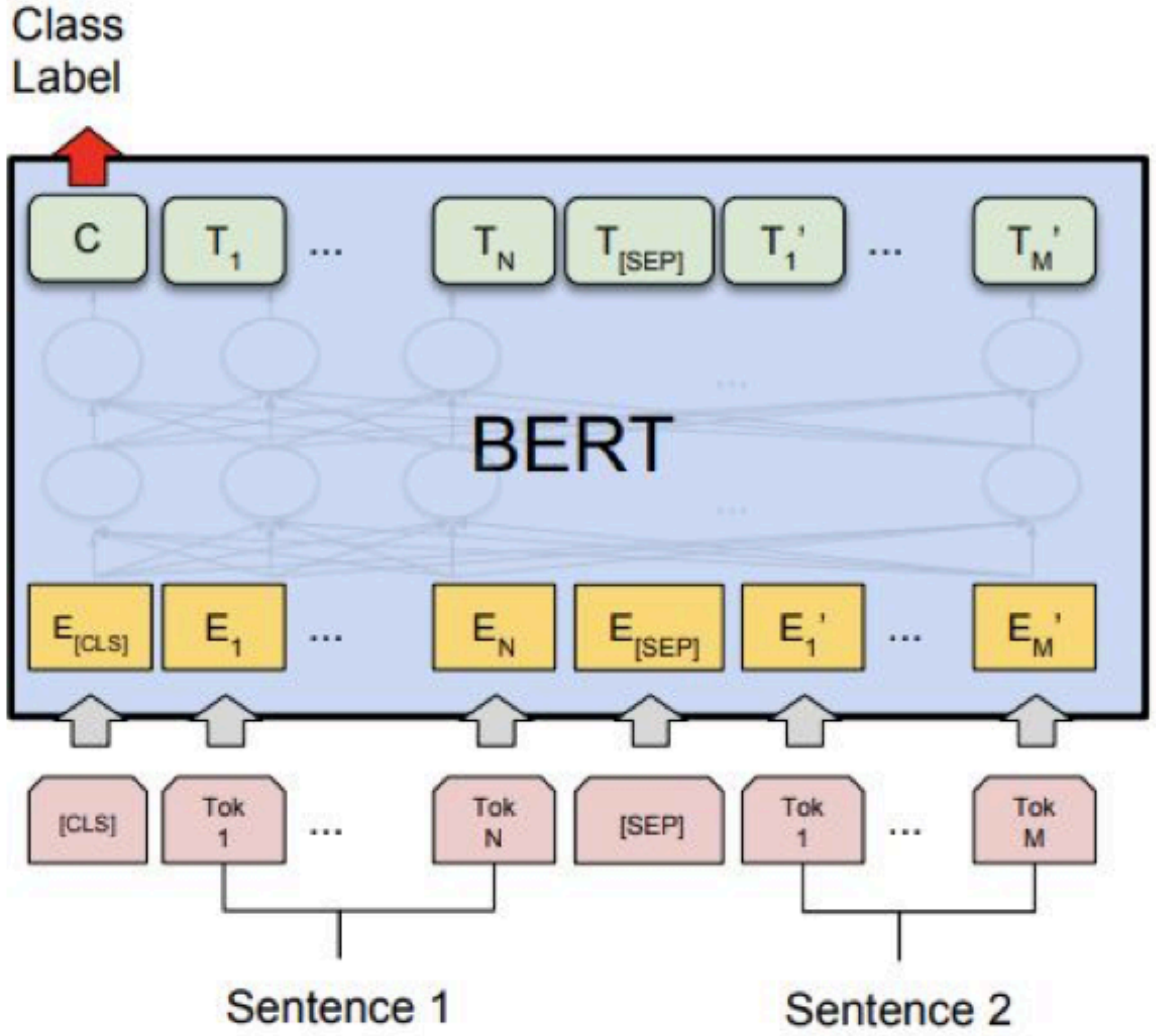


Fig from J. Devlin BERT slides

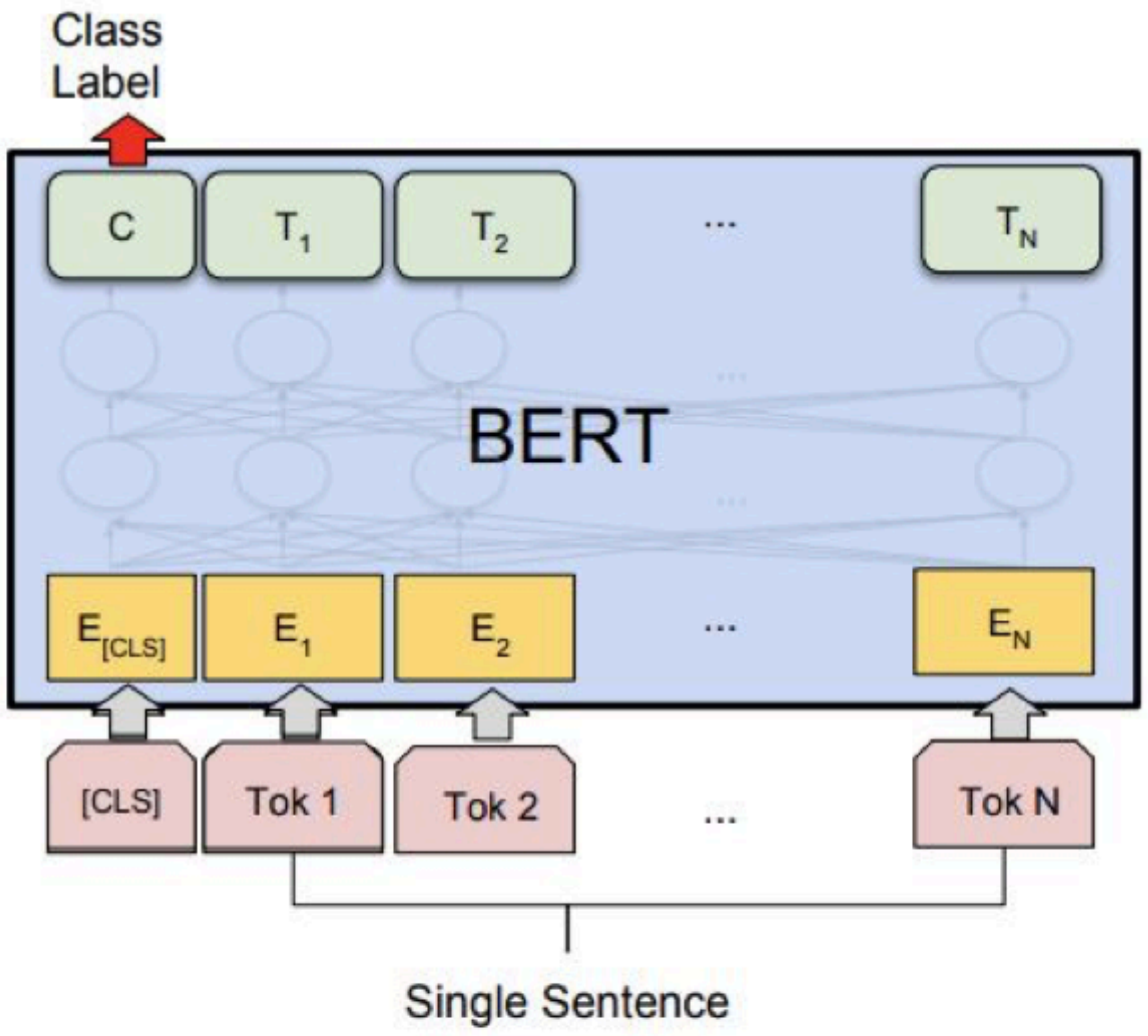
Fine-tuning for sentence pair classification



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

Fig from J. Devlin BERT slides

Fine-tuning for single sentence classification



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

Fig from J. Devlin BERT slides

Fine-tuning for question answering tasks

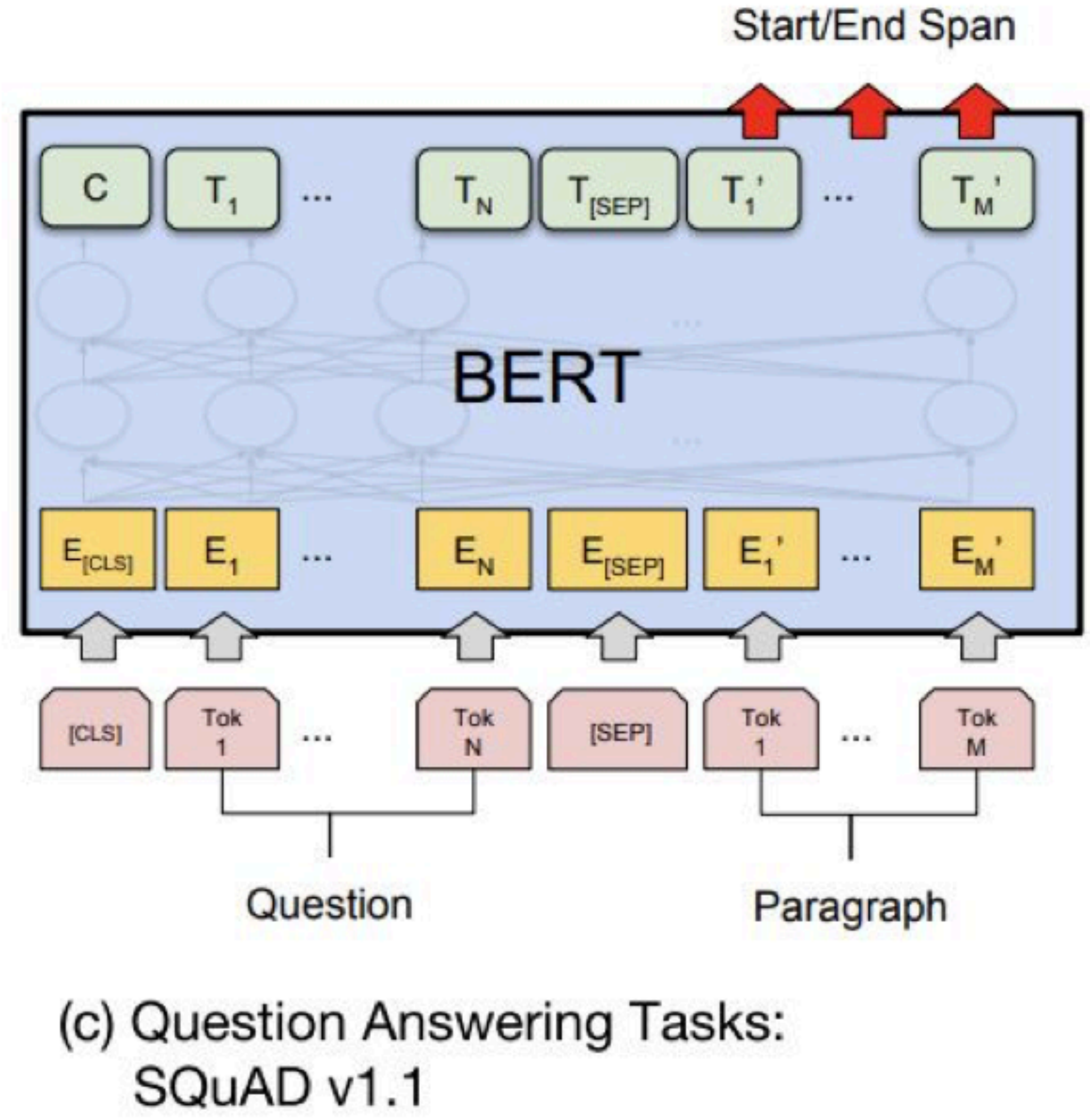
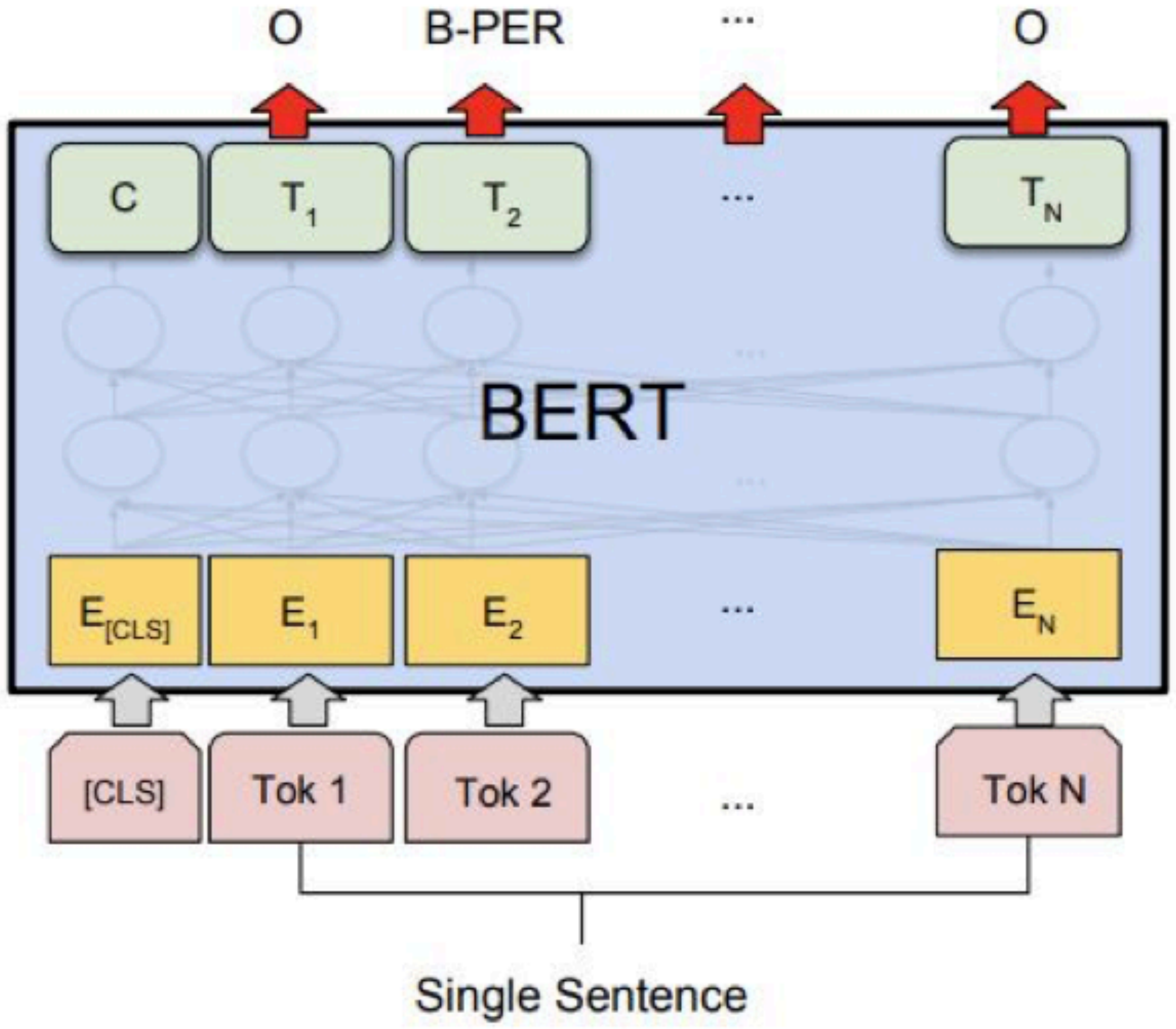


Fig from J. Devlin BERT slides

Fine-tuning for single sentence tagging tasks



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

Fig from J. Devlin BERT slides

GLUE Results

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
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

MultiNLI

Premise: Hills and mountains are especially sanctified in Jainism.
Hypothesis: Jainism hates nature.
Label: Contradiction

CoLa

Sentence: The wagon rumbled down the road.
Label: Acceptable

Sentence: The car honked down the road.
Label: Unacceptable

Fig from J. Devlin BERT slides

Directionality and training time

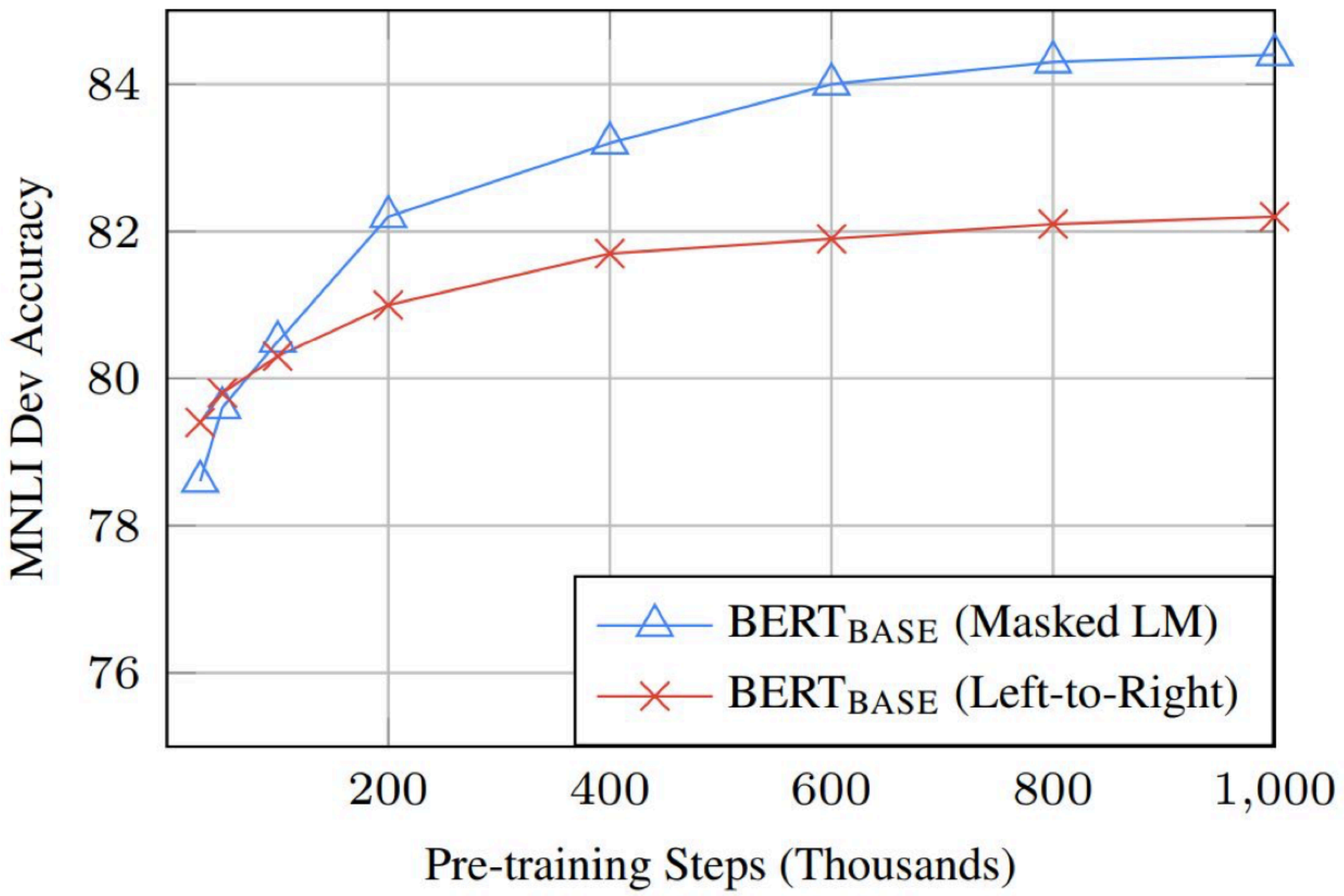


Fig from J. Devlin BERT slides

Effect of Model Size

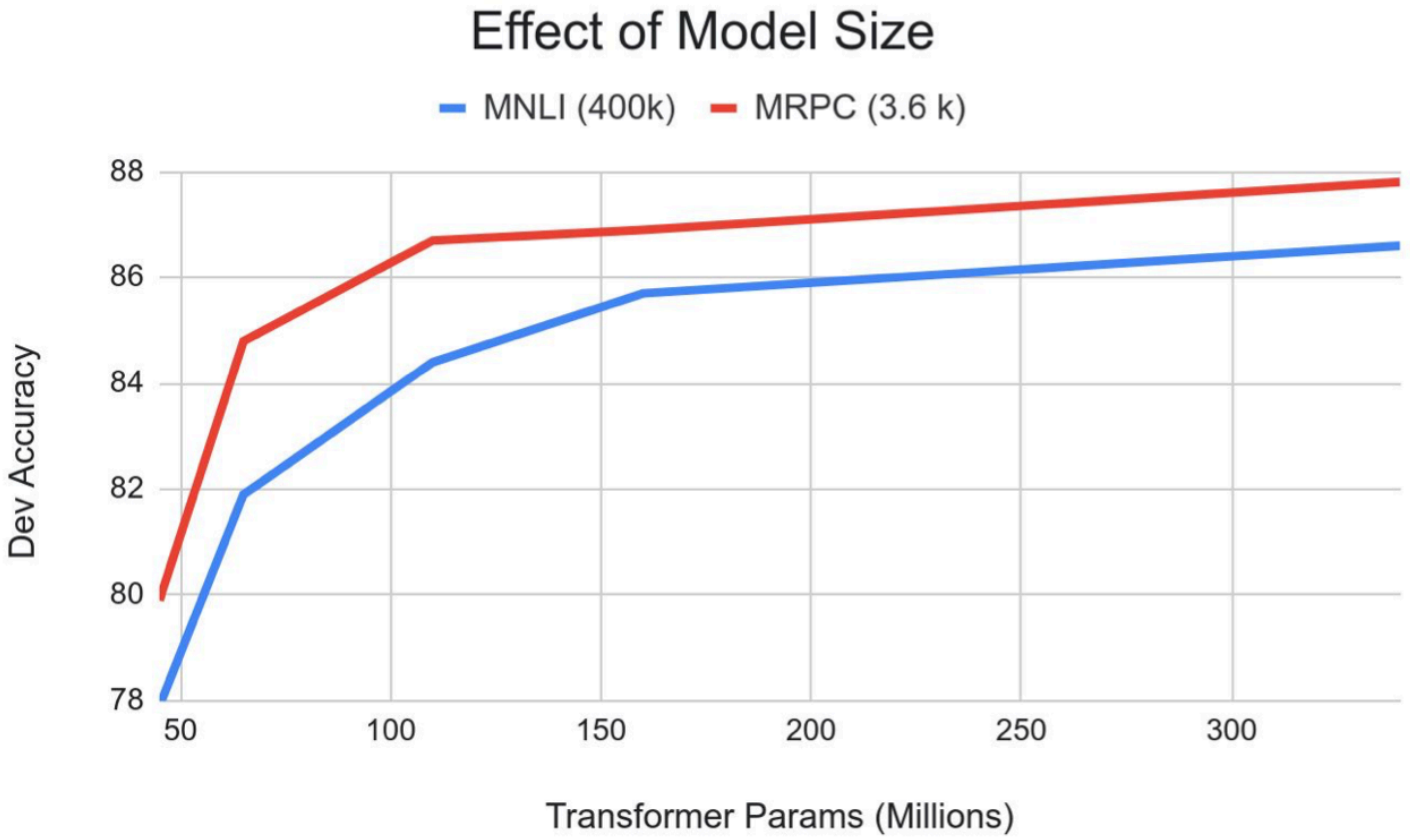


Fig from J. Devlin BERT slides

Open Source Release

BERT was successful due to full open-source release

- BERT-base and BERT-large released under a permissive license (Apache 2.0)
- Model-only release (not part of a larger codebase): open source DL toolkits
- No dependencies except TensorFlow or PyTorch
- Abstracted so all you had to do was import a single module
- End-to-end examples to train SoTA models on many tasks
- Comprehensive README and readable, well-documented code
- Good support (for first few months)

Environmental Impact

Model name	Number of parameters	Power consumption	CO_2eq emissions
GPT-3	175B	1,287 MWh	*502 tons*
Gopher	280B	*1,066 MWh*	*352 tons*
OPT	175B	*324 MWh*	70 tons
BLOOM	176B	433 MWh	25 tons

BERT Extensions

RoBERTa https://arxiv.org/abs/1907.11692

Liu+ 2019

- Robustly optimized BERT pre-training: dynamic masking; train on text blocks
- Train BERT on more data and for more epochs
  - Even on same data, training for longer helps
  - More data leads to a better model
- Remove Next Sentence Prediction (NSP) loss

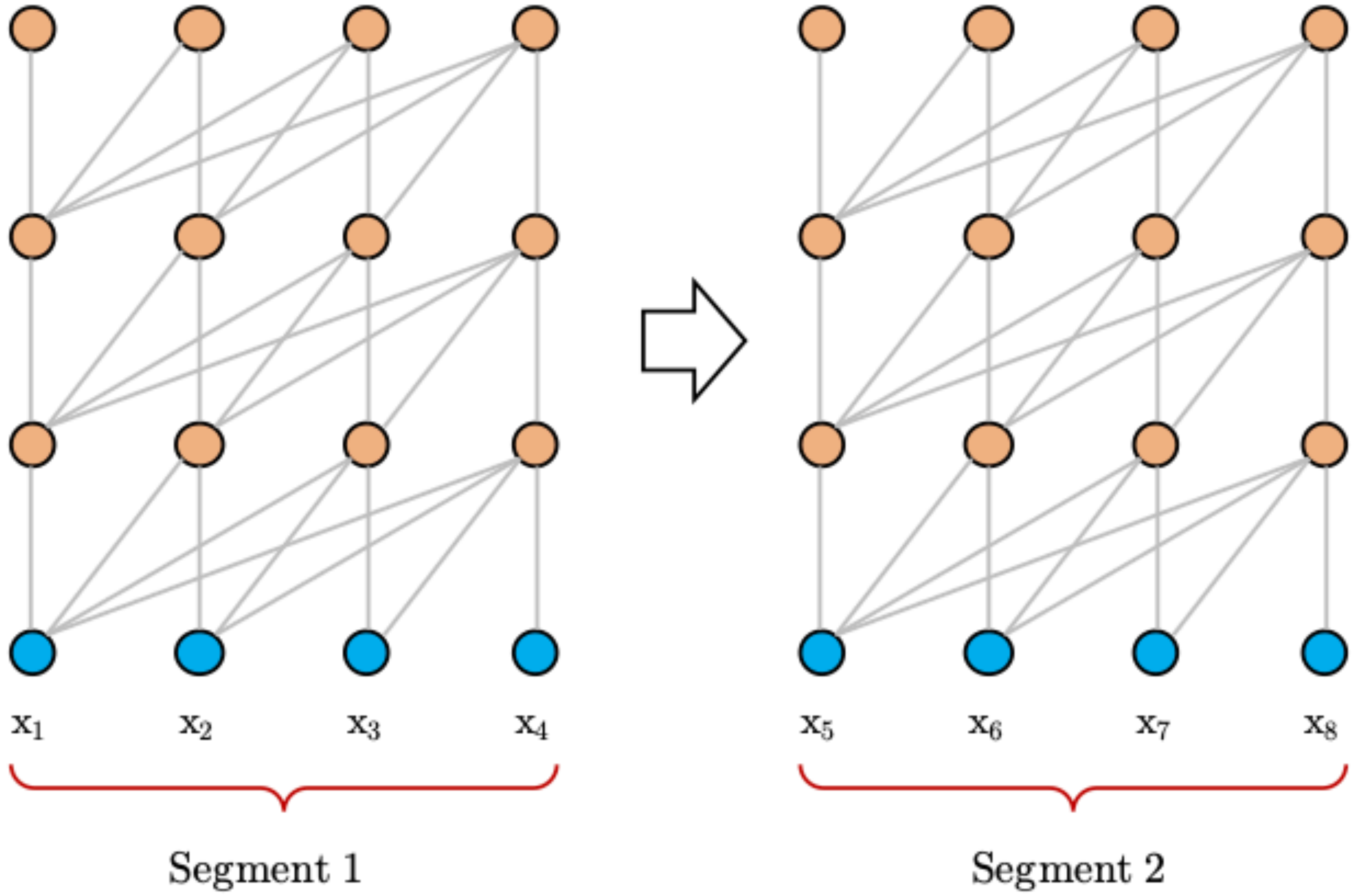
	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
$\text{BERT}_{\text{LARGE}}$	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
$\text{XLNet}_{\text{LARGE}}$	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	**90.2/90.2**	**94.7**	**92.2**	**86.6**	**96.4**	**90.9**	**68.0**	**92.4**	**91.3**	-

Transformer-XL

https://arxiv.org/abs/1901.02860

Dai+ 2019

- Vanilla Model



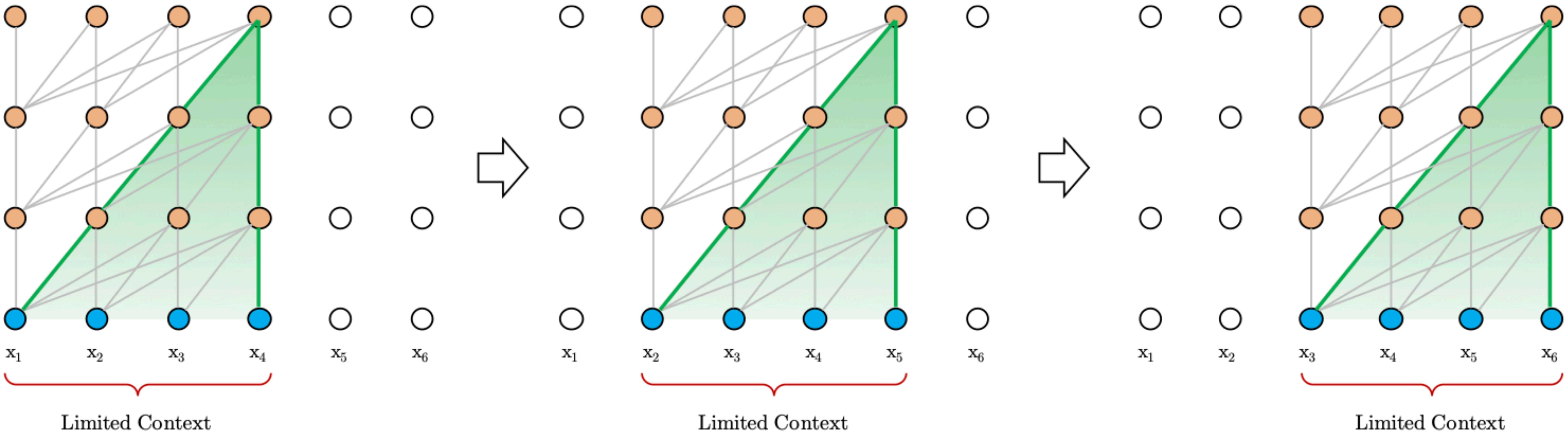
(a) Train phase.

Transformer-XL

https://arxiv.org/abs/1901.02860

Dai+ 2019

- Vanilla Model



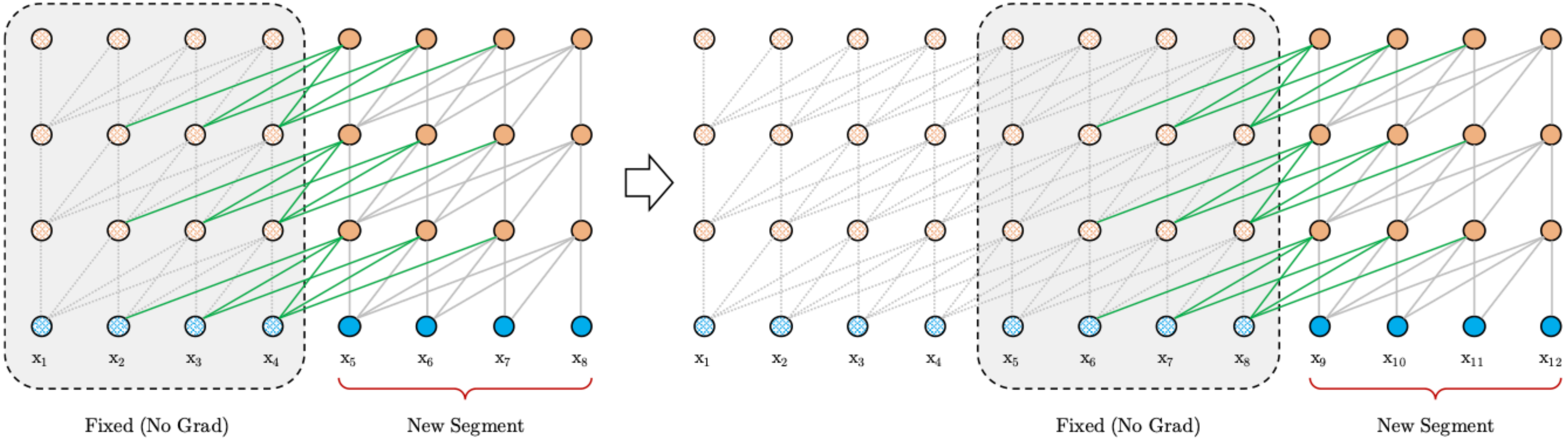
(b) Evaluation phase.

Transformer-XL

https://arxiv.org/abs/1901.02860

Dai+ 2019

- Autoregressive LM (different from GPT)
- segment level recurrence (reuse states) + relative positional embeddings



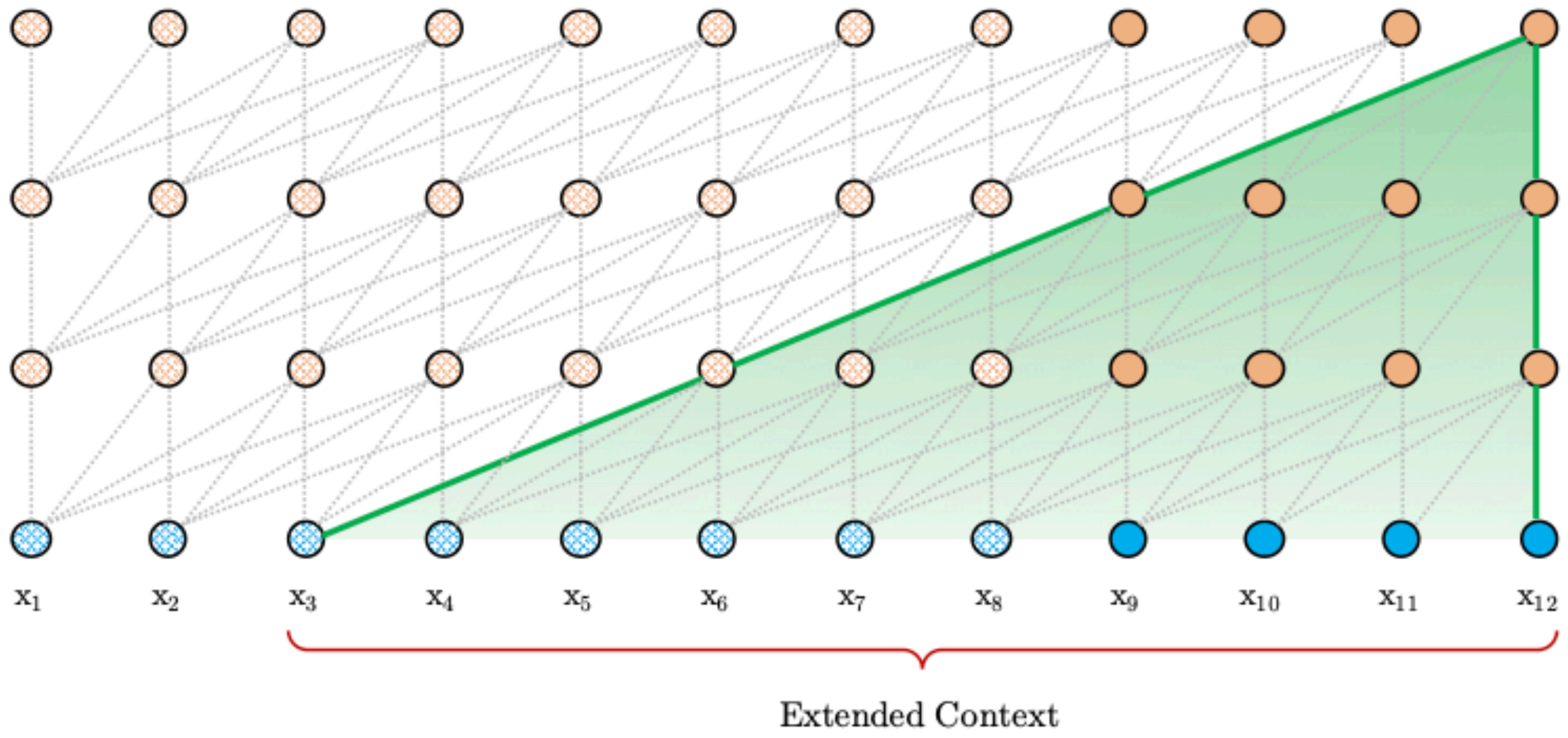
(a) Training phase.

Transformer-XL

https://arxiv.org/abs/1901.02860

Dai+ 2019

- Autoregressive LM (different from GPT)



(b) Evaluation phase.

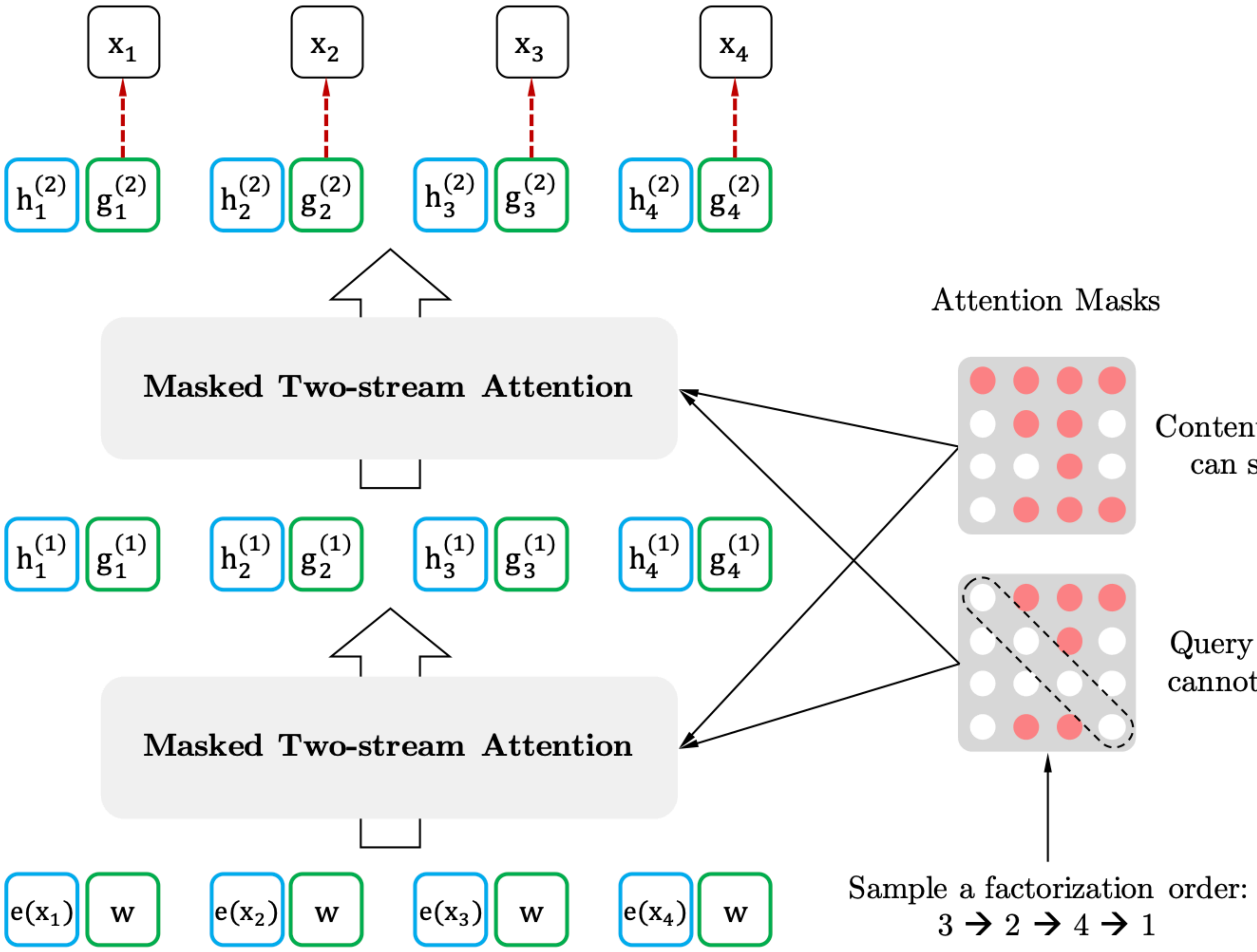
XLNet

https://arxiv.org/abs/1906.08237

Yang+ 2019

- Relative position embeddings (using auto-regressive TransformerXL)
  - Absolute attention: position 4 → 5; position 128 → 129
  - Relative attention: position $t \rightarrow (t-1)$
- Mask prediction over all token positions using permutation on factorization order (sample a factorization order: 3 → 2 → 1 → 4)
  - Two stream self-attention: standard and query on [MASK] token
  - Permute only factorization order, not sequence order

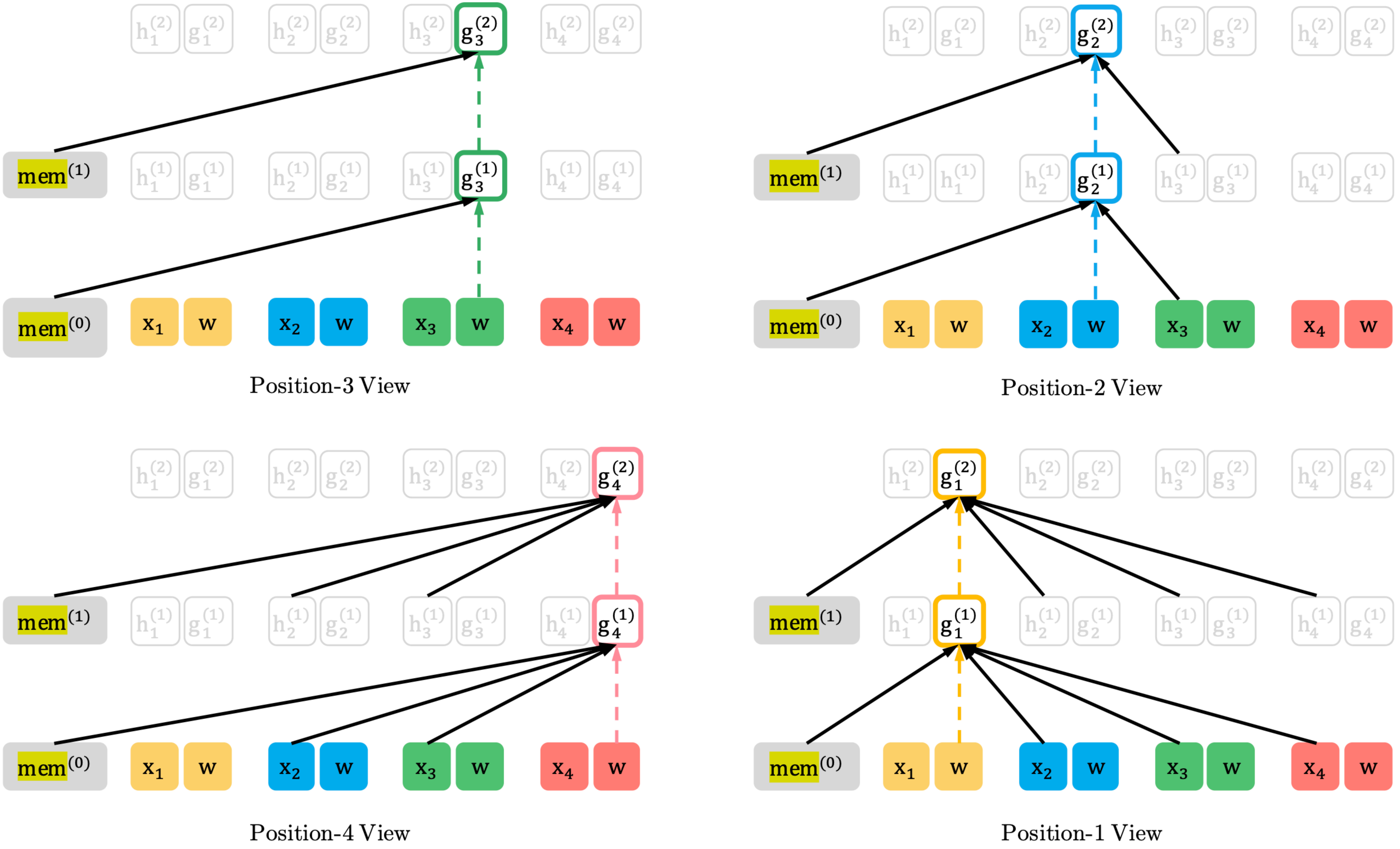
XLNet



x_1
x_2
x_3
x_4
$h_1^{(2)}$ $g_1^{(2)}$ $h_2^{(2)}$ $g_2^{(2)}$ $h_3^{(2)}$ $g_3^{(2)}$ $h_4^{(2)}$ $g_4^{(2)}$
Attention Masks
Masked Two-stream Attention
Content stream:
can see self
$h_1^{(1)}$ $g_1^{(1)}$ $h_2^{(1)}$ $g_2^{(1)}$ $h_3^{(1)}$ $g_3^{(1)}$ $h_4^{(1)}$ $g_4^{(1)}$
Query stream:
cannot see self
Masked Two-stream Attention
$e(x_1)$ w $e(x_2)$ w $e(x_3)$ w $e(x_4)$ w
Sample a factorization order:
$3 \rightarrow 2 \rightarrow 4 \rightarrow 1$

XLNet

Split View of the Query Stream
(Factorization order: 3 → 2 → 4 → 1)



XLNet

Model	MNLI	QNLI	QQP	RTE	SST-2	MRPC	CoLA	STS-B
Single-task single models on dev								
BERT [2]	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0
RoBERTa [21]	90.2/90.2	94.7	92.2	**86.6**	96.4	**90.9**	68.0	92.4
XLNet	**90.8/90.8**	**94.9**	**92.3**	85.9	**97.0**	90.8	**69.0**	**92.5**

ALBERT

https://arxiv.org/abs/1909.11942

Lan+ 2019

- Factorized embedding parameterization
  - Use small embedding size (128) and project to Transformer hidden size (1024) using a parameter matrix

ALBERT https://arxiv.org/abs/1909.11942

- Cross-layer parameter sharing
  - $h^{\ell+1}$ parameters are shared with h^{ℓ}

Models	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS
Single-task single models on dev								
BERT-large	86.6	92.3	91.3	70.4	93.2	88.0	60.6	90.0
XLNet-large	89.8	93.9	91.8	83.8	95.6	89.2	63.6	91.8
RoBERTa-large	90.2	94.7	**92.2**	86.6	96.4	**90.9**	68.0	92.4
ALBERT (1M)	90.4	95.2	92.0	88.1	96.8	90.2	68.7	92.7
ALBERT (1.5M)	**90.8**	**95.3**	**92.2**	**89.2**	**96.9**	**90.9**	**71.4**	**93.0**

ALBERT

https://arxiv.org/abs/1909.11942

- Light on parameters; not necessarily faster than BERT

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3	4.7x
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2	1.0
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1	5.6x
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4	1.7x
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5	0.6x
	xxlarge	235M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	0.3x

T5

https://arxiv.org/abs/1910.10683

Raffel+ 2019

- Ablation study on many aspects of pre-training and fine-tuning
    - Model size (bigger is better; 11B parameters)
    - Amount of training data (more is better)
    - Domain / cleanliness of training data [-ve]
    - Pre-training objective (e.g. span length of masked text) [-ve]
    - Ensemble models [-ve]
    - Fine-tuning recipe (e.g. only allow top k layers to fine-tune) [-ve]
    - Multi-task training [-ve]

Table	Experiment	GLUE													CNN/DM			SQuAD		SuperGLUE												WMT		
		Score Average	CoLA MCC	SST-2 Acc	MRPC F1	MRPC Acc	STSB PCC	STSB SCC	QQP F1	QQP Acc	MNLI_m Acc	MNLI_mm Acc	QNLI Acc	RTE Acc	R-1-F	R-2-F	R-L-F	EM	F1	Score Average	BoolQ Acc	CB F1	CB Acc	COPA Acc	MultiRC F1	MultiRC EM	ReCoRD F1	ReCoRD EM	RTE Acc	WiC Acc	WSC Acc	EnDe BLEU	EnFr BLEU	EnRo BLEU
1	★ Baseline average	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
1	Baseline standard deviation	0.235	1.111	0.569	0.729	1.019	0.374	0.418	0.108	0.070	0.291	0.231	0.361	1.393	0.065	0.065	0.058	0.343	0.226	0.416	0.365	3.237	2.560	2.741	0.716	1.011	0.370	0.379	1.228	0.850	2.029	0.112	0.090	0.108
1	No pre-training	66.22	12.29	80.62	81.42	73.04	72.58	72.97	81.94	86.62	68.02	67.98	75.69	58.84	39.19	17.60	36.69	50.31	61.97	53.04	65.38	71.61	76.79	62.00	59.10	0.84	20.33	17.95	54.15	54.08	65.38	25.86	39.77	24.04
2	★ Enc/dec, denoising	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
2																																		27.46
2																																		26.95
2																																		25.86
2																																		27.39
2																																		26.86
2																																		27.05
2																																		26.89
2																																		25.38
2																																		26.76
4																																		27.49
4																																		27.41
4																																		25.62
5																																		27.41
5																																		27.55
5	★																																	27.65
5																																		27.82
6																																		27.44
6	★																																	27.65
6																																		27.47
6																																		27.49
7	★																																	27.65
7																																		27.63
7																																		27.62
7																																		27.53
7																																		27.69
8	★																																	27.65
8																																		27.21
8																																		27.48
8																																		27.59
8																																		27.67
8																																		27.57
9	★																																	27.65
9																																		27.63
9																																		27.33
9																																		26.80
9																																		25.81
10	★																																	27.65
10																																		15.54
10																																		22.63
10																																		25.81
10																																		26.93
10																																		26.93
11	★																																	27.65
11																																		26.78
11																																		27.10
11																																		27.25
11																																		27.39
11																																		27.76
11																																		27.68
11																																		27.13
11																																		27.20
11																																		27.45
11																																		27.17
12	★																																	27.65
12																																		27.76
12																																		28.07
12																																		27.87
12	Supervised multi-task pre-training	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	28.04
13	★ Baseline	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
13	1× size, 4× training steps	85.33	60.29	93.81	94.06	91.67	89.42	89.25	89.15	91.87	86.01	85.70	91.63	78.34	41.52	19.33	38.96	82.45	90.19	74.72	79.17	94.75	92.86	71.00	67.34	29.70	72.63	71.59	78.34	72.10	82.69	27.08	40.66	27.93
13	1× size, 4× batch size	84.60	56.08	93.12	92.31	89.22	88.85	88.84	89.35	92.07	85.98	86.13	91.07	80.14	41.70	19.42	39.08	82.52	90.21	74.64	78.78	93.69	94.64	72.00	68.09	30.95	74.73	73.90	76.53	70.06	81.73	27.07	40.60	27.84
13	2× size, 2× training steps	86.18	62.04	93.69	93.36	90.69	89.18	89.23	89.35	92.05	87.23	87.05	92.68	81.95	41.74	19.66	39.14	84.18	91.29	77.18	80.98	97.36	96.43	74.00	71.34	35.68	77.11	76.34	80.51	69.28	85.58	27.52	41.03	28.19
13	4× size, 1× training steps	85.91	57.58	94.38	92.67	89.95	89.60	89.60	89.44	92.14	87.05	87.12	93.12	83.39	41.60	19.73	39.08	83.86	91.32	78.04	81.38	89.09	94.64	73.00	73.74	40.40	78.25	77.40	81.59	70.22	91.35	27.47	40.71	28.10
13	4× ensembled	84.77	56.14	93.46	93.31	90.67	89.71	89.60	89.62	92.24	86.22	86.53	91.60	77.98	42.10	20.10	39.56	83.09	90.40	71.74	77.58	89.85	91.07	66.00	69.32	29.49	72.67	71.94	76.90	69.12	72.12	28.05	40.53	28.09
13	4× ensembled, fine-tune only	84.05	54.78	92.78	93.15	90.44	88.34	88.12	89.27	91.97	85.33	85.88	90.98	77.62	41.66	19.57	39.12	82.36	89.86	71.56	77.43	90.07	92.86	69.00	67.31	26.34	70.47	69.64	75.45	68.18	74.04	27.55	40.22	28.09

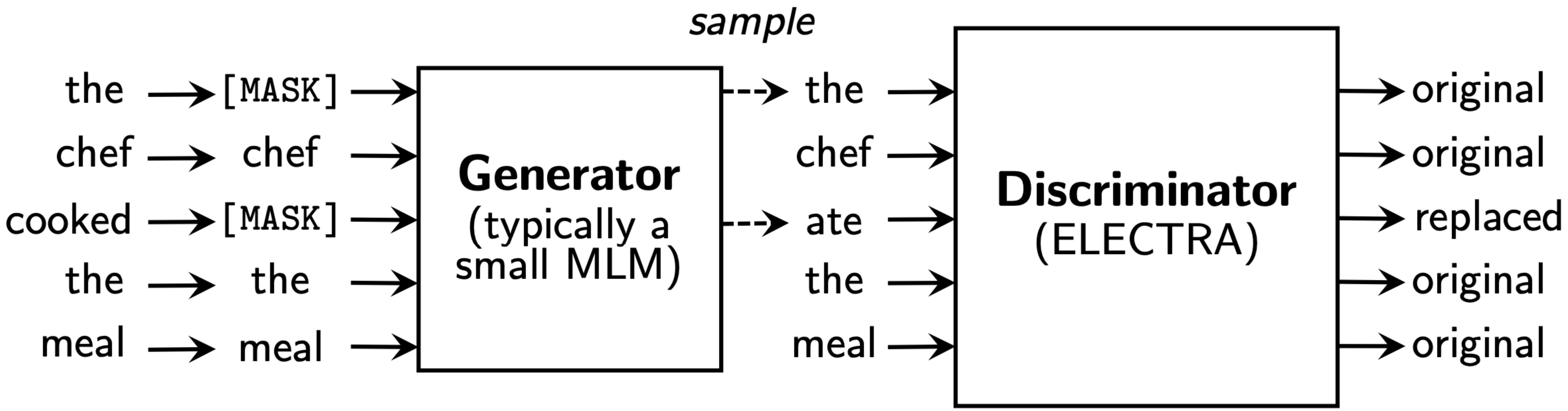
Model	GLUE Average	CoLA Matthew's	SST-2 Accuracy	MRPC F1	MRPC Accuracy	STS-B Pearson	STS-B Spearman
Previous best	89.4[a]	69.2[b]	97.1[a]	**93.6**[b]	**91.5**[b]	92.7[b]	92.3[b]
T5-Small	77.4	41.0	91.8	89.7	86.6	85.6	85.0
T5-Base	82.7	51.1	95.2	90.7	87.5	89.4	88.6
T5-Large	86.4	61.2	96.3	92.4	89.9	89.9	89.2
T5-3B	88.5	67.1	97.4	92.5	90.0	90.6	89.8
T5-11B	**90.3**	**71.6**	**97.5**	92.8	90.4	**93.1**	**92.8**

Model	QQP F1	QQP Accuracy	MNLI-m Accuracy	MNLI-mm Accuracy	QNLI Accuracy	RTE Accuracy	WNLI Accuracy
Previous best	74.8[c]	**90.7**[b]	91.3[a]	91.0[a]	**99.2**[a]	89.2[a]	91.8[a]
T5-Small	70.0	88.0	82.4	82.3	90.3	69.9	69.2
T5-Base	72.6	89.4	87.1	86.2	93.7	80.1	78.8
T5-Large	73.9	89.9	89.9	89.6	94.8	87.2	85.6
T5-3B	74.4	89.7	91.4	91.2	96.3	91.1	89.7
T5-11B	**75.1**	90.6	**92.2**	**91.9**	96.9	**92.8**	**94.5**

ELECTRA https://arxiv.org/abs/2003.10555

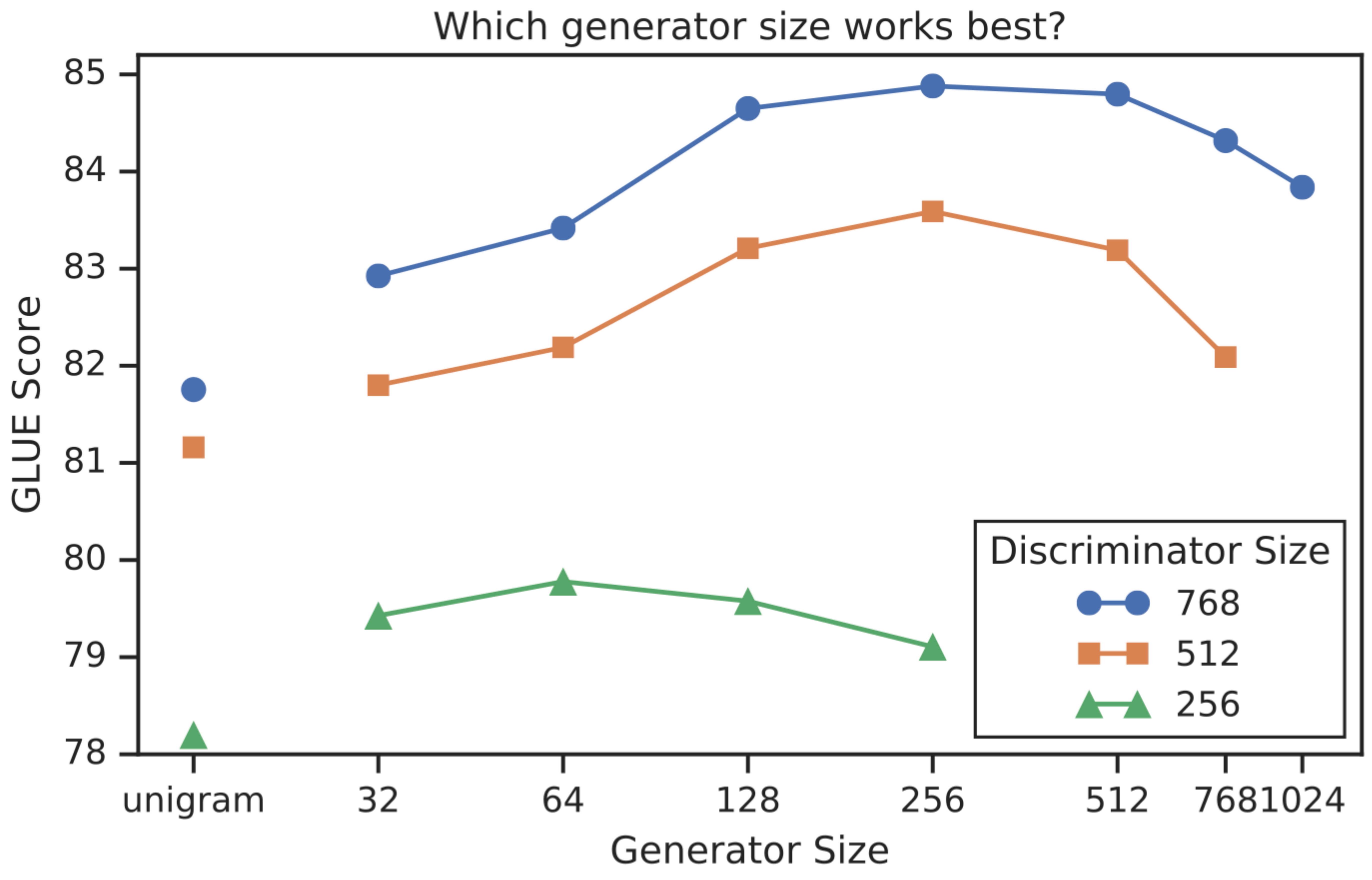
Clark+ 2020

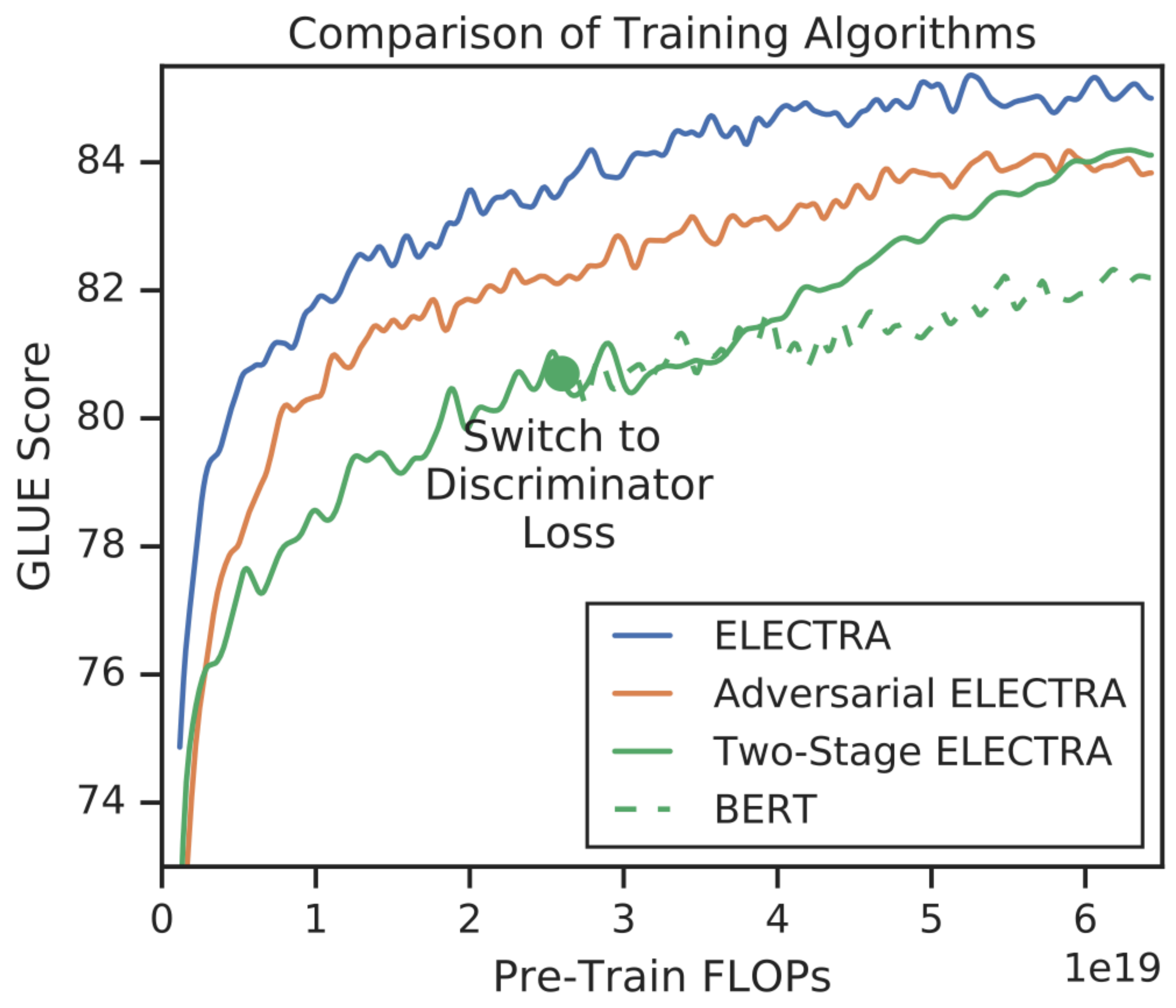
- Train model to discriminate locally plausible text from real text



ELECTRA

https://arxiv.org/abs/2003.10555





ELECTRA

https://arxiv.org/abs/2003.10555

Model	Train FLOPs	Params	CoLA	SST	MRPC	STS	QQP	MNLI	QNLI	RTE	Avg.
BERT	1.9e20 (0.27x)	335M	60.6	93.2	88.0	90.0	91.3	86.6	92.3	70.4	84.0
RoBERTa-100K	6.4e20 (0.90x)	356M	66.1	95.6	**91.4**	92.2	92.0	89.3	94.0	82.7	87.9
RoBERTa-500K	3.2e21 (4.5x)	356M	68.0	96.4	90.9	92.1	92.2	90.2	94.7	86.6	88.9
XLNet	3.9e21 (5.4x)	360M	69.0	**97.0**	90.8	92.2	92.3	90.8	94.9	85.9	89.1
BERT (ours)	7.1e20 (1x)	335M	67.0	95.9	89.1	91.2	91.5	89.6	93.5	79.5	87.2
ELECTRA-400K	7.1e20 (1x)	335M	**69.3**	96.0	90.6	92.1	**92.4**	90.5	94.5	86.8	89.0
ELECTRA-1.75M	3.1e21 (4.4x)	335M	69.1	96.9	90.8	**92.6**	**92.4**	**90.9**	**95.0**	**88.0**	**89.5**

Other BERT Extensions

- Many many extensions to BERT; too many to cover here; mostly pre-training
  - Auto-regressive BERT variants (BART; XLM; etc.)
  - SpanBERT; Entity-based BERT (LUKE; SpanLinkBERT)
  - Mainly training on more data, or different data, slight variants (Megatron)
- Efficient fine-tuning (covered separately)
- Efficient inference
  - Distillation of BERT models (covered separately)