



# Natural Language Processing; Language Modeling



[YouTube Playlist](#)

**Maziar Raissi**

**Assistant Professor**

Department of Applied Mathematics

University of Colorado Boulder

[maziar.raissi@colorado.edu](mailto:maziar.raissi@colorado.edu)



Boulder



[YouTube Video](#)

# Deep contextualized word representations

**ELMo:** Embeddings from Language Models  
 $(t_1, t_2, \dots, t_N) \rightarrow$  sequence of  $N$  tokens

**Forward Language Model**

$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_1, t_2, \dots, t_{k-1})$$

$x_k^{LM} \rightarrow$  context-independent token representation

$x_k^{LM} \triangleright L$  layers of forward LSTMs

$$\overrightarrow{h}_{k,j}^{LM}, j = 1, \dots, L$$

$\overrightarrow{h}_{k,L}^{LM} \triangleright$  softmax  $\rightarrow$  predict next token  $t_{k+1}$

**Backward LM**

$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_{k+1}, t_{k+2}, \dots, t_N)$$

$$\overleftarrow{h}_{k,j}^{LM}, j = 1, \dots, L$$

**Bidirectional LM**

$$\sum_{k=1}^N (\log p(t_k | t_1, \dots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k | t_{k+1}, \dots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s))$$

**ELMo**

$$\begin{aligned} R_k &= \{x_k^{LM}, \overrightarrow{h}_{k,j}^{LM}, \overleftarrow{h}_{k,j}^{LM} \mid j = 1, \dots, L\} \\ &= \{h_{k,j}^{LM} \mid j = 0, \dots, L\}, \end{aligned}$$

$$\begin{aligned} h_{k,0}^{LM} &= x_k^{LM} \\ h_{k,j}^{LM} &= [\overrightarrow{h}_{k,j}^{LM}; \overleftarrow{h}_{k,j}^{LM}] \\ \text{ELMo}_k &= E(R_k; \Theta_e) \\ E(R_k) &= h_{k,L}^{LM} \\ \text{ELMo}_k^{\text{task}} &= E(R_k; \Theta^{\text{task}}) = \gamma^{\text{task}} \sum_{j=0}^L s_j^{\text{task}} h_{k,j}^{LM} \\ s_j^{\text{task}} &\rightarrow \text{softmax-normalized weights} \\ [x_k; \text{ELMo}_k^{\text{task}}] &\rightarrow \text{input to task RNN} \\ [h_k; \text{ELMo}_k^{\text{task}}] &\rightarrow \text{output of task RNN} \end{aligned}$$

- |   |                         |
|---|-------------------------|
| <ol style="list-style-type: none"> <li>1. Question Answering</li> <li>2. Textual Entailment</li> <li>3. Semantic Role Labeling</li> <li>4. Coreference Resolution</li> <li>5. Named Entity Extraction</li> <li>6. Sentiment Analysis</li> </ol> | <b>Downstream Tasks</b> |
|---|-------------------------|

TASK	PREVIOUS SOTA	OUR BASELINE		INCREASE (ABSOLUTE/RELATIVE)
		ELMo + BASELINE	ELMo + RELATIVE	
Stanford Question Answering Dataset $\leftarrow$ SQuAD	Liu et al. (2017) F <sub>1</sub> 84.4	81.1	85.8	4.7 / 24.9%
Stanford Natural Language Inference $\leftarrow$ SNLI	Chen et al. (2017) accuracy 88.6	88.0	88.7 $\pm$ 0.17	0.7 / 5.8%
Semantic Role Labeling $\leftarrow$ SRL	He et al. (2017) F <sub>1</sub> 81.7	81.4	84.6	3.2 / 17.2%
Coreference Resolution $\leftarrow$ Coref	Lee et al. (2017) average F <sub>1</sub> 67.2	67.2	70.4	3.2 / 9.8%
Named Entity Recognition $\leftarrow$ NER	Peters et al. (2017) F <sub>1</sub> 91.93 $\pm$ 0.19	90.15	92.22 $\pm$ 0.10	2.06 / 21%
Stanford Sentiment Treebank $\leftarrow$ SST-5	McCann et al. (2017) accuracy 53.7	51.4	54.7 $\pm$ 0.5	3.3 / 6.8%

Source		Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder {...}	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway play for Garson {...}	{...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

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

## Sequence Modeling

$x_0, \dots, x_T \rightarrow$  input sequence

$y_0, \dots, y_T \rightarrow$  output sequence

$f : \mathcal{X}^{T+1} \rightarrow \mathcal{Y}^{T+1}$

sequence modeling network

$\hat{y}_0, \dots, \hat{y}_T = f(x_0, \dots, x_T)$

## Causal Constraint

$y_t$  depends only on  $x_0, \dots, x_t$  and not on any “future” inputs  $x_{t+1}, \dots, x_T$

$L(y_0, \dots, y_T, f(x_0, \dots, x_T)) \rightarrow$  loss

1D fully-convolutional network (FCN)  
zero padding

causal convolutions

TCN = 1D FCN + causal convolutions

temporal convolutional network

## Dilated Convolutions

longer history

$x \in \mathbb{R}^n \rightarrow$  1D sequence input

$f : \{0, \dots, k-1\} \rightarrow \mathbb{R}$

filter

$$F(s) = (x *_d f)(s) = \sum_{i=0}^{k-1} f(i)x_{s-di}$$

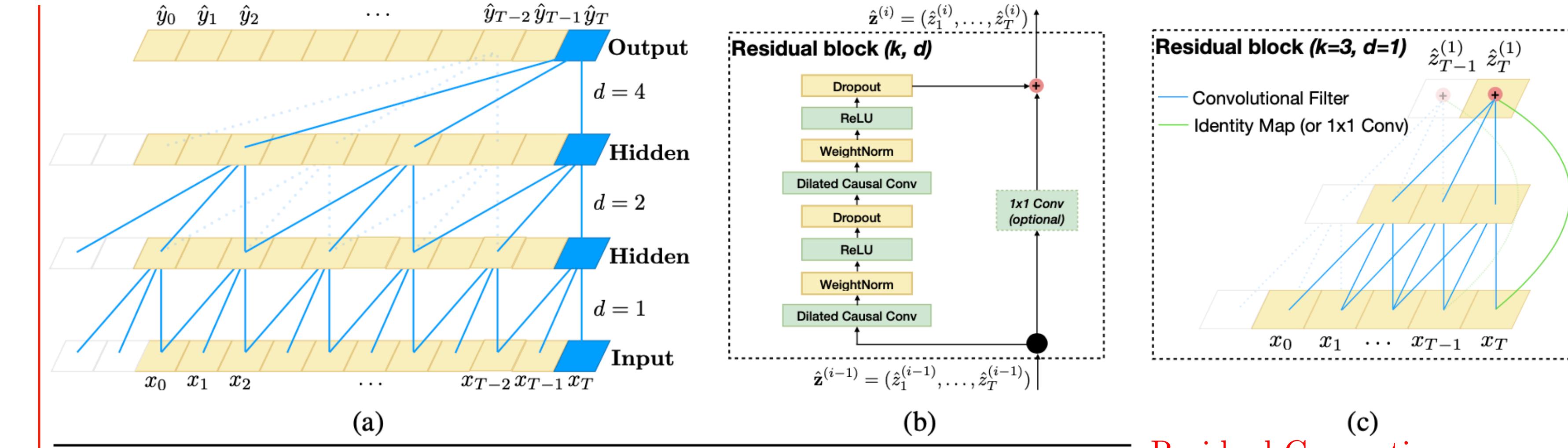
$d \rightarrow$  dilation factor

$k \rightarrow$  filter size

$s - di \rightarrow$  direction of the past

$(k-1)d \rightarrow$  effective history of such a layer

$d = O(2^i)$  at level  $i$  of the network

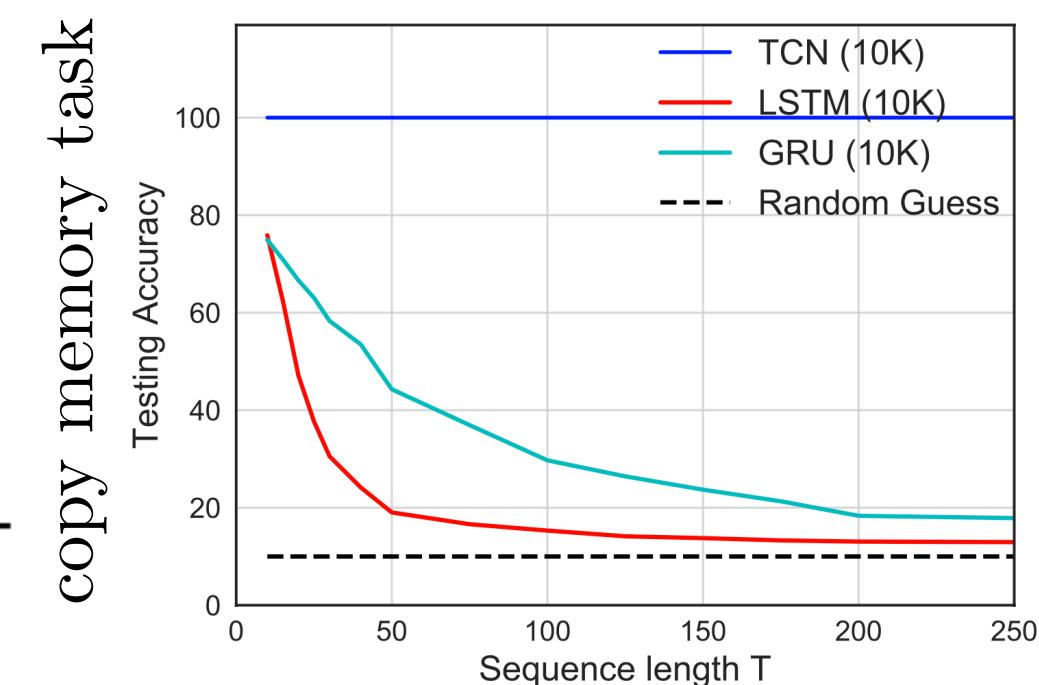


Sequence Modeling Task	Model Size ( $\approx$ )	Models			
		LSTM	GRU	RNN	TCN
Seq. MNIST (accuracy <sup>h</sup> )	70K	87.2	96.2	21.5	<b>99.0</b>
Permuted MNIST (accuracy)	70K	85.7	87.3	25.3	<b>97.2</b>
Adding problem $T=600$ (loss <sup>ℓ</sup> )	70K	0.164	<b>5.3e-5</b>	0.177	<b>5.8e-5</b>
Copy memory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

## Residual Connections

$$o = \text{Activation}(x + \mathcal{F}(x))$$

**JSB Chorales:** Each input is a sequence of elements. Each element is an 88-bit binary code that corresponds to the 88 keys on a piano, with 1 indicating a key that is pressed at a given time.





Boulder

# Improving Language Understanding by Generative Pre-Training



[YouTube Video](#)

## Unsupervised pre-training

$$\mathcal{U} = \{u_1, u_2, \dots, u_n\} \rightarrow \text{unsupervised corpus of tokens}$$

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

size of the context window

Transformer decoder

$$U = (u_{-k}, \dots, u_{-1}) \rightarrow \text{context vector of tokens}$$

$$h_0 = U W_e + W_p$$

position embedding matrix

token embedding matrix

$$h_l = \text{transformer\_block}(h_{l-1}), l = 1, \dots, L$$

number of layers

$$P(u) = \text{softmax}(h_L W_e^T)$$

## Supervised fine-tuning

$\mathcal{C} \rightarrow \text{labeled dataset}$

$x^1, \dots, x^m \rightarrow \text{input tokens}$

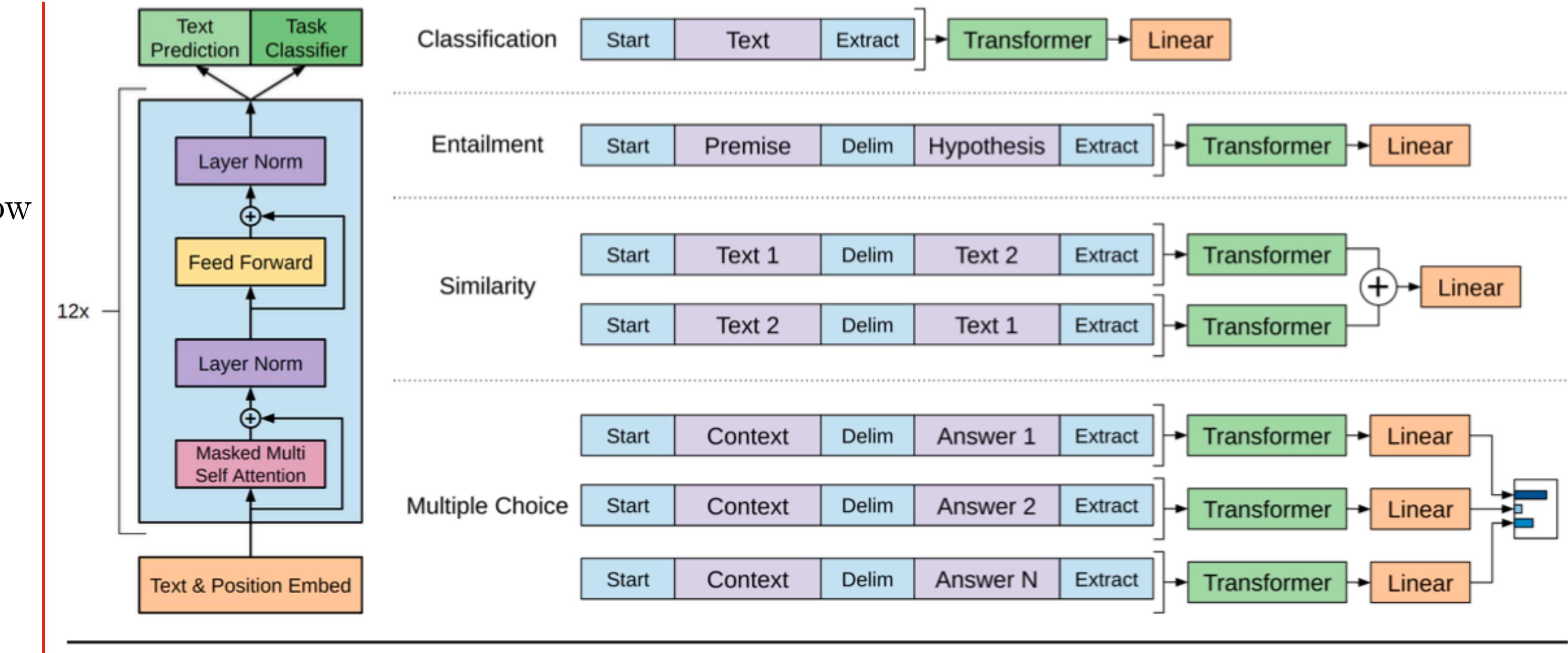
$y \rightarrow \text{label}$

$h_L^m \rightarrow \text{final transformer block's activation}$

$$p(y|x^1, \dots, x^m) = \text{softmax}(h_L^m W_y)$$

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

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



## Task

## Datasets

Natural language inference

SNLI [5], MultiNLI [66], Question NLI [64], RTE [4], SciTail [25]

Question Answering

RACE [30], Story Cloze [40]

Sentence similarity

MSR Paraphrase Corpus [14], Quora Question Pairs [9], STS Benchmark [6]

Classification

Stanford Sentiment Treebank-2 [54], CoLA [65]

**Natural Language Inference** entailment, contradiction or neutral

image captions (SNLI)

transcribed speech, popular fiction, and government reports (MNLI)

Wikipedia articles (QNLI)

Corpus of Linguistic Acceptability (CoLA)

science exams (SciTail)

Semantic Textual Similarity (STS-B)

news articles (RTE)

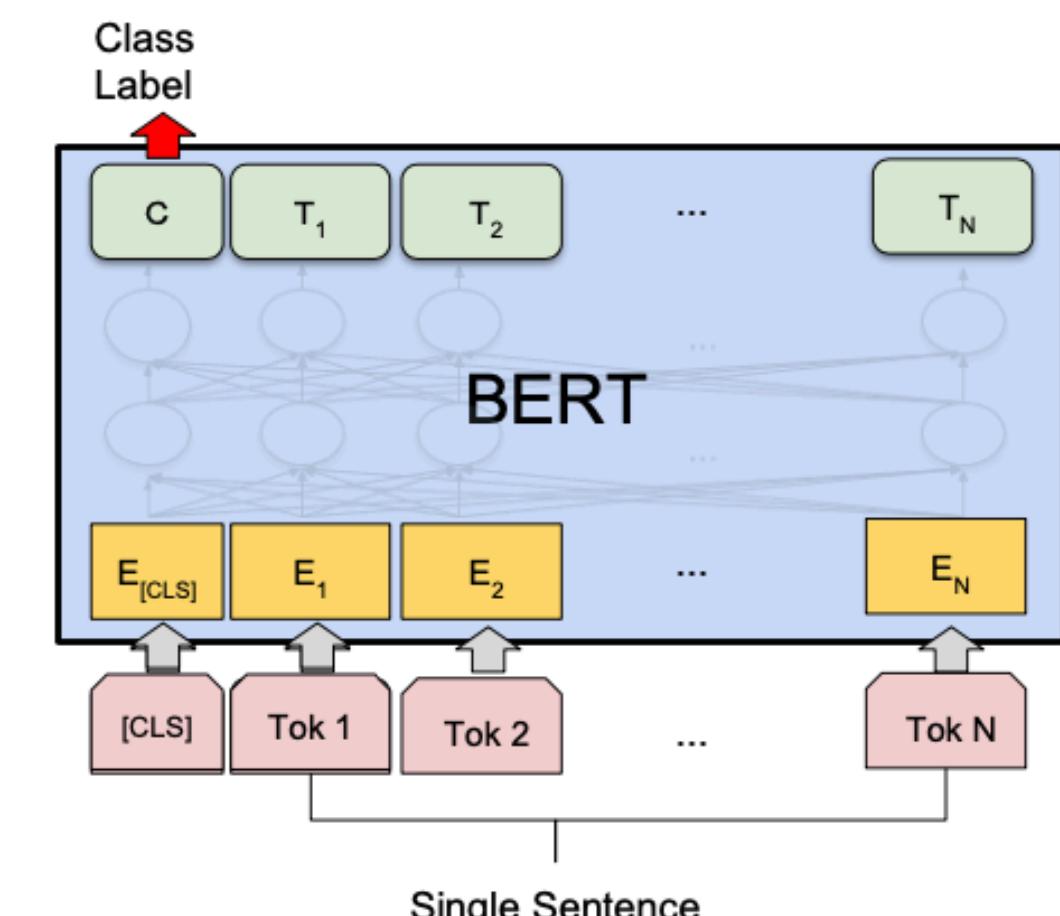
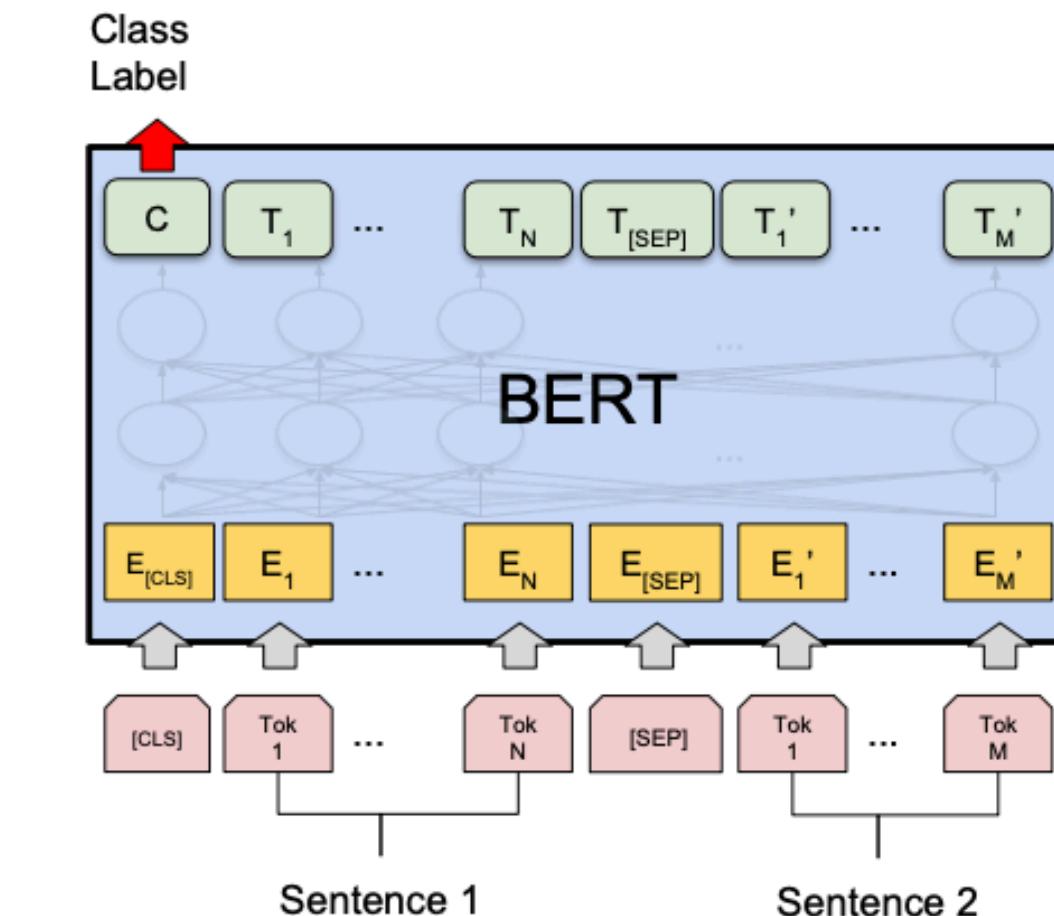
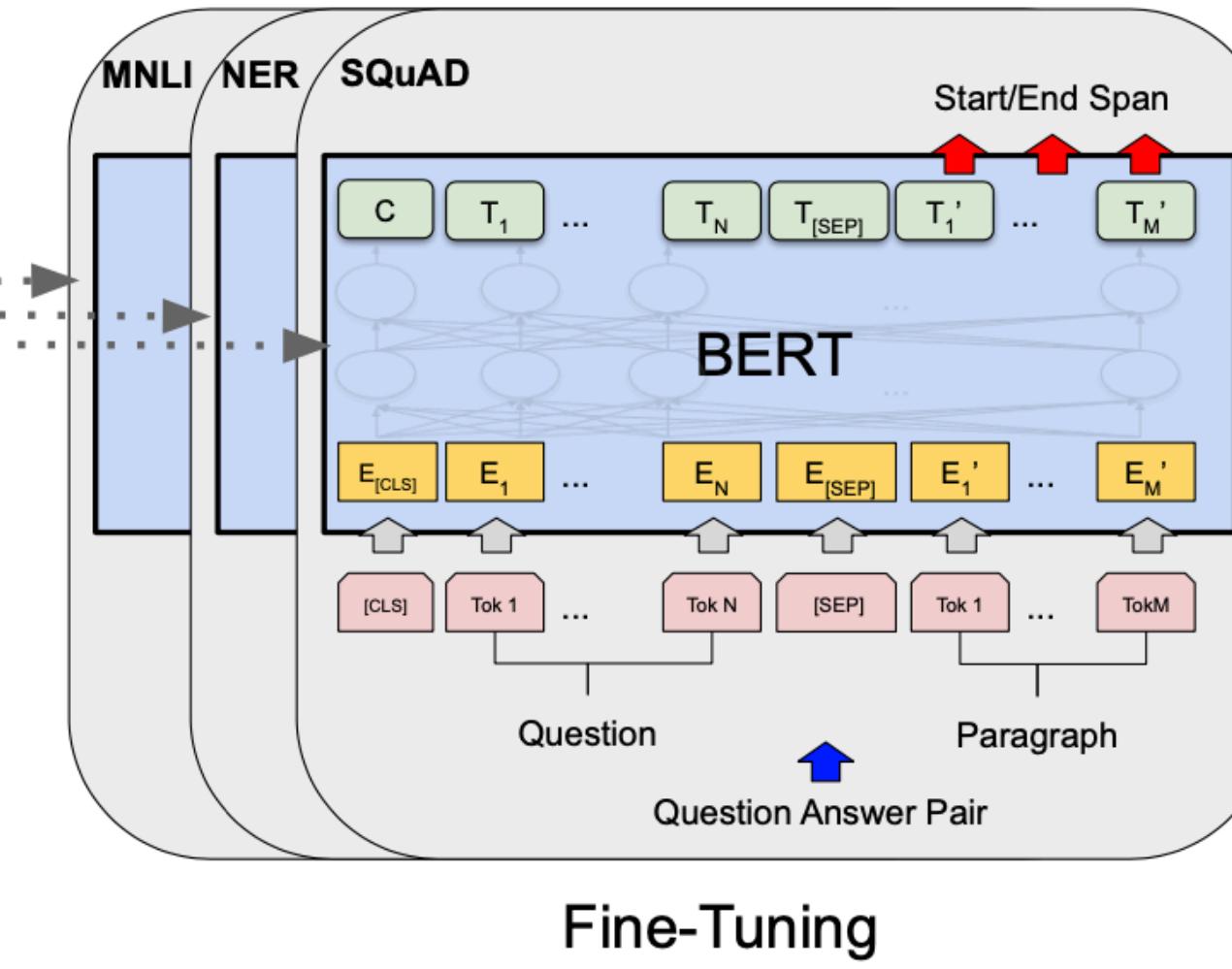
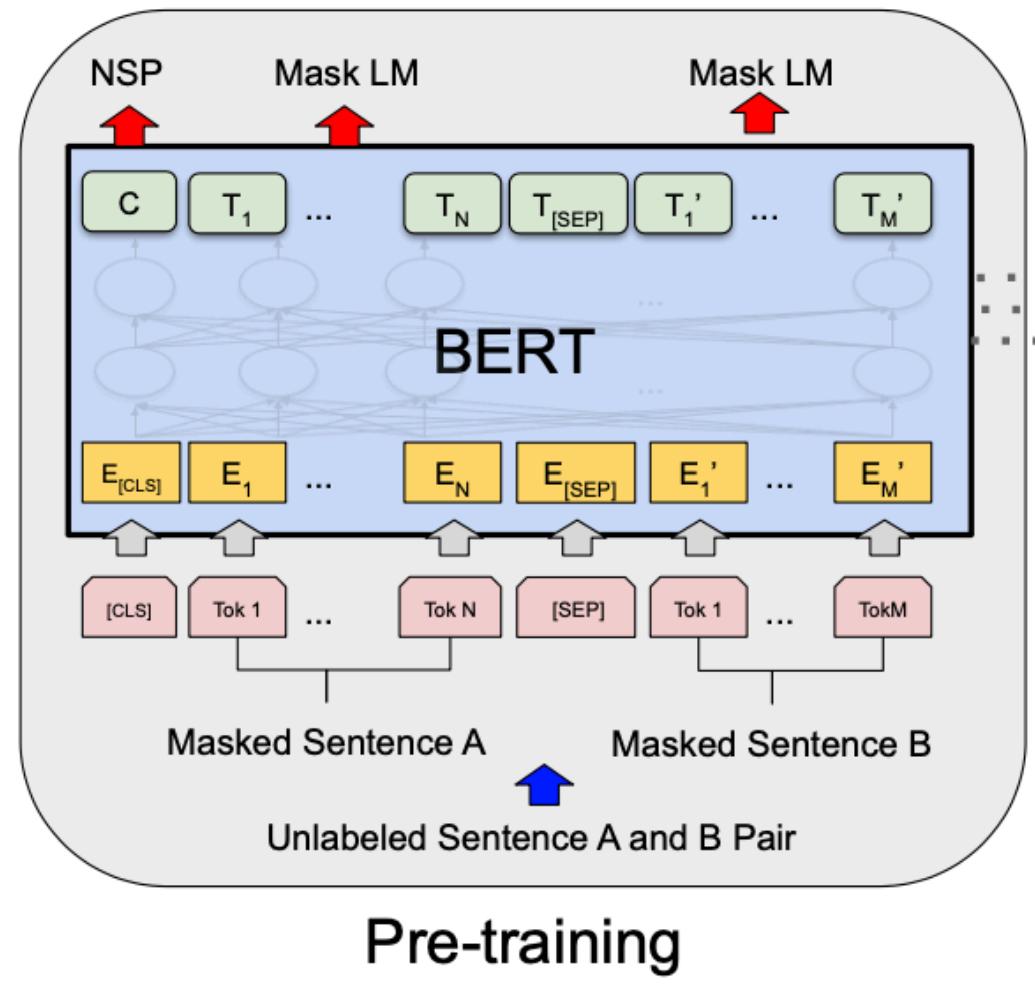


# BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding



[YouTube Playlist](#)

## Bidirectional Encoder Representations from Transformers

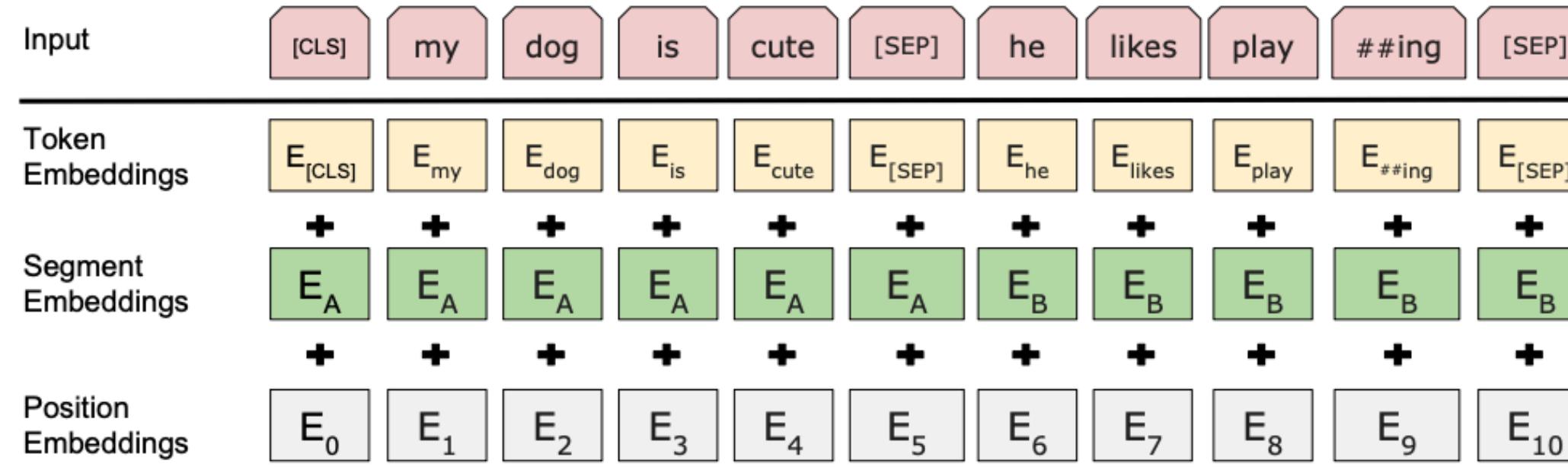


## Task 1: Masked LM

mask 15% of all WordPiece tokens in each sequence at random

## Task 2: Next Sentence Prediction (NSP)

IsNext vs NotNext



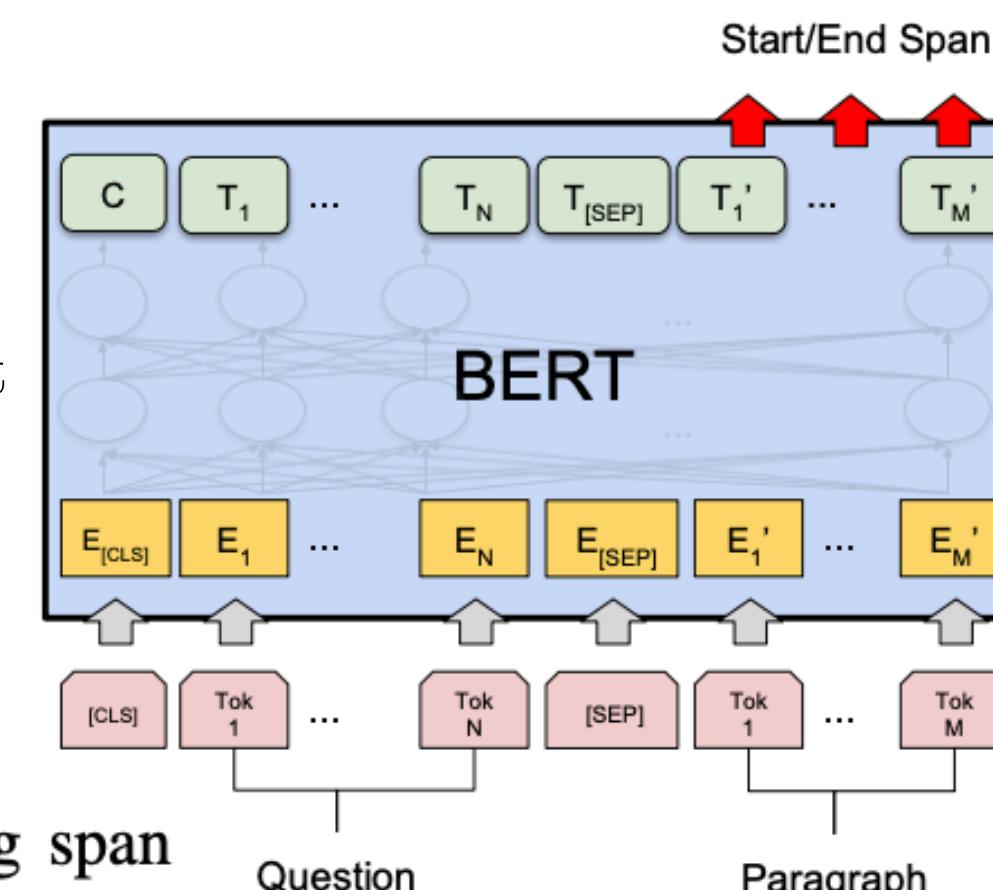
$$S \in \mathbb{R}^H \rightarrow \text{start}$$

$$E \in \mathbb{R}^H \rightarrow \text{end}$$

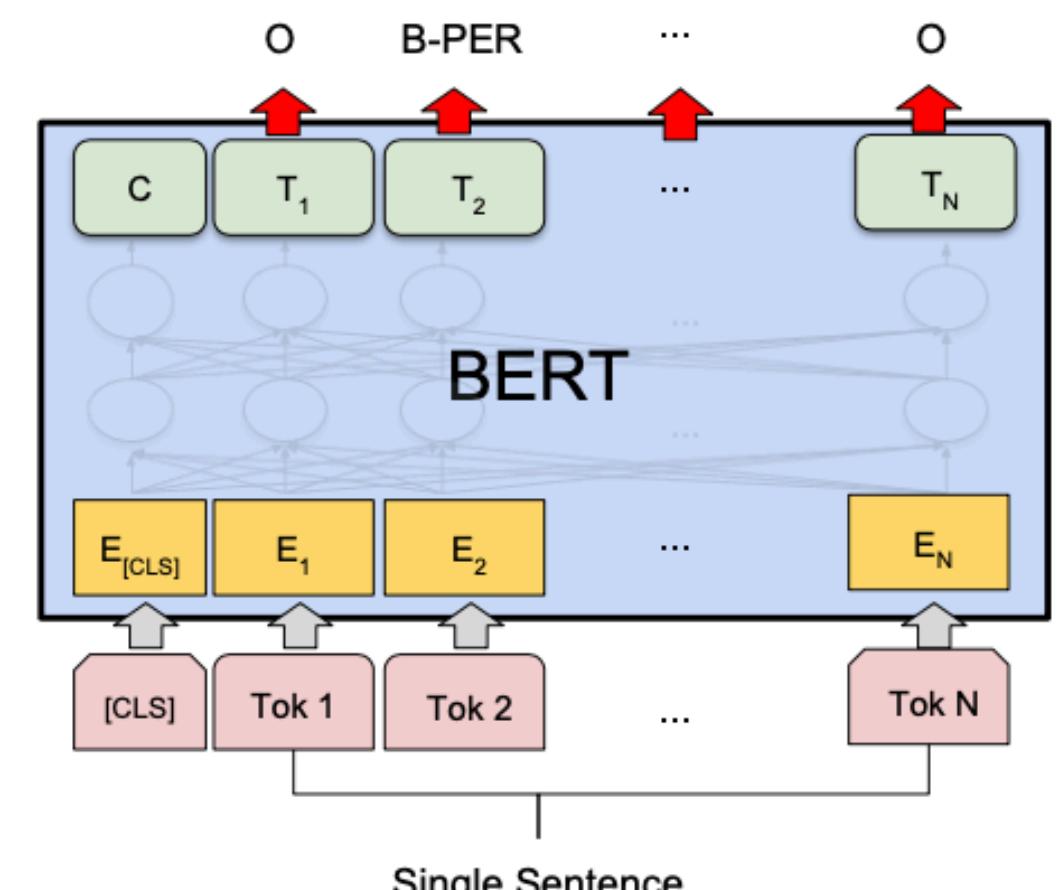
$$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$$

$$S \cdot T_i + E \cdot T_j$$

maximum scoring span



(c) Question Answering Tasks:  
SQuAD v1.1



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



Boulder

# Language Models are Unsupervised Multitask Learners



[YouTube Video](#)

- Question Answering
- Machine Translation
- Reading Comprehension
- Summarization

GPT-2 → a 1.5B parameter Transformer

$(x_1, x_2, \dots, x_n) \rightarrow$  set of examples

$x = (s_1, s_2, \dots, s_n) \rightarrow$  variable-length sequence of symbols

$$p(x) = \prod_{i=1}^n p(s_i | s_1, \dots, s_{i-1})$$

$$p(s_{n-k}, \dots, s_n | s_1, \dots, s_{n-k-1})$$

$$p(\text{output} | \text{input})$$

$$p(\text{output} | \text{input}, \text{task})$$

Task conditioning: 1. architectural level  
2. algorithmic level

(e.g., Model-Agnostic Meta-Learning)

translation training example

(translate to french, english text, french text)

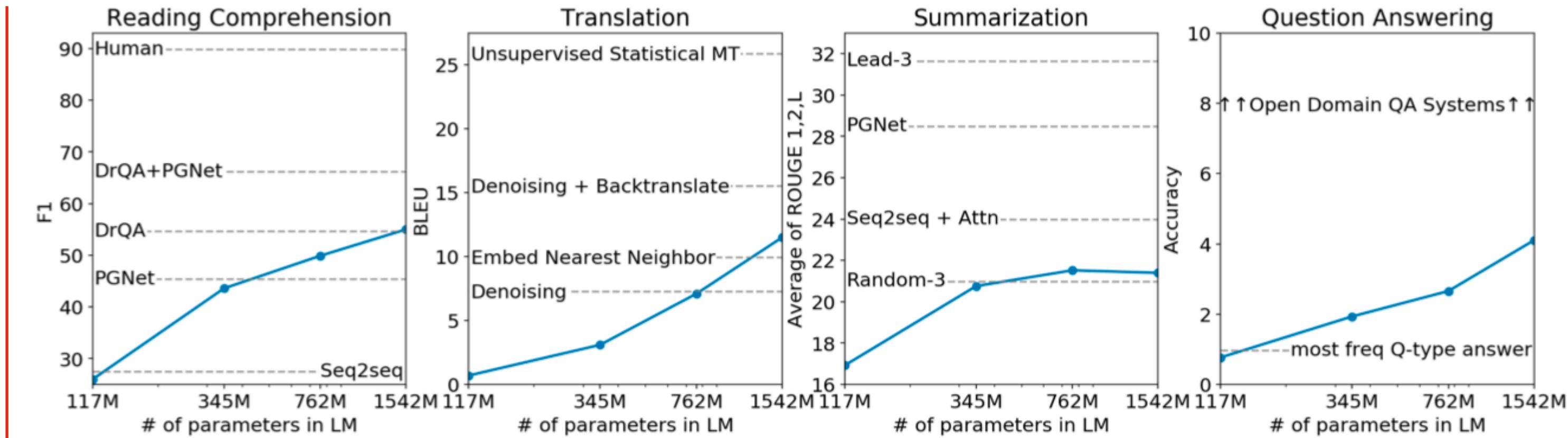
reading comprehension training example

(answer the question, document, question, answer)

pre-processing steps: lower-casing,

tokenization, and out-of-vocabulary tokens

Byte Pair Encoding (BPE): a practical middle ground between character and word level language modeling



"I'm not the cleverest man in the world, but like they say in French: **Je ne suis pas un imbecile** [I'm not a fool].

In a now-deleted post from Aug. 16, Soheil Eid, Tory candidate in the riding of Joliette, wrote in French: "**Mentez mentez, il en restera toujours quelque chose**," which translates as, "**Lie lie and something will always remain**."

"I hate the word '**perfume**'," Burr says. "It's somewhat better in French: '**parfum**'."

If listened carefully at 29:55, a conversation can be heard between two guys in French: "**-Comment on fait pour aller de l'autre côté? -Quel autre côté?**", which means "**- How do you get to the other side? - What side?**".

If this sounds like a bit of a stretch, consider this question in French: **As-tu aller au cinéma?**, or **Did you go to the movies?**, which literally translates as Have-you to go to movies/theater?

**"Brevet Sans Garantie Du Gouvernement"**, translated to English: **"Patented without government warranty"**.

**Context (passage and previous question/answer pairs)**

Tom goes everywhere with Catherine Green, a 54-year-old secretary. He moves around her office at work and goes shopping with her. "Most people don't seem to mind Tom," says Catherine, who thinks he is wonderful. "He's my fourth child," she says. She may think of him and treat him that way as her son. He moves around buying his food, paying his health bills and his taxes, but in fact Tom is a dog.

Catherine and Tom live in Sweden, a country where everyone is expected to lead an orderly life according to rules laid down by the government, which also provides a high level of care for its people. This level of care costs money.

People in Sweden pay taxes on everything, so aren't surprised to find that owning a dog means more taxes. Some people are paying as much as 500 Swedish kronor in taxes a year for the right to keep their dog, which is spent by the government on dog hospitals and sometimes medical treatment for a dog that falls ill. However, most such treatment is expensive, so owners often decide to offer health and even life - for their dog.

In Sweden dog owners must pay for any damage their dog does. A Swedish Kennel Club official explains what this means: if your dog runs out on the road and gets hit by a passing car, you, as the owner, have to pay for any damage done to the car, even if your dog has been killed in the accident.

Q: How old is Catherine?  
A: 54

Q: where does she live?  
A:

**Model answer:** Stockholm  
**Turker answers:** Sweden, Sweden, in Sweden, Sweden

**Table 1.** Examples of naturally occurring demonstrations of English to French and French to English translation found throughout the WebText training set.



Boulder

# ALBERT: A Lite BERT for Self-supervised Learning of Language Representations



[YouTube Video](#)

$V \rightarrow$  vocabulary size ( $\approx 30,000$ )

$E \rightarrow$  vocabulary embedding size

$L \rightarrow$  number of encoder layers

$H \rightarrow$  hidden size

$4H \rightarrow$  feedforward/filter size

$H/64 \rightarrow$  number of attention heads

## Factorized embedding parameterization

Reduce the embedding parameters from

$\mathcal{O}(V \times H)$  to  $\mathcal{O}(V \times E + E \times H)$

This parameter reduction is significant when  $H \gg E$ .

## Cross-layer parameter sharing

### Inter-sentence coherence loss

In addition to the masked language modeling (MLM) loss, BERT uses an additional loss called next-sentence prediction (NSP).

### Positive examples:

consecutive segments from the corpus

### Negative examples:

pairing segments from different documents  
sentence-order prediction (SOP) loss

### Positive examples:

consecutive segments from the corpus

### Negative examples:

swapped consecutive segments

Model	Parameters	Layers	Hidden	Embedding	Parameter-sharing
BERT	base	108M	12	768	768
	large	334M	24	1024	1024
ALBERT	base	12M	12	768	128
	large	18M	24	1024	128
	xlarge	60M	24	2048	128
	xxlarge	235M	12	4096	128

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
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5
	xxlarge	235M	<b>94.1/88.3</b>	<b>88.1/85.1</b>	<b>88.0</b>	<b>95.2</b>	<b>82.3</b>	<b>88.7</b>

The effect of vocabulary embedding size on the performance of ALBERT-base.

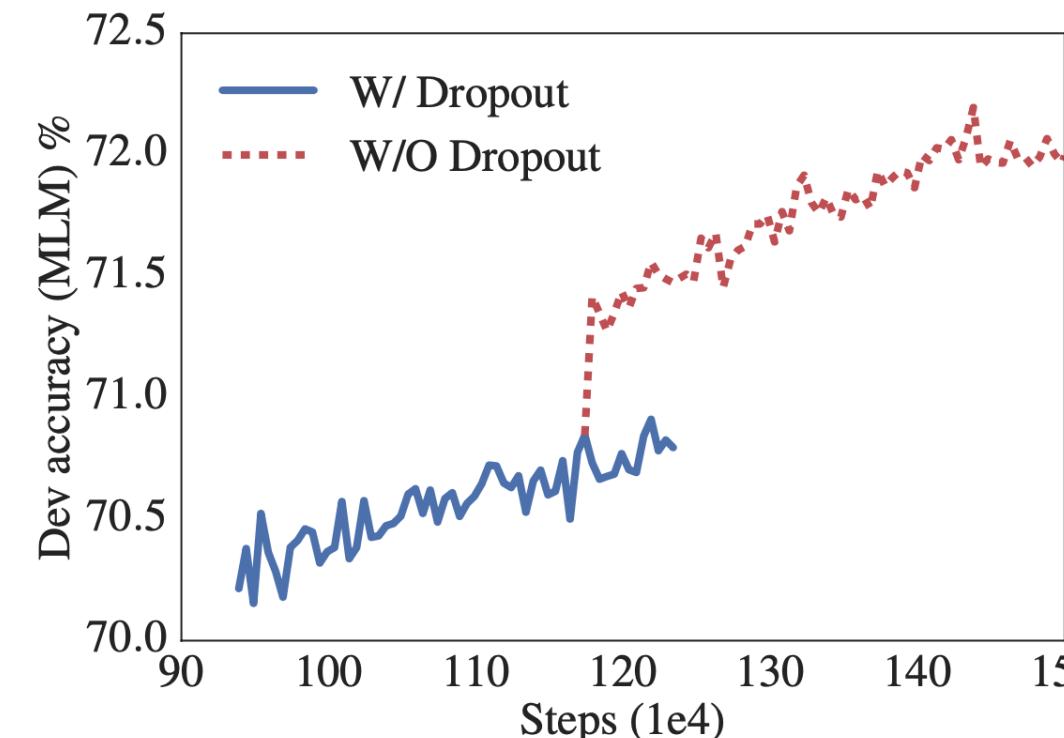
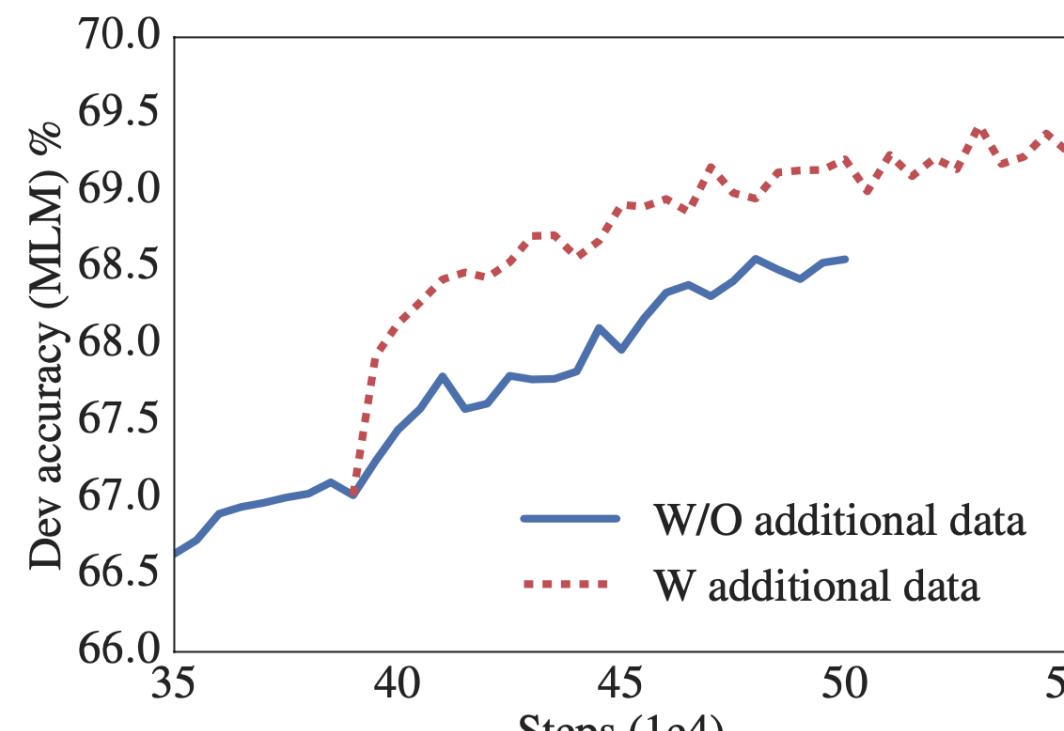
Model	$E$	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT	64	87M	89.9/82.9	80.1/77.8	82.9	91.5	66.7	81.3
	128	89M	89.9/82.8	80.3/77.3	83.7	91.5	67.9	81.7
	256	93M	90.2/83.2	80.3/77.4	84.1	91.9	67.3	81.8
	768	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT	64	10M	88.7/81.4	77.5/74.8	80.8	89.4	63.5	79.0
	128	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1
	256	16M	88.8/81.5	79.1/76.3	81.5	90.3	63.4	79.6
	768	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8

The effect of cross-layer parameter-sharing strategies, ALBERT-base configuration.

Model	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT	all-shared	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3
	shared-attention	83M	89.9/82.7	80.0/77.2	84.0	91.4	67.7
	shared-FFN	57M	89.2/82.1	78.2/75.4	81.5	90.8	62.6
	not-shared	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2
ALBERT	all-shared	12M	89.3/82.3	80.0/77.1	82.0	90.3	64.0
	shared-attention	64M	89.9/82.8	80.7/77.9	83.4	91.9	67.6
	shared-FFN	38M	88.9/81.6	78.6/75.6	82.3	91.7	64.4
	not-shared	89M	89.9/82.8	80.3/77.3	83.2	91.5	67.9

SP tasks	Intrinsic Tasks		
	MLM	NSP	SOP
None	54.9	52.4	53.3
NSP	54.5	90.5	52.0
SOP	54.0	78.9	86.5

SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Downstream Tasks	
					Avg	
88.6/81.5	78.1/75.3	81.5	89.9	61.7	79.0	
88.4/81.5	77.2/74.6	81.6	<b>91.1</b>	62.3	79.2	
<b>89.3/82.3</b>	<b>80.0/77.1</b>	<b>82.0</b>	90.3	<b>64.0</b>	<b>80.1</b>	





# RoBERTa: A Robustly Optimized BERT Pretraining Approach

Boulder

## Robustly optimized BERT approach

replication study of BERT pre-training

### BERT

$x_1, x_2, \dots, x_N \rightarrow$  segment 1 (sequence of tokens)

$y_1, y_2, \dots, y_M \rightarrow$  segment 2 (sequence of tokens)

$[CLS], x_1, x_2, \dots, x_N, [SEP], y_1, y_2, \dots, y_M, [EOS] \rightarrow$  single input sequence

$M + N < T \rightarrow$  maximum sequence length during training

$L \rightarrow$  number of layers

$A \rightarrow$  number of self-attention heads

$H \rightarrow$  hidden dimension

- masked language modeling (MLM)

15% of input tokens are selected for possible replacement. Of the selected tokens, 80% are replaced with [MASK], 10% are left unchanged, and 10% are replaced by a randomly selected vocabulary token.

- next sentence prediction (NSP)

positive examples (consecutive sentences from the text corpus)

negative examples (pairing segments from different documents)

The NSP objective was designed to improve performance on downstream tasks,

such as Natural Language Inference

$T = 512 \rightarrow$  maximum length

$S = 1,000,000 \rightarrow$  updates

$B = 256 \rightarrow$  minibatch

16GB of uncompressed text (BOOKCORPUS + English WIKIPEDIA)

RoBERTa's modifications are simple, they include: (1) training the model longer, with bigger batches, over more data; (2) removing the next sentence prediction objective; (3) training on longer sequences; (4) dynamically changing the masking pattern applied to the training data; and (5) collecting a large new dataset (CC-NEWS) of comparable size to other privately used datasets, to better control for training set size effects.

160GB of uncompressed text:

- BOOKCORPUS + English WIKIPEDIA (16GB)
- CC-NEWS (76GB)
- OPENWEBTEXT (38GB)
- STORIES (31GB)

masking once during data preprocessing

	static	78.3	84.3	92.5
dynamic	78.7	84.0	92.9	

Model	SQuAD 1.1/2.0	MNLI-m	SST-2	RACE
-------	---------------	--------	-------	------

*Our reimplementation (with NSP loss):*

SEGMENT-PAIR	90.4/78.7	84.0	92.9	64.2
SENTENCE-PAIR	88.7/76.2	82.9	92.1	63.0

*Our reimplementation (without NSP loss):*

FULL-SENTENCES	90.4/79.1	84.7	92.5	64.8
DOC-SENTENCES	90.6/79.7	84.7	92.7	65.6
BERT <sub>BASE</sub>	88.5/76.3	84.3	92.8	64.3
XLNet <sub>BASE</sub> (K = 7)	-/81.3	85.8	92.7	66.1
XLNet <sub>BASE</sub> (K = 6)	-/81.0	85.6	93.4	66.7

Model	data	bsz	steps	batch size		
				SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa	with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0
	+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3
	+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0
	+ pretrain even longer	160GB	8K	500K	94.6/89.4	90.2
BERT <sub>LARGE</sub>	with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6
	XLNet <sub>LARGE</sub>	13GB	256	1M	94.0/87.8	88.4
	+ additional data	126GB	2K	500K	94.5/88.8	89.8
						95.6

Model	SQuAD 1.1 EM	SQuAD 2.0 F1	SQuAD 1.1 EM	SQuAD 2.0 F1
Single models on dev, w/o data augmentation				
BERT <sub>LARGE</sub>	84.1	90.9	79.0	81.8
XLNet <sub>LARGE</sub>	<b>89.0</b>	94.5	86.1	88.8
RoBERTa	88.9	<b>94.6</b>	<b>86.5</b>	<b>89.4</b>

Single models on dev, w/o data augmentation		
BERT <sub>LARGE</sub>	84.1	90.9
XLNet <sub>LARGE</sub>	<b>89.0</b>	94.5
RoBERTa	88.9	<b>94.6</b>

Single models on test (as of July 25, 2019)

XLNet<sub>LARGE</sub> 86.3<sup>†</sup> 89.1<sup>†</sup>

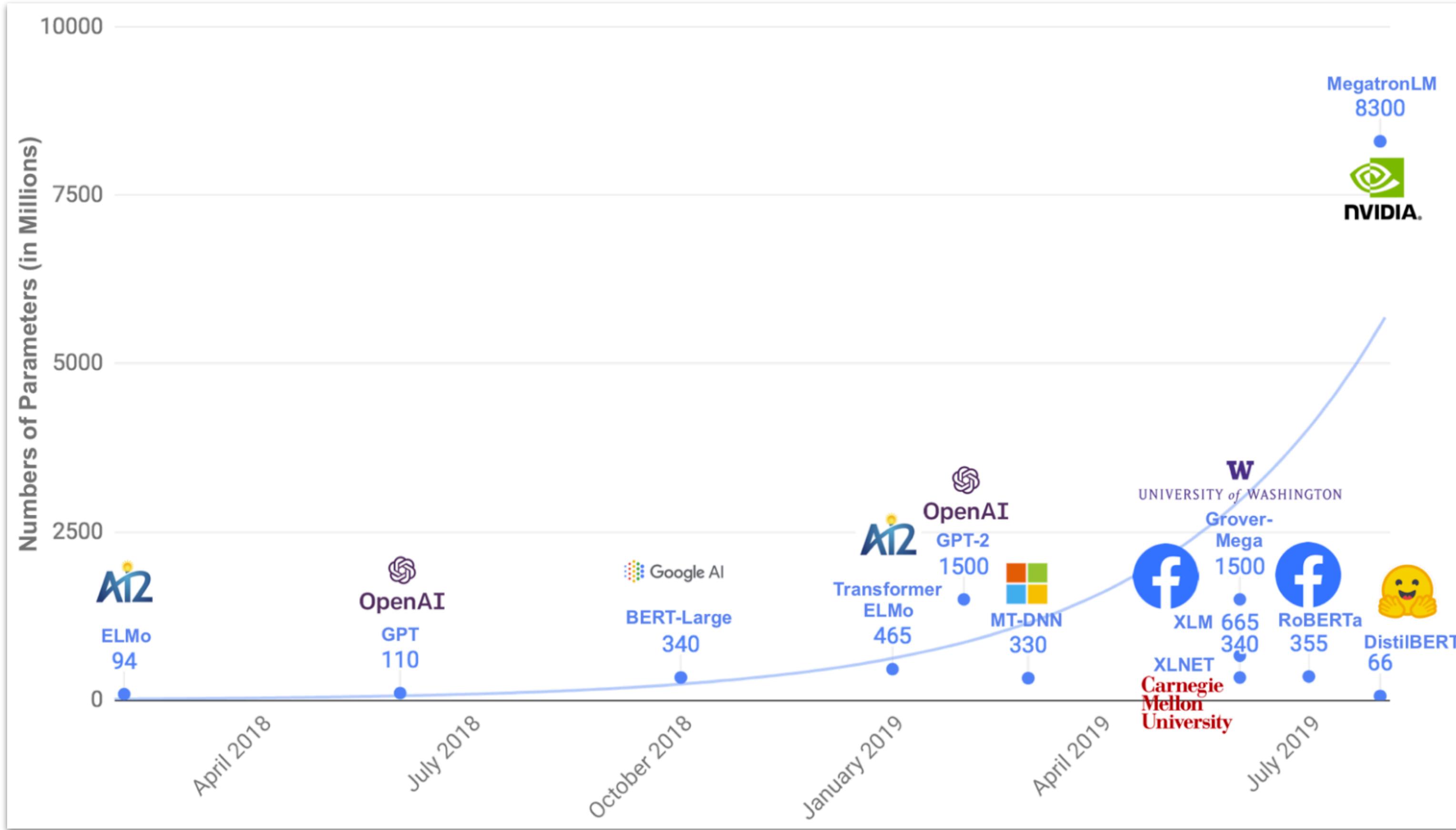
RoBERTa 86.8 89.8

XLNet + SG-Net Verifier 87.0<sup>†</sup> 89.9<sup>†</sup>

Results on the RACE test set.

Model	Accuracy	Middle	High
Single models on test (as of July 25, 2019)			
BERT <sub>LARGE</sub>	72.0	76.6	70.1
XLNet <sub>LARGE</sub>	81.7	85.4	80.2
RoBERTa	<b>83.2</b>	<b>86.5</b>	<b>81.3</b>

# DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter



**DistilBERT retains 97% of BERT performance.**

Model	Score	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	STS-B	WNLI
ELMo	68.7	44.1	68.6	76.6	71.1	86.2	53.4	91.5	70.4	56.3
BERT-base	79.5	56.3	86.7	88.6	91.8	89.6	69.3	92.7	89.0	53.5
DistilBERT	77.0	51.3	82.2	87.5	89.2	88.5	59.9	91.3	86.9	56.3

## Knowledge distillation

compact model → student  
larger model → teacher

$$L_{ce} = \sum_i t_i \log s_i \rightarrow \text{distillation loss}$$

$t_i$  → probability estimated by a teacher  
 $s_i$  → probability estimated by a student

$$p_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)} \rightarrow \text{soft-temperature}$$

Same temperature  $T$  for student and teacher at training  
 $T = 1$  at inference

## Triple Loss

Combining language modeling, distillation and cosine-distance losses!

$L_{mlm}$  → masked language modeling loss

$L_{cos}$  → cosine embedding loss (align the directions of the student and teacher hidden states vectors)

## Student Architecture & Initialization

Number of layers is reduced by a factor of 2!

Initialize the student from the teacher by taking one layer out of two!

Model	# param. (Millions)	Inf. time (seconds)
ELMo	180	895
BERT-base	110	668
DistilBERT	66	410



Boulder

# Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context



[YouTube Playlist](#)

$x = (x_1, x_2, \dots, x_T) \rightarrow$  corpus of tokens

$p(x) = \prod_t p(x_t | x_{<t}) \rightarrow$  Language Modeling

encode the context  $x_{<t}$  into a fixed size hidden state

$$\begin{cases} s_\tau = [x_{\tau,1}, x_{\tau,2}, \dots, x_{\tau,L}] \\ s_{\tau+1} = [x_{\tau+1,1}, x_{\tau+1,2}, \dots, x_{\tau+1,L}] \end{cases}$$

consecutive segments of length  $L$

$$h_\tau^n \in \mathbb{R}^{L \times d}$$

$\nwarrow$   $n$ -th layer hidden state sequence produced for the  $\tau$ -th segment  $s_\tau$

$$\tilde{h}_{\tau+1}^{n-1} = [\text{SG}(h_\tau^{n-1}), h_{\tau+1}^{n-1}] \in \mathbb{R}^{2L \times d}$$

$\nwarrow$  stop gradient

$$q_{\tau+1}^n = h_{\tau+1}^{n-1} W_q^T$$

$$k_{\tau+1}^n = \tilde{h}_{\tau+1}^{n-1} W_k^T$$

$$v_{\tau+1}^n = \tilde{h}_{\tau+1}^{n-1} W_v^T$$

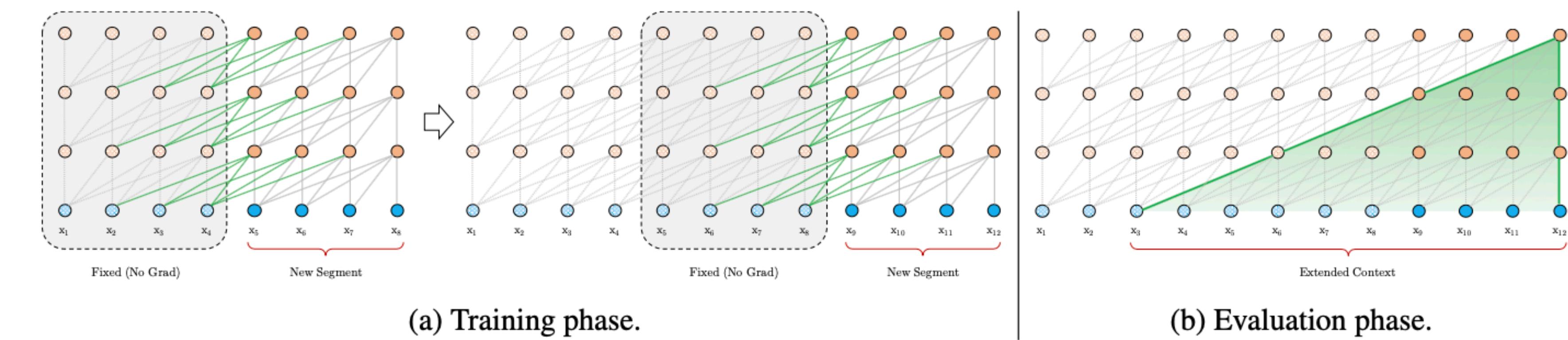
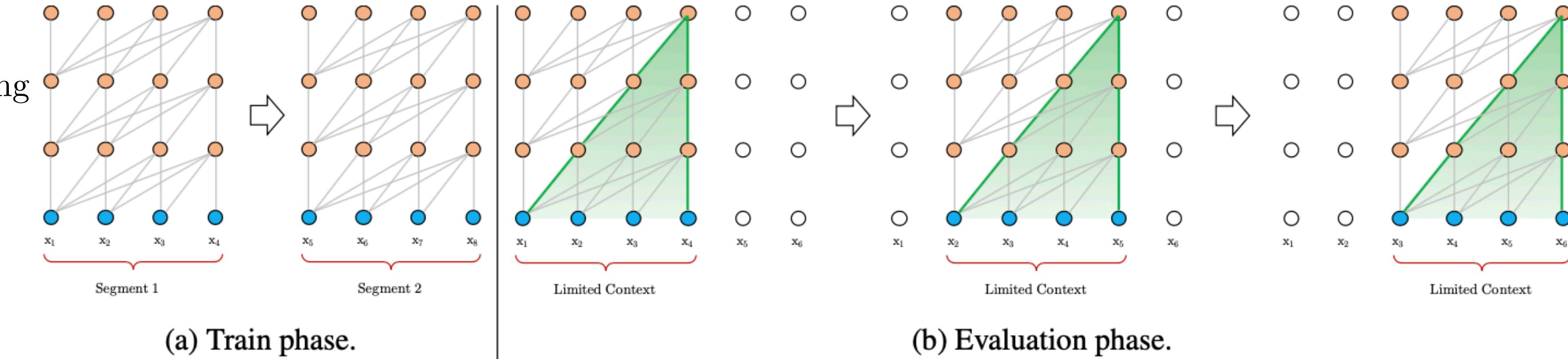
$$h_{\tau+1}^n = \text{transformer-layer}(q_{\tau+1}^n, k_{\tau+1}^n, v_{\tau+1}^n)$$

$$m_\tau^n \in \mathbb{R}^{M \times d} \quad (\text{e.g., } m_\tau^n = h_\tau^n) \rightarrow \text{memory}$$

## Relative Positional Encodings

$$U \in \mathbb{R}^{L_{\max} \times d}$$

$\nwarrow$  “absolute” positional encodings



The model has no information to distinguish the positional difference between  $x_{\tau,j}$  and  $x_{\tau+1,j}$  for any  $j = 1, \dots, L$

$R \in \mathbb{R}^{L_{\max} \times d}$   $i$ -th row: relative distance of  $i$  between two positions

$\nwarrow$  “relative” positional encoding

attention score between query  $q_i$  and key vector  $k_j$

$$\mathbf{A}_{i,j}^{\text{rel}} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(b)}$$

$$+ \underbrace{\mathbf{u}^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(c)} + \underbrace{\mathbf{v}^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(d)}.$$

$$\mathbf{A}_{i,j}^{\text{abs}} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(b)}$$

$$+ \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(c)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(d)}.$$



Boulder

# XLNet: Generalized Autoregressive Pretraining for Language Understanding



[YouTube Video](#)

Pre-training objectives:

- Autoencoding (AE) Language Modeling
- Autoregressive (AR) Language Modeling

$$x = (x_1, x_2, \dots, x_T)$$

$$\left. \begin{aligned} p(x) &= \prod_{t=1}^T p(x_t | x_{<t}) \rightarrow \text{forward product} \\ p(x) &= \prod_{t=T}^1 p(x_t | x_{>t}) \rightarrow \text{backward product} \end{aligned} \right\}_{\text{AR}}$$

[MASK] → AE (pretrain-finetune discrepancy)  
all possible permutations of the factorization order

$$\begin{aligned} \max_{\theta} \log p_{\theta}(\mathbf{x}) &= \sum_{t=1}^T \log p_{\theta}(x_t | \mathbf{x}_{<t}) \\ &= \sum_{t=1}^T \log \frac{\exp(h_{\theta}(\mathbf{x}_{1:t-1})^\top e(x_t))}{\sum_{x'} \exp(h_{\theta}(\mathbf{x}_{1:t-1})^\top e(x'))} \end{aligned}$$

$x \rightarrow \hat{x} \rightarrow \bar{x}$   
mask tokens  
corrupted ([MASK] 15%)

$$\max_{\theta} \log p_{\theta}(\bar{\mathbf{x}} | \hat{\mathbf{x}}) \approx \sum_{t=1}^T m_t \log p_{\theta}(x_t | \hat{\mathbf{x}})$$

$m_t = 1$  indicates  
 $x_t$  is masked

$$= \sum_{t=1}^T m_t \log \frac{\exp(H_{\theta}(\hat{\mathbf{x}})_t^\top e(x_t))}{\sum_{x'} \exp(H_{\theta}(\hat{\mathbf{x}})_t^\top e(x'))}$$

$\mathcal{Z}_T \rightarrow$  set of all possible permutations of  $\{1, 2, \dots, T\}$

$z_t \rightarrow t\text{-th element of a permutation } z \in \mathcal{Z}_T$

$z_{<t} \rightarrow$  the first  $t - 1$  elements of a permutation  $z \in \mathcal{Z}_T$

$$\max_{\theta} \mathbb{E}_{\mathbf{z} \sim \mathcal{Z}_T} \left[ \sum_{t=1}^T \log p_{\theta}(x_{z_t} | \mathbf{x}_{\mathbf{z}_{<t}}) \right]$$

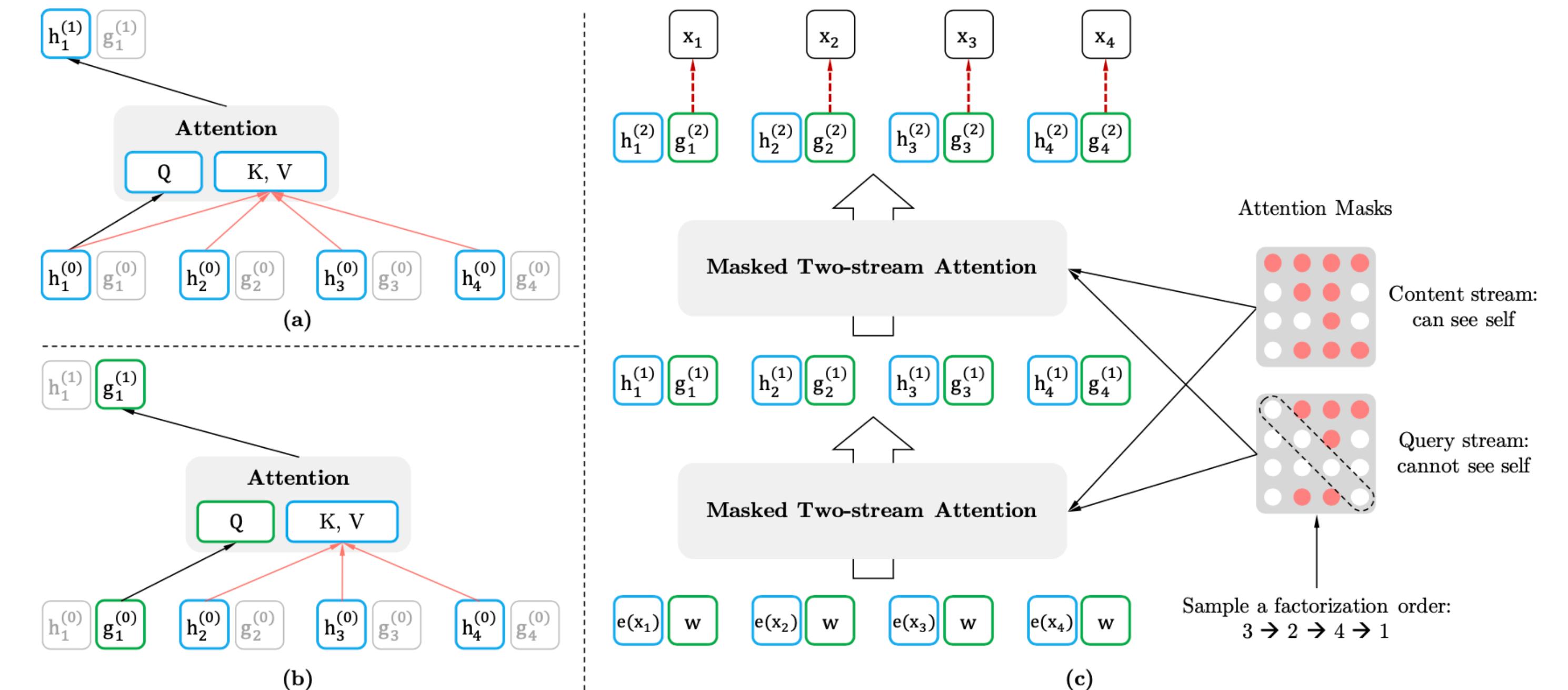
$$\mathcal{J}_{\text{BERT}} = \log p(\text{New} | \text{is a city}) + \log p(\text{York} | \text{is a city}),$$

$$\mathcal{J}_{\text{XLNet}} = \log p(\text{New} | \text{is a city}) + \log p(\text{York} | \text{New, is a city}).$$

Partial Prediction

$$\max_{\theta} \mathbb{E}_{\mathbf{z} \sim \mathcal{Z}_T} [\log p_{\theta}(\mathbf{x}_{\mathbf{z}_{>c}} | \mathbf{x}_{\mathbf{z}_{\leq c}})] = \mathbb{E}_{\mathbf{z} \sim \mathcal{Z}_T} \left[ \sum_{t=c+1}^{|\mathbf{z}|} \log p_{\theta}(x_{z_t} | \mathbf{x}_{\mathbf{z}_{<t}}) \right]$$

Make the objective function depend on the position (i.e., the value of  $z_t$ ) it will predict.

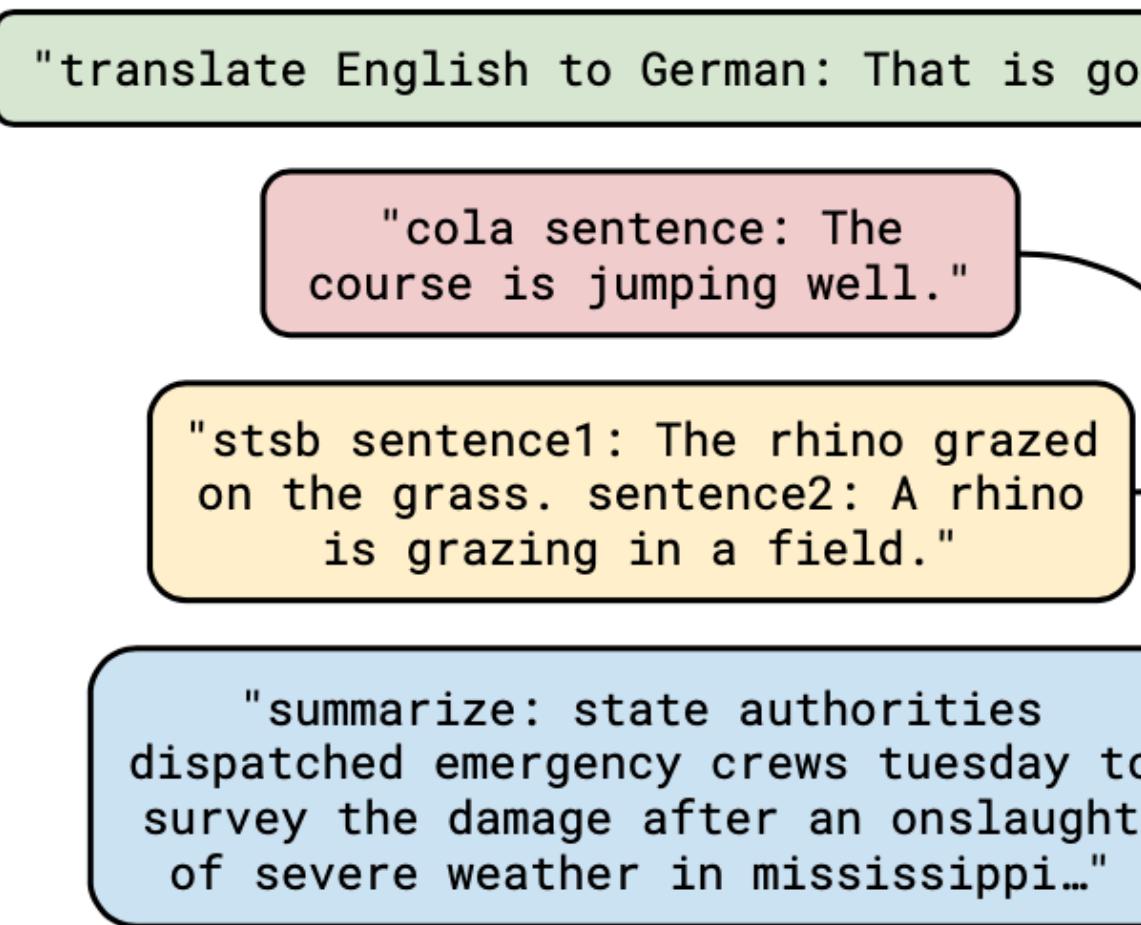


$$\begin{aligned} g_{z_t}^{(m)} &\leftarrow \text{Attention}(Q = g_{z_t}^{(m-1)}, KV = \mathbf{h}_{\mathbf{z}_{<t}}^{(m-1)}; \theta), & (\text{query stream: use } z_t \text{ but cannot see } x_{z_t}) \\ h_{z_t}^{(m)} &\leftarrow \text{Attention}(Q = h_{z_t}^{(m-1)}, KV = \mathbf{h}_{\mathbf{z}_{\leq t}}^{(m-1)}; \theta), & (\text{content stream: use both } z_t \text{ and } x_{z_t}). \end{aligned}$$



Boulder

# Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

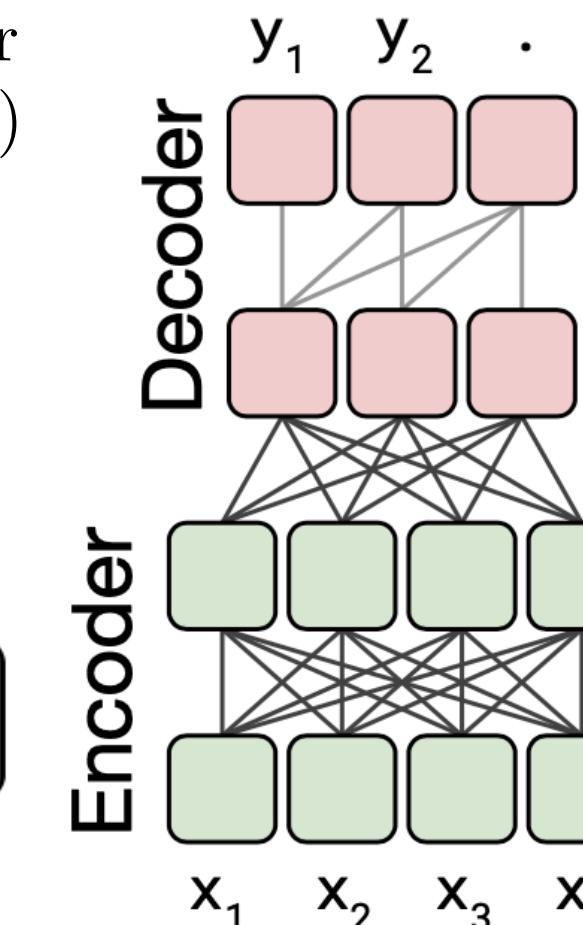


T5: Text-To-Text Transfer Transformer  
C4: Colossal Clean Crawled Corpus (750 GB)

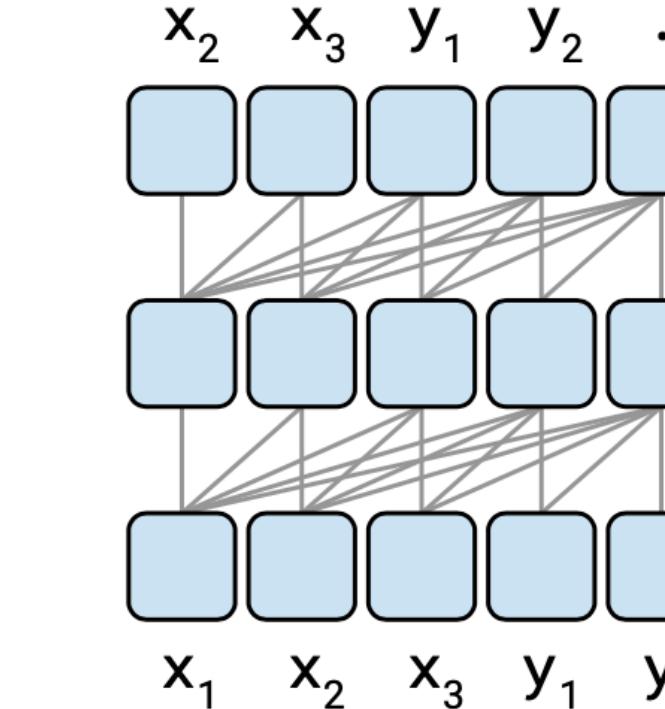
Common Crawl  
20TB of scraped text data each month

**T5**

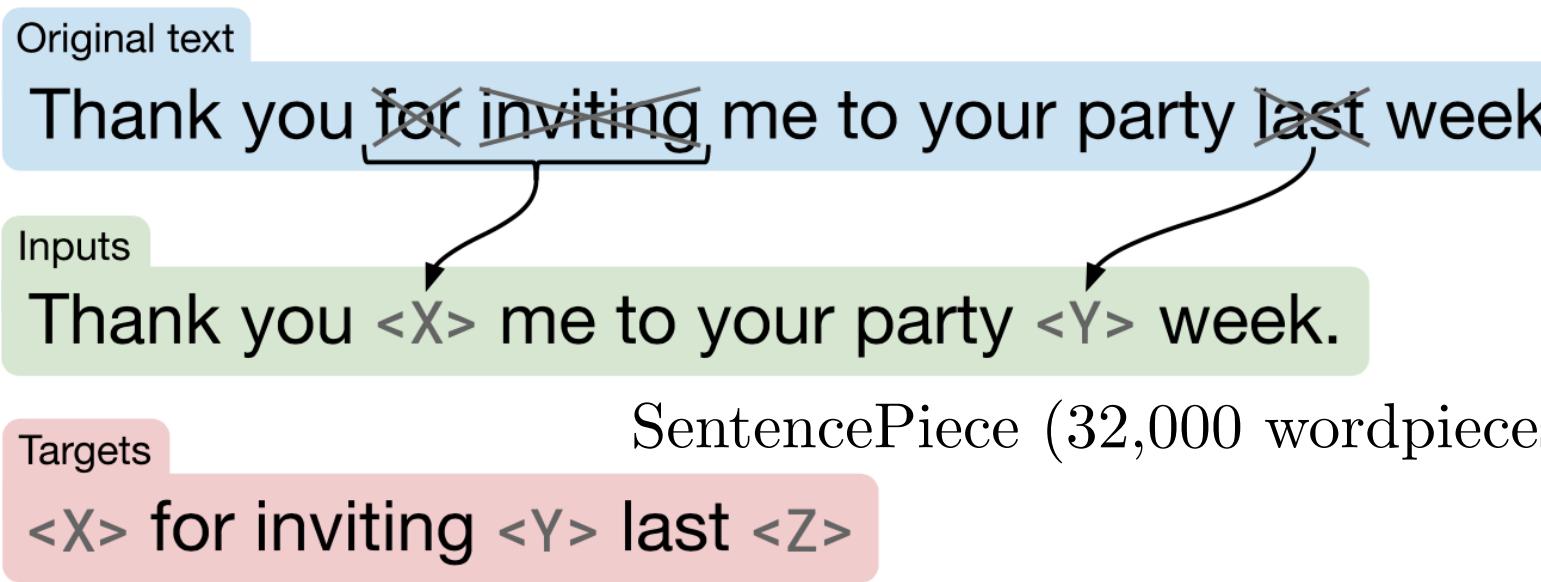
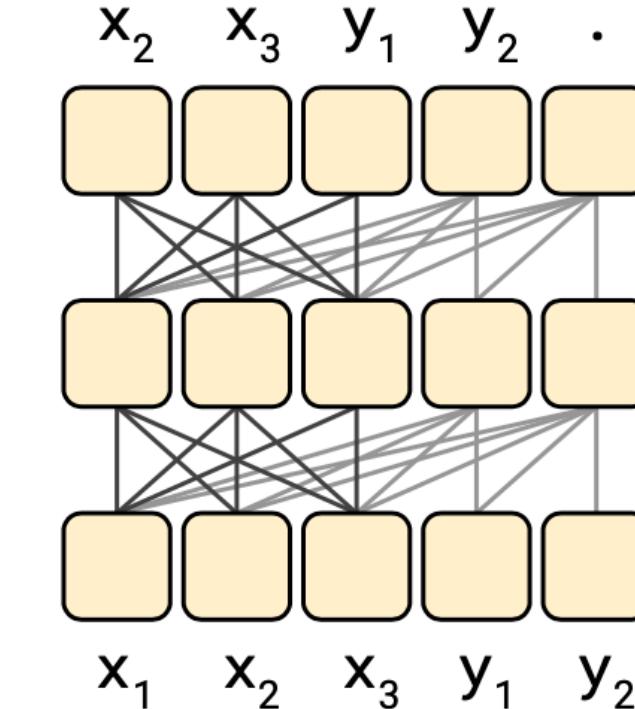
"Das ist gut."  
"not acceptable"  
"3.8"  
"six people hospitalized after a storm in attala county."



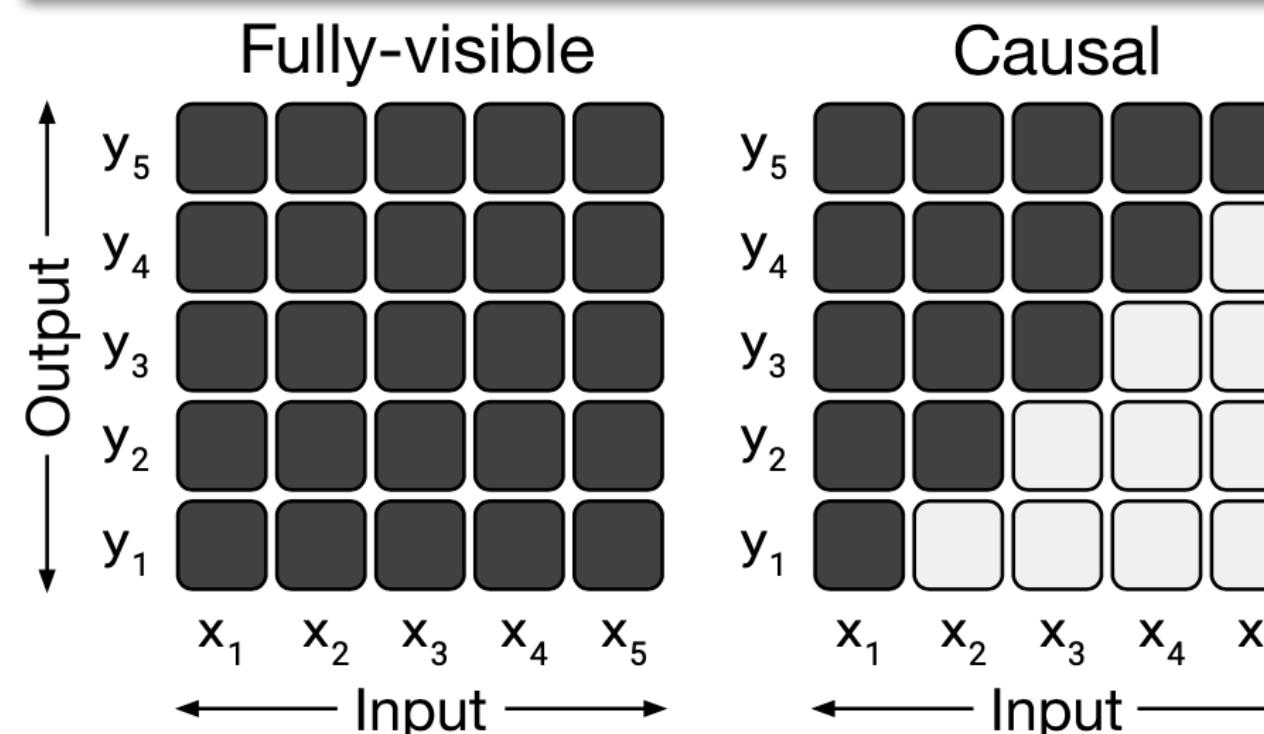
Language model



Prefix LM



Objective	Inputs	Targets
Prefix language modeling BERT-style Devlin et al. (2018)	Thank you for inviting	me to your party last week .
Deshuffling MASS-style Song et al. (2019)	Thank you <M> <M> me to your party apple week . party me for your to . last fun you inviting week Thank	(original text) (original text)
I.i.d. noise, replace spans I.i.d. noise, drop tokens	Thank you <M> <M> me to your party <M> week . Thank you <X> me to your party <Y> week . Thank you me to your party week .	(original text) <X> for inviting <Y> last <Z> for inviting last
Random spans	Thank you <X> to <Y> week .	<X> for inviting me <Y> your party last <Z>



1 / $\sqrt{\max(n, 10^4)}$ → "inverse square root" learning rate schedule								
Data set	Size	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ C4	745GB	83.28	<b>19.24</b>	80.88	71.36	<b>26.98</b>	<b>39.82</b>	<b>27.65</b>
C4, unfiltered	6.1TB	81.46	19.14	78.78	68.04	26.55	39.34	27.21
RealNews-like	35GB	<b>83.83</b>	<b>19.23</b>	80.39	72.38	<b>26.75</b>	<b>39.90</b>	<b>27.48</b>
WebText-like	17GB	<b>84.03</b>	<b>19.31</b>	<b>81.42</b>	71.40	<b>26.80</b>	<b>39.74</b>	<b>27.59</b>
Wikipedia	16GB	81.85	<b>19.31</b>	81.29	68.01	<b>26.94</b>	39.69	<b>27.67</b>
Wikipedia + TBC	20GB	83.65	<b>19.28</b>	<b>82.08</b>	<b>73.24</b>	<b>26.77</b>	39.63	<b>27.57</b>



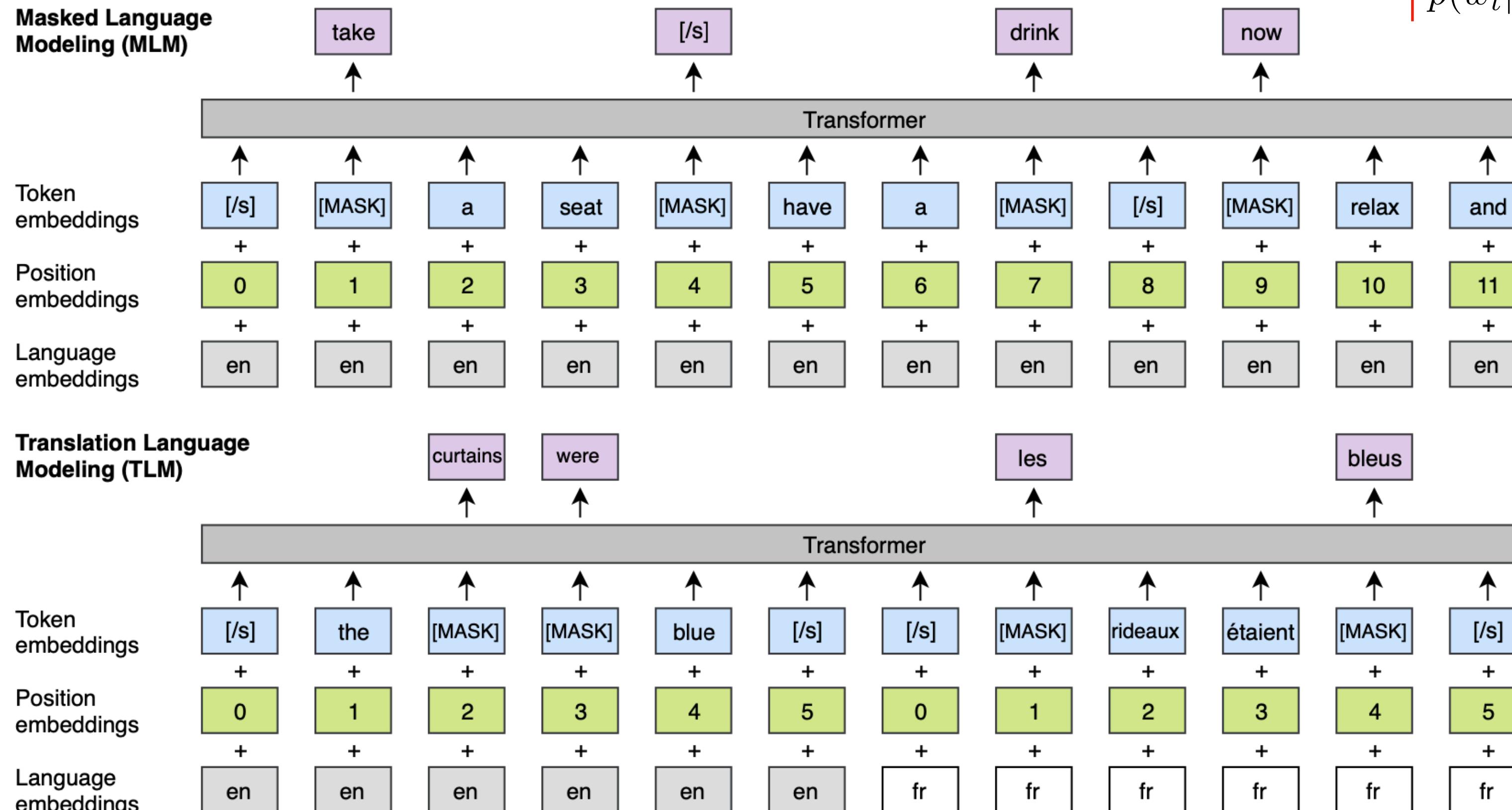
Boulder



[YouTube Video](#)

# Cross-lingual Language Model Pretraining

- multiple languages
  - cross-lingual pretraining
  - cross-lingual language models (XLMs)
  - monolingual data (unsupervised)
  - parallel data (supervised)
- $\{C_i\}_{i=1}^N \rightarrow N$  monolingual corpora  
 $n_i \rightarrow$  number of sentences in  $C_i$



## Shared sub-word vocabulary

Shared vocabulary created using Byte Pair Encoding (BPE)  
Learn the BPE splits on the concatenation of sentences sampled randomly from the monolingual corpora

Sentences are sampled according to a multinomial distribution with probabilities  $\{q_i\}_{i=1}^N$

$$q_i = \frac{p_i^\alpha}{\sum_{j=1}^N p_j^\alpha} \text{ with } p_i = \frac{n_i}{\sum_{k=1}^N n_k}, \alpha = 0.5$$

Alleviates the bias towards high-resource languages!

## Causal Language Modeling (CLM)

$p(w_t | w_1, \dots, w_{t-1}; \theta) \rightarrow$  a Transformer language model

### Results on cross-lingual classification accuracy. Test accuracy on the 15 XNLI languages.

	en	fr	es	de	el	bg	ru	tr	ar	vi	th	zh	hi	sw	ur	$\Delta$
<i>Machine translation baselines (TRANSLATE-TRAIN)</i>																
Devlin et al. (2018)	81.9	-	77.8	75.9	-	-	-	-	70.7	-	-	76.6	-	-	61.6	-
XLM (MLM+TLM)	85.0	80.2	80.8	80.3	78.1	79.3	78.1	74.7	76.5	76.6	75.5	78.6	72.3	70.9	63.2	76.7
<i>Machine translation baselines (TRANSLATE-TEST)</i>																
Devlin et al. (2018)	81.4	-	74.9	74.4	-	-	-	-	70.4	-	-	70.1	-	-	62.1	-
XLM (MLM+TLM)	85.0	79.0	79.5	78.1	77.8	77.6	75.5	73.7	73.7	70.8	70.4	73.6	69.0	64.7	65.1	74.2
<i>Evaluation of cross-lingual sentence encoders</i>																
Conneau et al. (2018b)	73.7	67.7	68.7	67.7	68.9	67.9	65.4	64.2	64.8	66.4	64.1	65.8	64.1	55.7	58.4	65.6
Devlin et al. (2018)	81.4	-	74.3	70.5	-	-	-	-	62.1	-	-	63.8	-	-	58.3	-
Artetxe and Schwenk (2018)	73.9	71.9	72.9	72.6	73.1	74.2	71.5	69.7	71.4	72.0	69.2	71.4	65.5	62.2	61.0	70.2
XLM (MLM)	83.2	76.5	76.3	74.2	73.1	74.0	73.1	67.8	68.5	71.2	69.2	71.9	65.7	64.6	63.4	71.5
XLM (MLM+TLM)	85.0	78.7	78.9	77.8	76.6	77.4	75.3	72.5	73.1	76.1	73.2	76.5	69.6	68.4	67.3	75.1

### BLEU

#### Previous state-of-the-art - Lample et al. (2018b)

	en-fr	fr-en	en-de	de-en	en-ro	ro-en
NMT	25.1	24.2	17.2	21.0	21.2	19.4
PBSMT	28.1	27.2	17.8	22.7	21.3	23.0
PBSMT + NMT	27.6	27.7	20.2	25.2	25.1	23.9

#### Our results for different encoder and decoder initializations

EMB	EMB	29.4	29.4	21.3	27.3	27.5	26.6
-	-	13.0	15.8	6.7	15.3	18.9	18.3
-	CLM	25.3	26.4	19.2	26.0	25.7	24.6
-	MLM	29.2	29.1	21.6	28.6	28.2	27.3
CLM	-	28.7	28.2	24.4	30.3	29.2	28.0
CLM	CLM	30.4	30.0	22.7	30.5	29.0	27.8
CLM	MLM	32.3	31.6	24.3	32.5	31.6	29.8
MLM	-	31.6	32.1	27.0	33.2	31.8	30.5
MLM	CLM	33.4	32.3	24.9	32.9	31.7	30.4
MLM	MLM	33.4	33.3	26.4	34.3	33.3	31.8

Low-resource language model	
Training languages	Nepali perplexity
Nepali	157.2
Nepali + English	140.1
Nepali + Hindi	115.6
Nepali + English + Hindi	109.3



Boulder

# SpanBERT: Improving Pre-training by Representing and Predicting Spans

Extractive Question Answering

“Which NFL team won Super Bowl 50?”

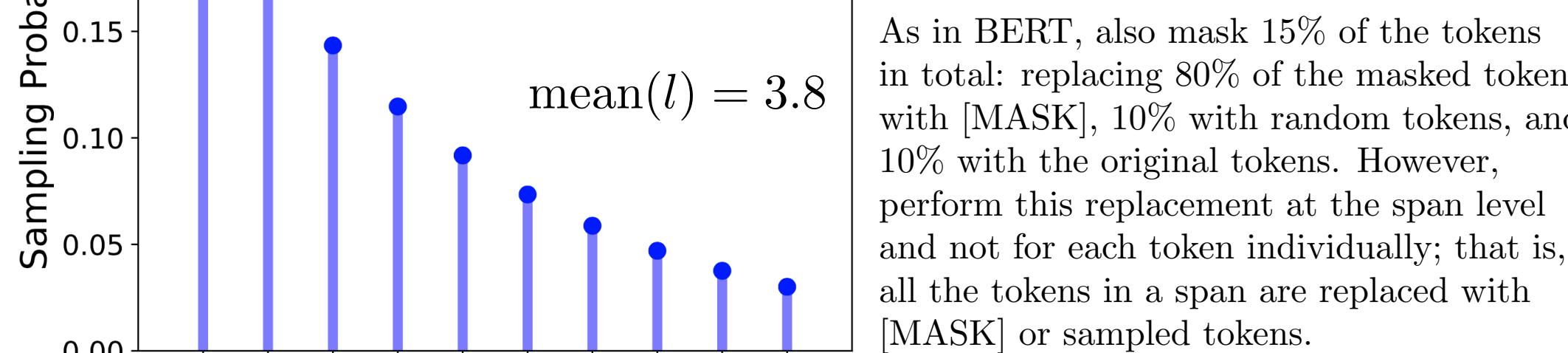
determining that “Denver Broncos” is a type of “NFL team” is critical for answering this question

## Span Masking

$X = (x_1, \dots, x_n) \rightarrow$  sequence of tokens

select a subset of tokens  $Y \subset X$  by iteratively sampling spans of text until the masking budget (e.g., 15% of  $X$ ) has been spent at each iteration sample a span length (number of words) from a geometric distribution (skewed towards shorter spans)

randomly (uniformly) select the starting point for the span to be masked



## Span Boundary Objective (SBO)

$x_1, \dots, x_n \rightarrow$  output of the transformer encoder for each token in the sequence

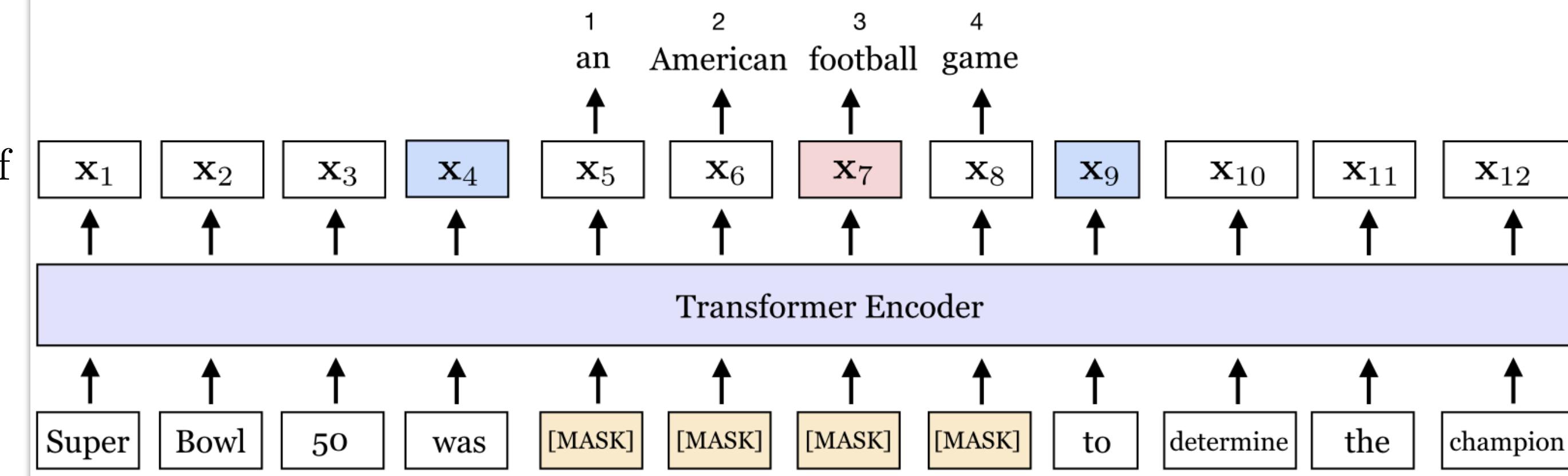
$(x_s, \dots, x_e) \in Y \rightarrow$  masked span of tokens

$(s, e) \rightarrow$  start and end positions

$x_{s-1}, x_{e+1} \rightarrow$  output encodings of the external boundary tokens

$$\mathcal{L}(\text{football}) = \mathcal{L}_{\text{MLM}}(\text{football}) + \mathcal{L}_{\text{SBO}}(\text{football})$$

$$= -\log P(\text{football} | \mathbf{x}_7) - \log P(\text{football} | \mathbf{x}_4, \mathbf{x}_9, \mathbf{p}_3)$$



$\mathbf{p}_{i-s+1} \rightarrow$  position embedding of the target token

$\mathbf{y}_i = f(\mathbf{x}_{s-1}, \mathbf{x}_{e+1}, \mathbf{p}_{i-s+1}) \rightarrow$  representation of token  $x_i$  in the span

$$\mathbf{h}_0 = [\mathbf{x}_{s-1}; \mathbf{x}_{e+1}; \mathbf{p}_{i-s+1}]$$

$\mathbf{h}_1 = \text{LayerNorm}(\text{GeLU}(\mathbf{W}_1 \mathbf{h}_0)) \rightarrow$  two layer feed-forward network

$$\mathbf{y}_i = \text{LayerNorm}(\text{GeLU}(\mathbf{W}_2 \mathbf{h}_1))$$

Gaussian error Linear Unit (GeLU)

$$\mathcal{L}(x_i) = \mathcal{L}_{\text{MLM}}(x_i) + \mathcal{L}_{\text{SBO}}(x_i)$$

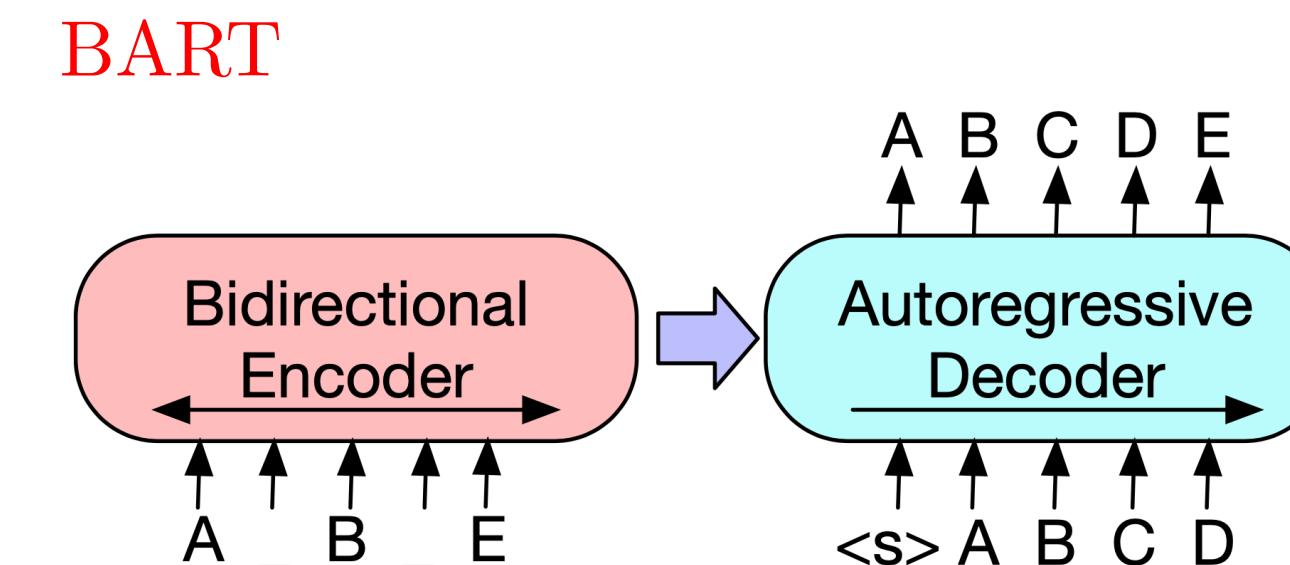
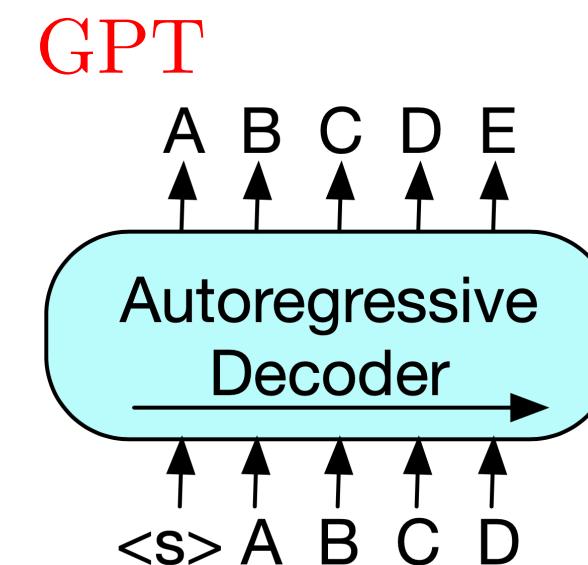
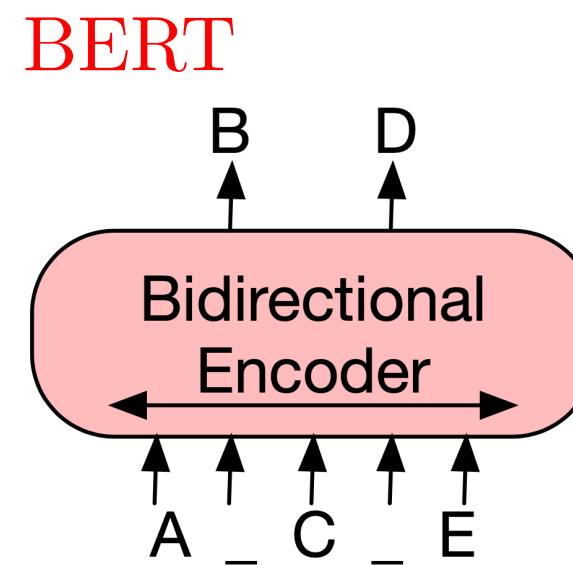
$$= -\log P(x_i | \mathbf{x}_i) - \log P(x_i | \mathbf{y}_i)$$

reuse the input embedding for the target tokens in both MLM and SBO

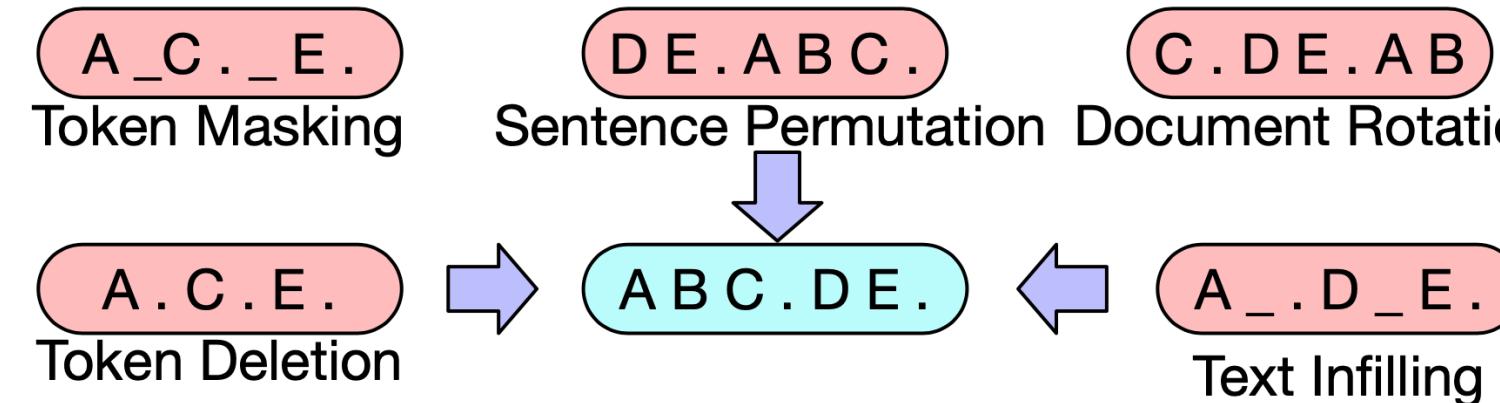
	SQuAD 1.1		SQuAD 2.0	
	EM	F1	EM	F1
Human Perf.	82.3	91.2	86.8	89.4
Google BERT	84.3	91.3	80.0	83.3
Our BERT	86.5	92.6	82.8	85.9
Our BERT-1seq	87.5	93.3	83.8	86.6
SpanBERT	<b>88.8</b>	<b>94.6</b>	<b>85.7</b>	<b>88.7</b>



# BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension

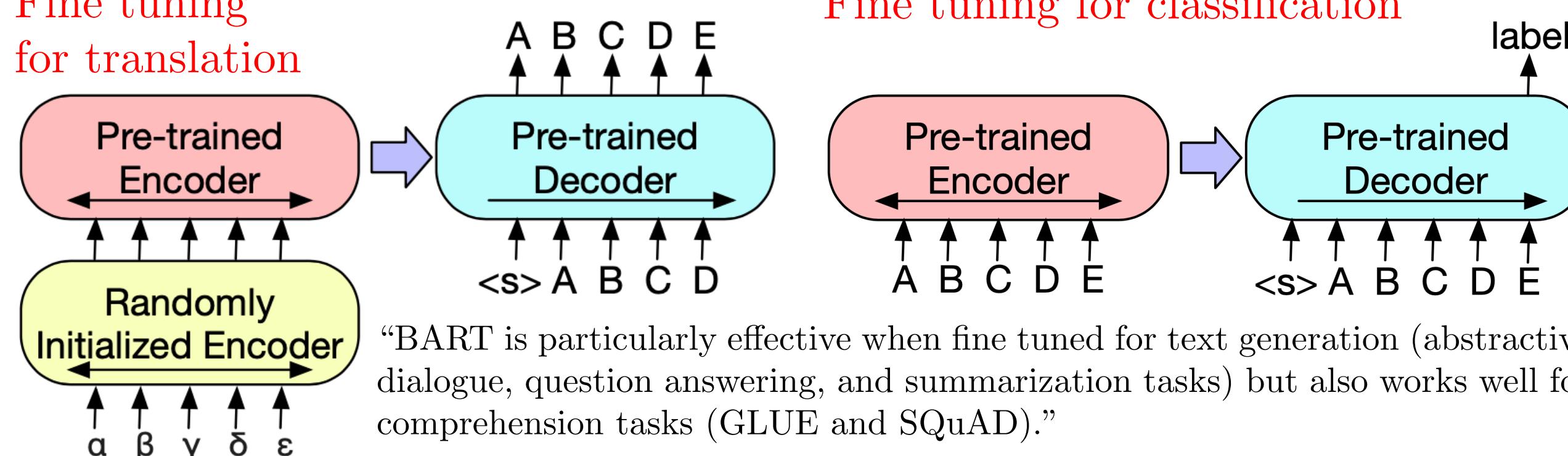


“A key advantage of this setup is the noising flexibility; arbitrary transformations can be applied to the original text, including changing its length.”



“Achieving the best performance by both randomly shuffling the order of the original sentences and using a novel in-filling scheme, where arbitrary length spans of text (including zero length) are replaced with a single mask token.”

Fine tuning  
for translation



	SQuAD 1.1 EM/F1	SQuAD 2.0 EM/F1	MNLI m/mm	SST Acc	QQP Acc	QNLI Acc	STS-B Acc	RTE Acc	MRPC Acc	CoLA Mcc
BERT	84.1/90.9	79.0/81.8	86.6/-	93.2	91.3	92.3	90.0	70.4	88.0	60.6
UniLM	-/-	80.5/83.4	87.0/85.9	94.5	-	92.7	-	70.9	-	61.1
XLNet	<b>89.0/94.5</b>	86.1/88.8	89.8/-	95.6	91.8	93.9	91.8	83.8	89.2	63.6
RoBERTa	88.9/94.6	<b>86.5/89.4</b>	<b>90.2/90.2</b>	96.4	92.2	94.7	<b>92.4</b>	86.6	<b>90.9</b>	<b>68.0</b>
<b>BART</b>	<b>88.8/94.6</b>	86.1/89.2	89.9/90.1	<b>96.6</b>	<b>92.5</b>	<b>94.9</b>	91.2	<b>87.0</b>	90.4	62.8

	Results on two standard summarization datasets.			CNN/DailyMail			XSum		
	R1	R2	RL	R1	R2	RL	R1	R2	RL
Lead-3	40.42	17.62	36.67	16.30	1.60	11.95			
PTGEN (See et al., 2017)	36.44	15.66	33.42	29.70	9.21	23.24			
PTGEN+COV (See et al., 2017)	39.53	17.28	36.38	28.10	8.02	21.72			
UniLM	43.33	20.21	40.51	-	-	-			
BERTSUMABS (Liu & Lapata, 2019)	41.72	19.39	38.76	38.76	16.33	31.15			
BERTSUMEXTABS (Liu & Lapata, 2019)	42.13	19.60	39.18	38.81	16.50	31.27			
<b>BART</b>	<b>44.16</b>	<b>21.28</b>	<b>40.90</b>	<b>45.14</b>	<b>22.27</b>	<b>37.25</b>			

Example summaries from the XSum-tuned BART model on WikiNews articles.

Source Document (abbreviated)	BART Summary
PG&E stated it scheduled the blackouts in response to forecasts for high winds amid dry conditions. The aim is to reduce the risk of wildfires. Nearly 800 thousand customers were scheduled to be affected by the shutoffs which were expected to last through at least midday tomorrow.	Power has been turned off to millions of customers in California as part of a power shutoff plan.

abstractive question answering	ELI5			conversational response generation			ConvAI2	
	R1	R2	RL	Valid F1	Valid PPL			
Best Extractive	23.5	3.1	17.5					
Language Model	27.8	4.7	23.1	Seq2Seq + Attention			16.02	35.07
Seq2Seq	28.3	5.1	22.8	Best System			19.09	17.51
Seq2Seq Multitask	28.9	5.4	23.1	BART			<b>20.72</b>	<b>11.85</b>



Boulder

# Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks

Sentence-pair tasks (e.g., semantic textual similarity)

STS

BERT: sentence 1, sentence 2  $\mapsto$  similarity score

Massive computational overhead!

Finding the most similar pair in a collection of  $n = 10,000$  sentences requires about 50 million ( $n(n-1)/2 = 49,995,000$ ) inference computations ( $\approx 65$  hours) with BERT.

BERT: sentence  $\mapsto$  vector

average pooling or output of the first token (i.e., the [CLS] token)

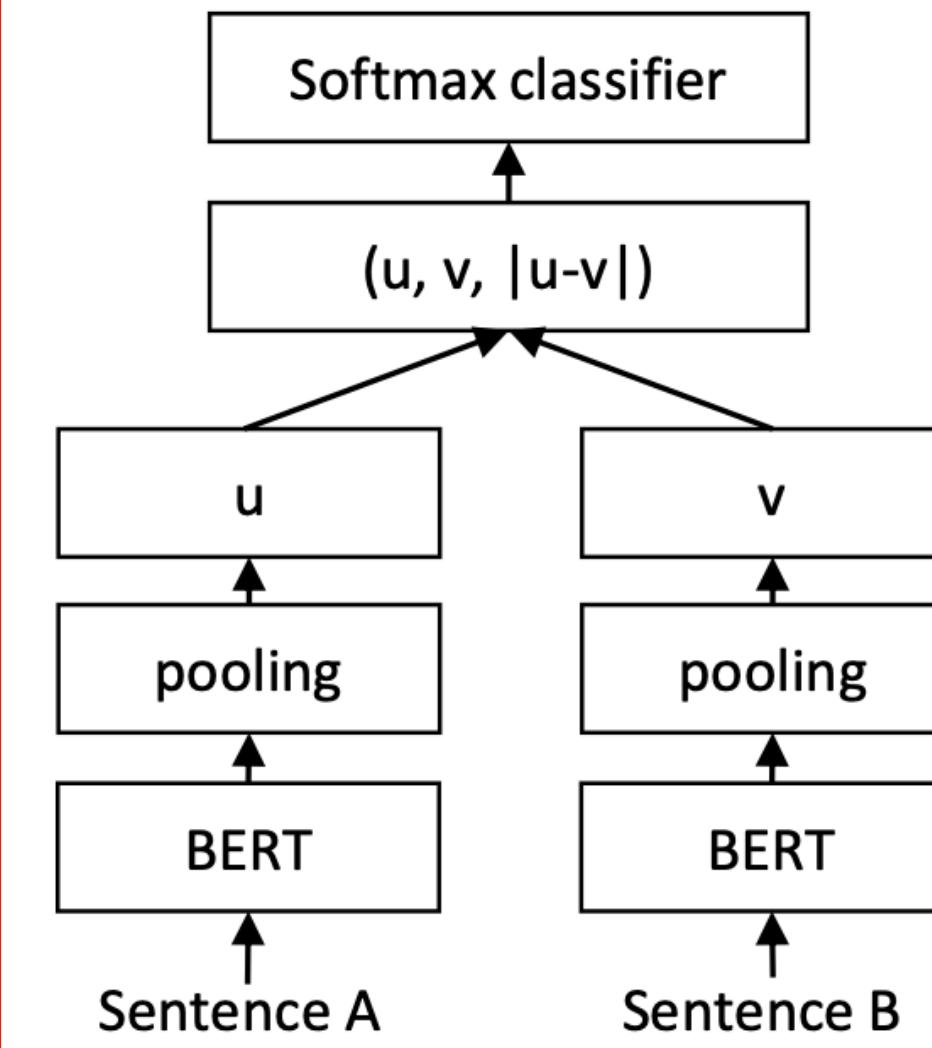
The complexity reduces from 65 hours to the computation of 10,000 sentence embeddings ( $\approx 5$  seconds).

This common practice yields rather bad sentence embeddings, often worse than averaging GloVe embeddings.

Spearman rank correlation between the cosine similarity of sentence representations and the gold labels for various Textual Similarity (STS) tasks.

Model	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
Avg. GloVe embeddings	55.14	70.66	59.73	68.25	63.66	58.02	53.76	61.32
Avg. BERT embeddings	38.78	57.98	57.98	63.15	61.06	46.35	58.40	54.81
BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	42.63	29.19
InferSent - Glove	52.86	66.75	62.15	72.77	66.87	68.03	65.65	65.01
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	<b>76.69</b>	71.22
SBERT-NLI-base	70.97	76.53	73.19	79.09	74.30	77.03	72.91	74.89
SBERT-NLI-large	72.27	<b>78.46</b>	<b>74.90</b>	80.99	76.25	<b>79.23</b>	73.75	76.55
SRoBERTa-NLI-base	71.54	72.49	70.80	78.74	73.69	77.77	74.46	74.21
SRoBERTa-NLI-large	<b>74.53</b>	77.00	73.18	<b>81.85</b>	<b>76.82</b>	79.10	74.29	<b>76.68</b>

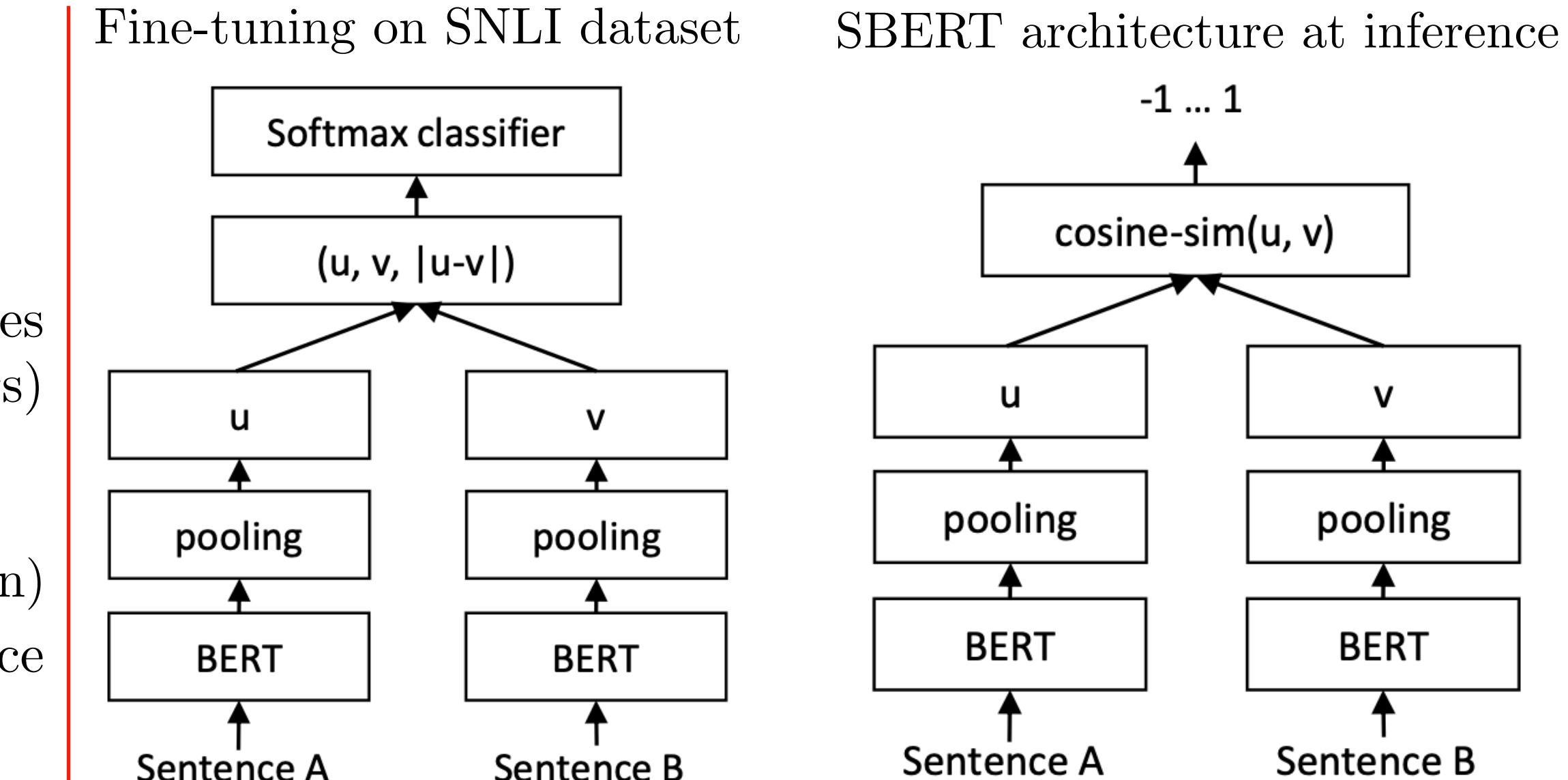
Fine-tuning on SNLI dataset



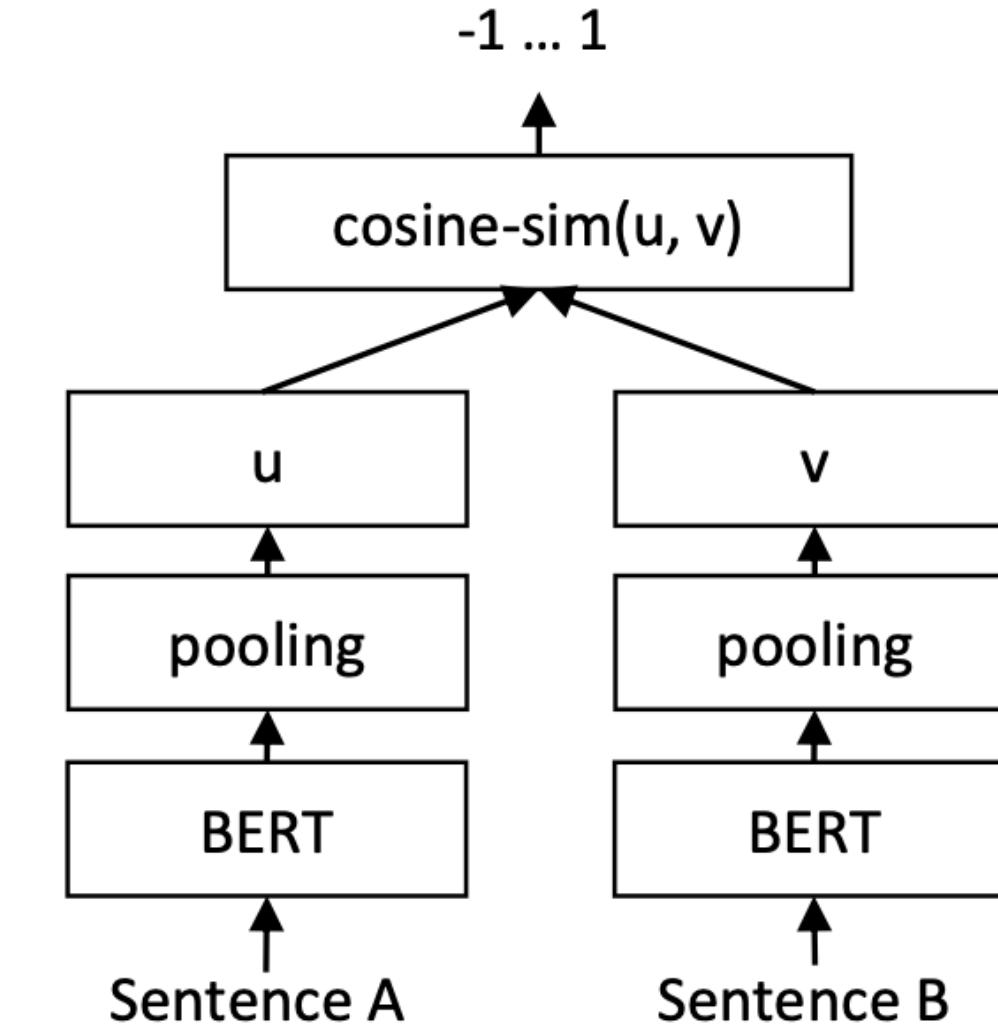
Model	Spearman
<i>Not trained for STS</i>	
Avg. GloVe embeddings	58.02
Avg. BERT embeddings	46.35
InferSent - GloVe	68.03
Universal Sentence Encoder	74.92
SBERT-NLI-base	77.03
SBERT-NLI-large	79.23
<i>Trained on STS benchmark dataset</i>	
BERT-STSb-base	$84.30 \pm 0.76$
SBERT-STSb-base	$84.67 \pm 0.19$
SRoBERTa-STSb-base	<b><math>84.92 \pm 0.34</math></b>
BERT-STSb-large	<b><math>85.64 \pm 0.81</math></b>
SBERT-STSb-large	$84.45 \pm 0.43$
SRoBERTa-STSb-large	$85.02 \pm 0.76$

Model	NLI	STSb
<i>Pooling Strategy</i>		
MEAN	<b>80.78</b>	<b>87.44</b>
MAX	79.07	69.92
CLS	79.80	86.62
<i>Concatenation</i>		
( $u, v$ )	66.04	-
( $ u - v $ )	69.78	-
( $u * v$ )	70.54	-
( $ u - v , u * v$ )	78.37	-
( $u, v, u * v$ )	77.44	-
( $u, v,  u - v $ )	<b>80.78</b>	-
( $u, v,  u - v , u * v$ )	80.44	-

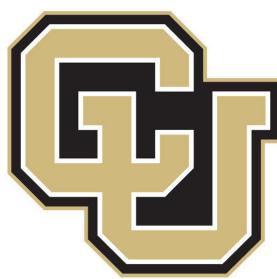
Unsupervised STS



SBERT architecture at inference



BERT is also not suitable for unsupervised tasks like clustering.



Boulder



[YouTube Video](#)

# Language Models are Few-Shot Learners

## Traditional fine-tuning (not used for GPT-3)

### Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.



## In-context Learning

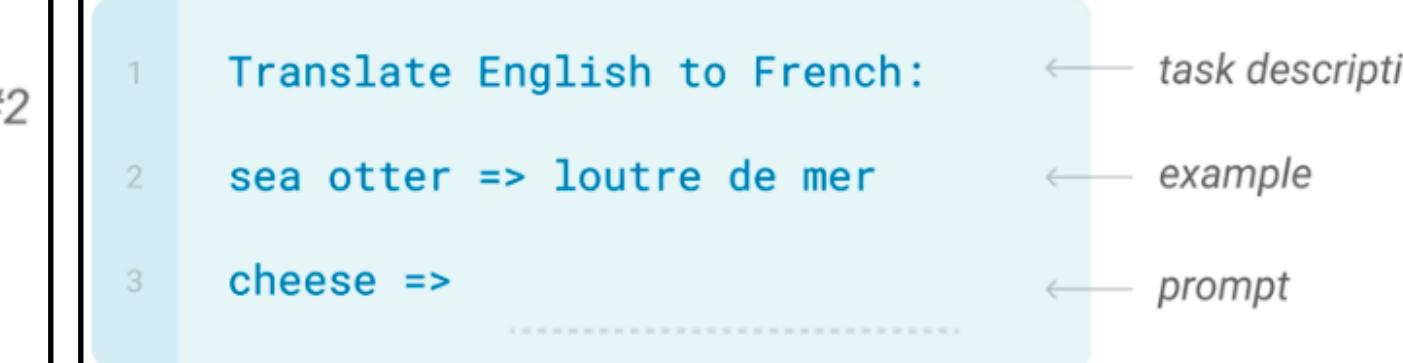
### Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



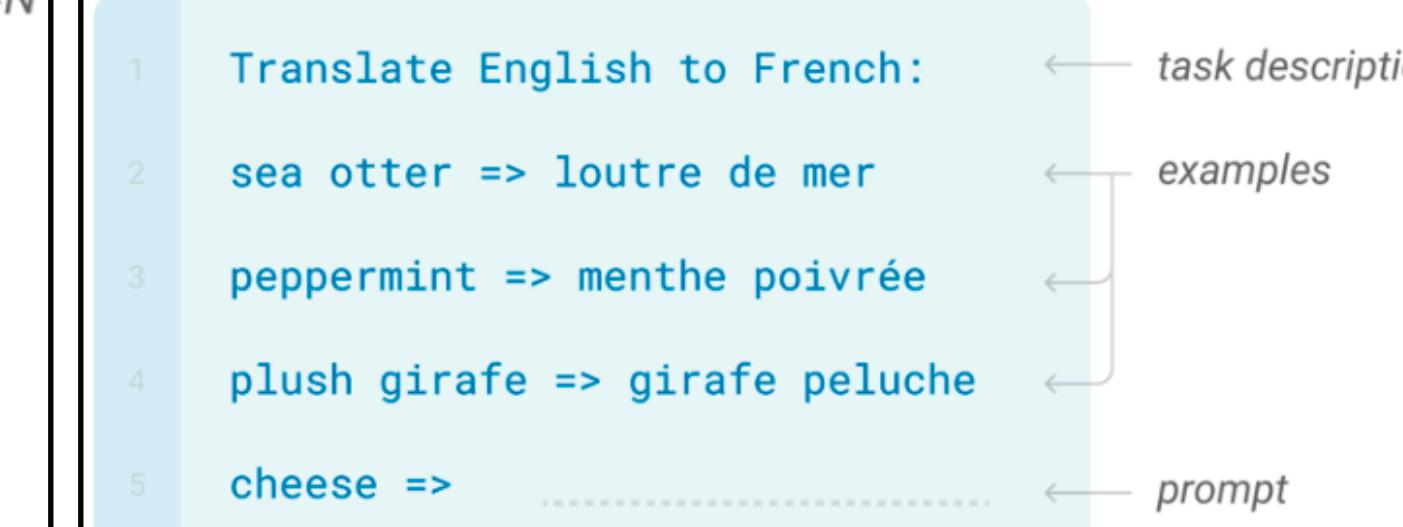
### One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



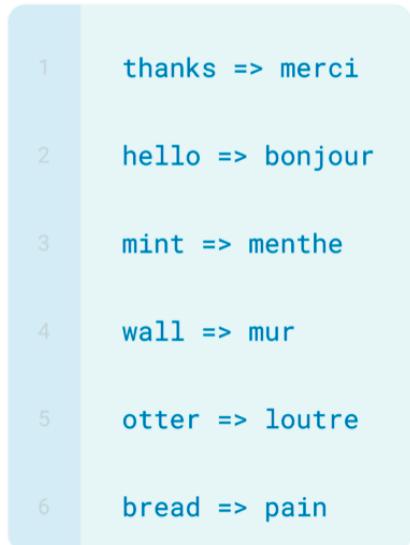
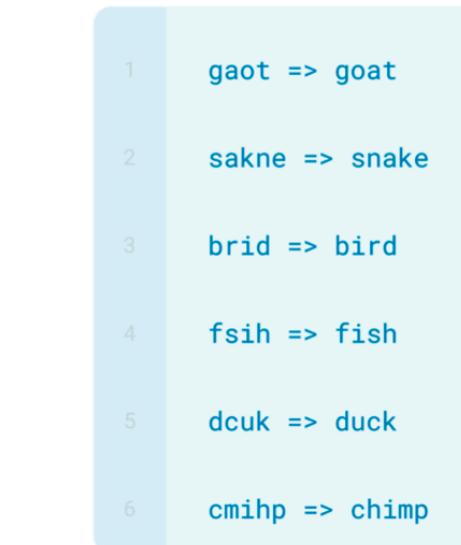
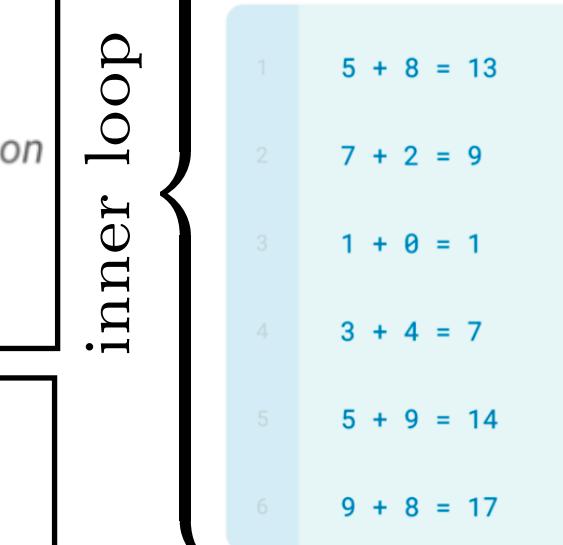
### Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



outer loop

Learning via SGD during unsupervised pre-training



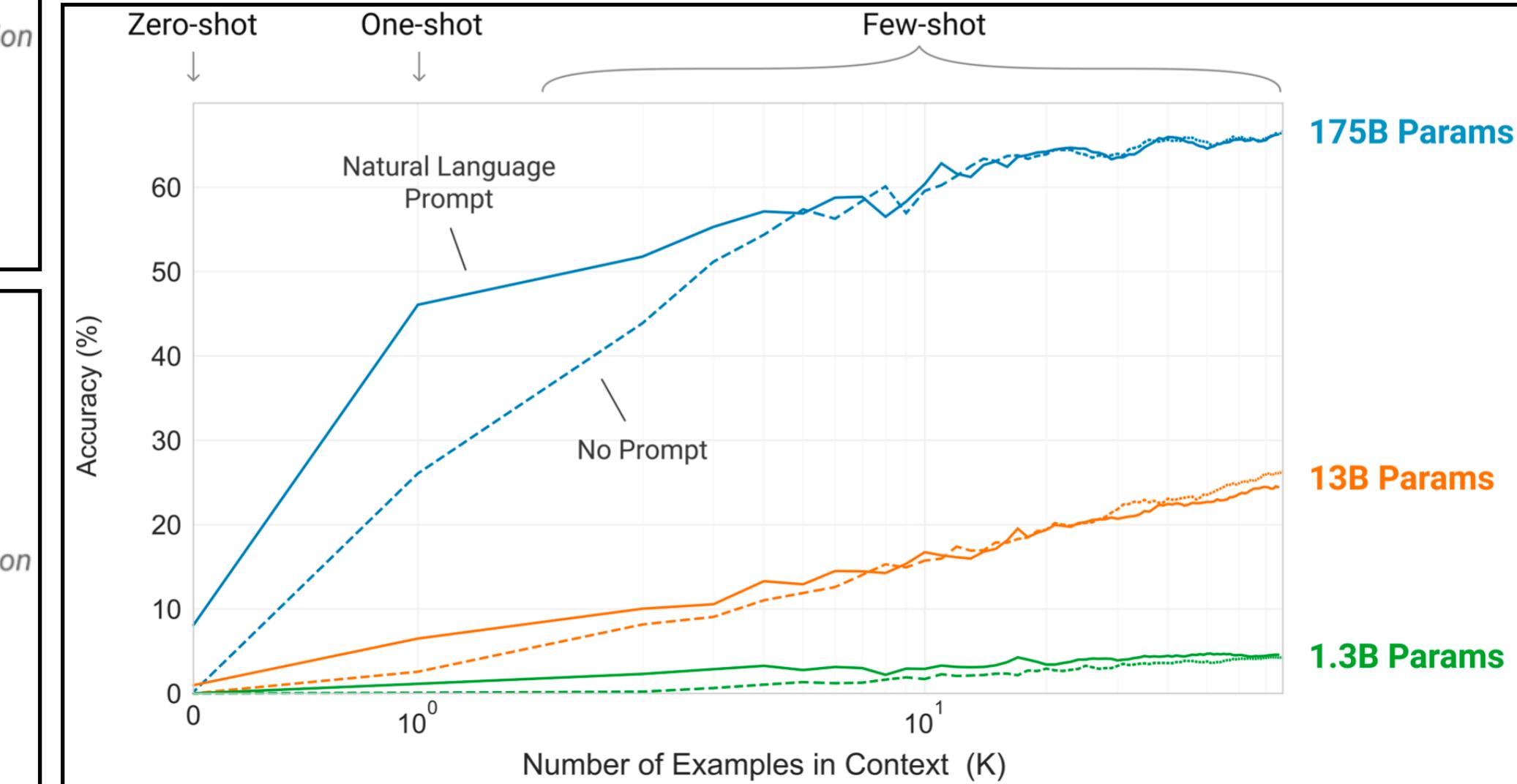
inner loop

In-context learning

In-context learning

In-context learning

## Larger models make increasingly efficient use of in-context information



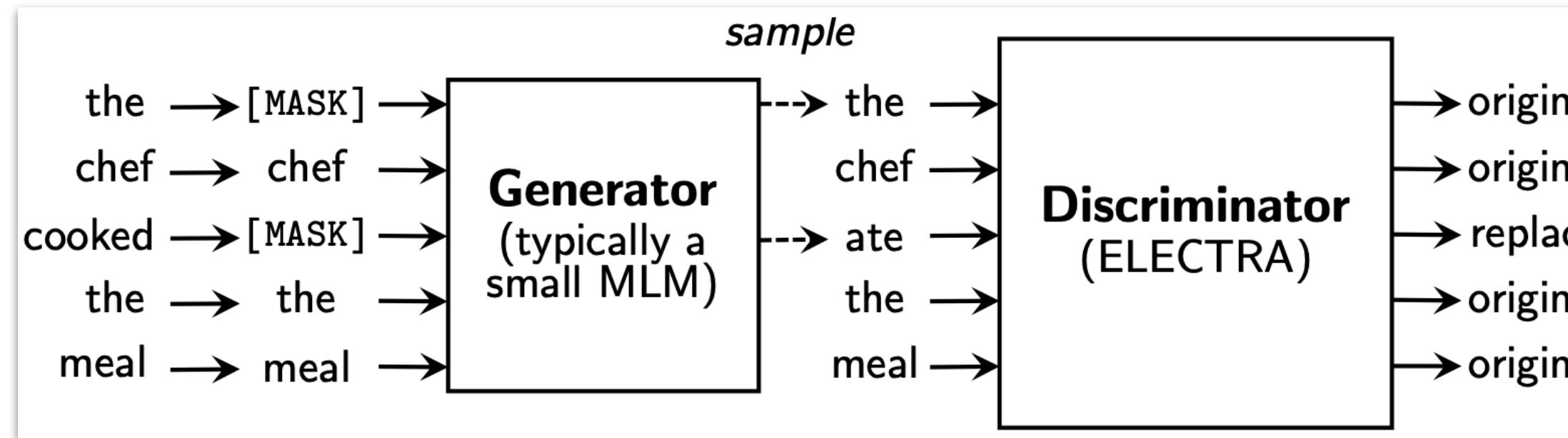
- Common Crawl dataset (filtered) – Webtext dataset
- Two internet-based books corpora – English language Wikipedia



Boulder

# ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators

Masked Language Modeling (MLM) approaches incur a substantial compute cost because the network only learns from 15% of the tokens per example!  
Efficiently Learning an Encoder that Classifies Token Replacements Accurately (ELECTRA)



$G \rightarrow$  generator

$D \rightarrow$  discriminator

$x = [x_1, \dots, x_n] \rightarrow$  sequence of input tokens

$h(x) = [h_1, \dots, h_n] \rightarrow$  sequence of contextualized vector representations

$h \rightarrow$  encoder (transformer network)

$$p_G(x_t|x) = \exp(e(x_t)^T h_G(x)_t) / \sum_{x'} \exp(e(x')^T h_G(x)_t)$$

$e(x_t) \rightarrow$  token embedding

$D(x, t) = \text{sigmoid}(w^T h_D(x_t)) \rightarrow$  probability of token  $x_t$  being “real”

$m = [m_1, \dots, m_k] \rightarrow$  random set of positions (integers between 1 and  $n$ ) to mask out

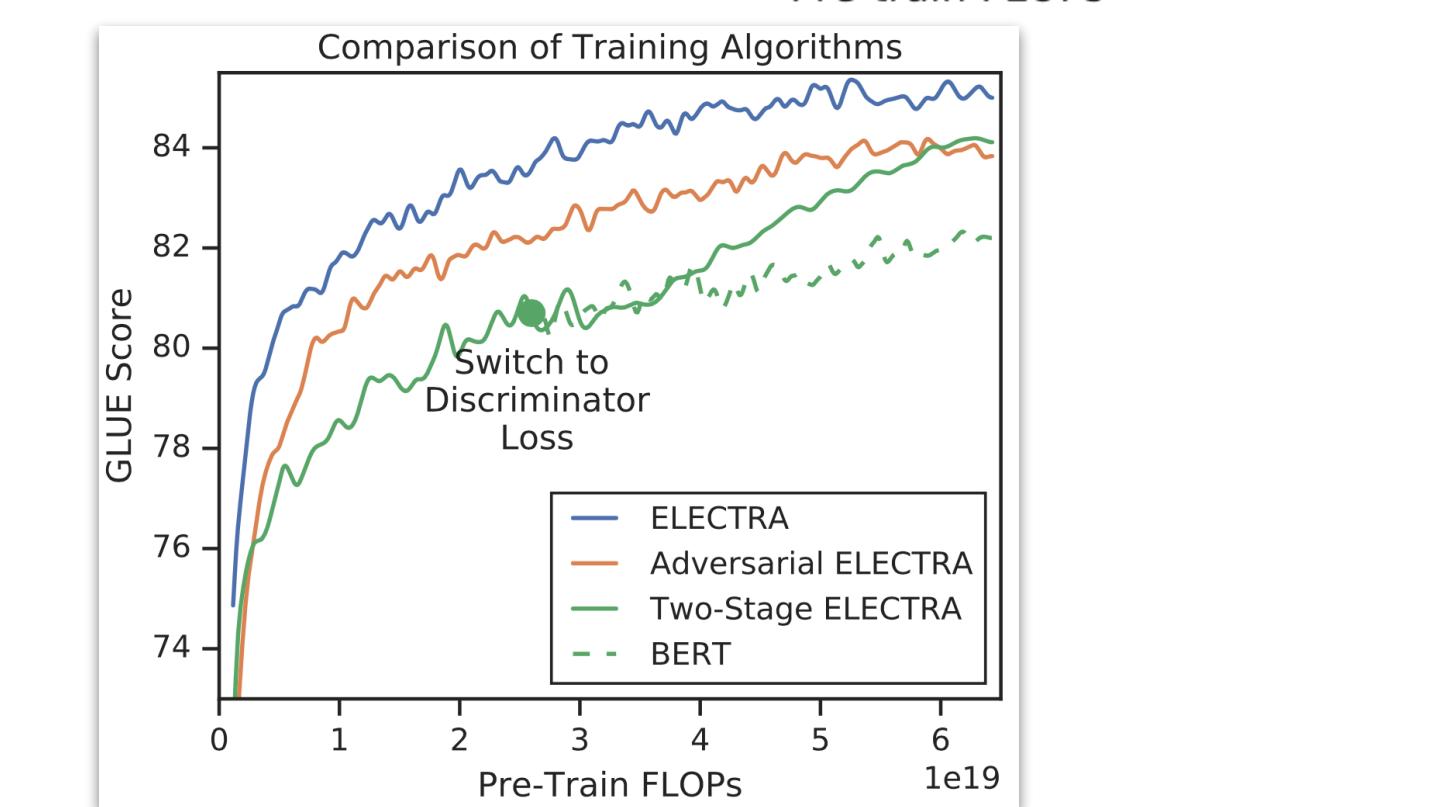
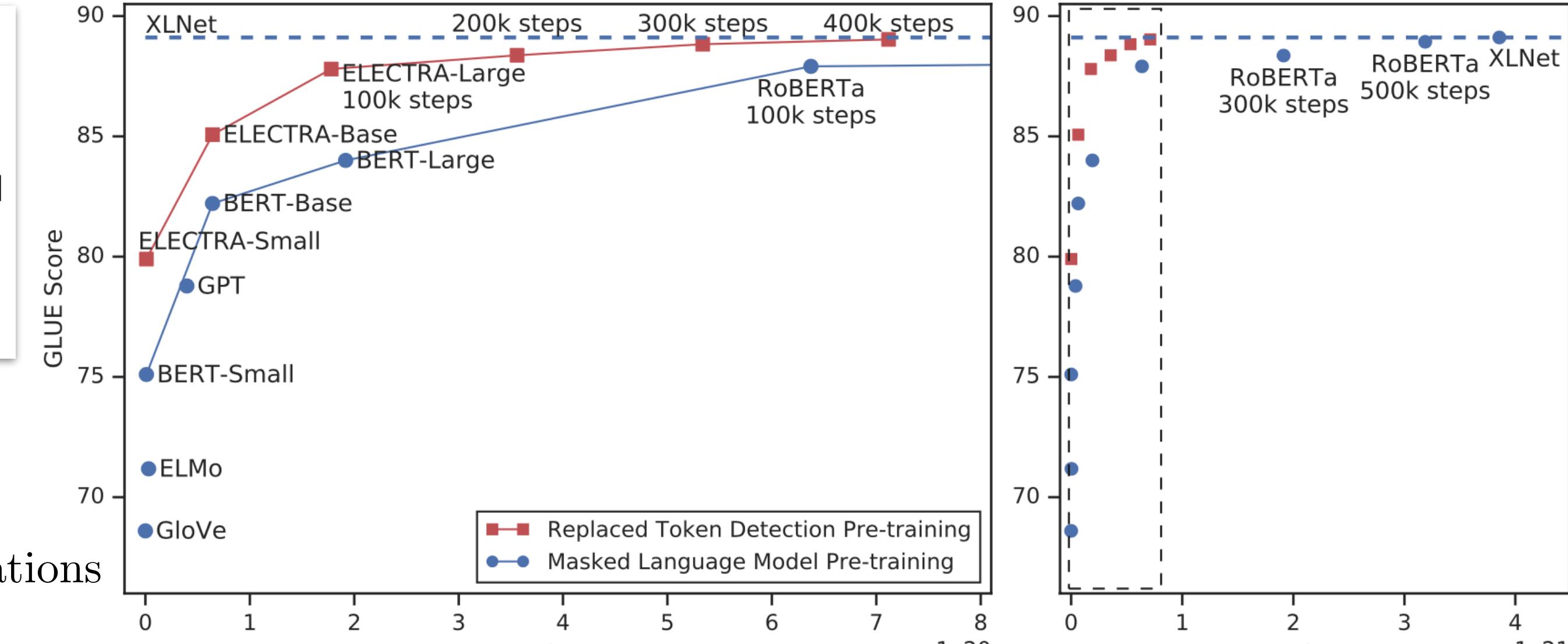
$k = \lceil 0.15n \rceil \rightarrow$  15% of tokens are masked out

$m_i \sim \text{unif}\{1, n\}$  for  $i = 1, \dots, k$

$x^{\text{mask}} = \text{REPLACE}(x, m, [\text{MASK}])$

$\hat{x}_i \sim p_G(x_i|x^{\text{mask}})$  for  $i \in m$

$x^{\text{corrupt}} = \text{REPLACE}(x, m, \hat{x})$



$$\min_{\theta_G, \theta_D} \sum_{x \in \mathcal{X}} \mathcal{L}_{\text{MLM}}(x, \theta_G) + \lambda \mathcal{L}_{\text{Disc}}(x, \theta_D)$$

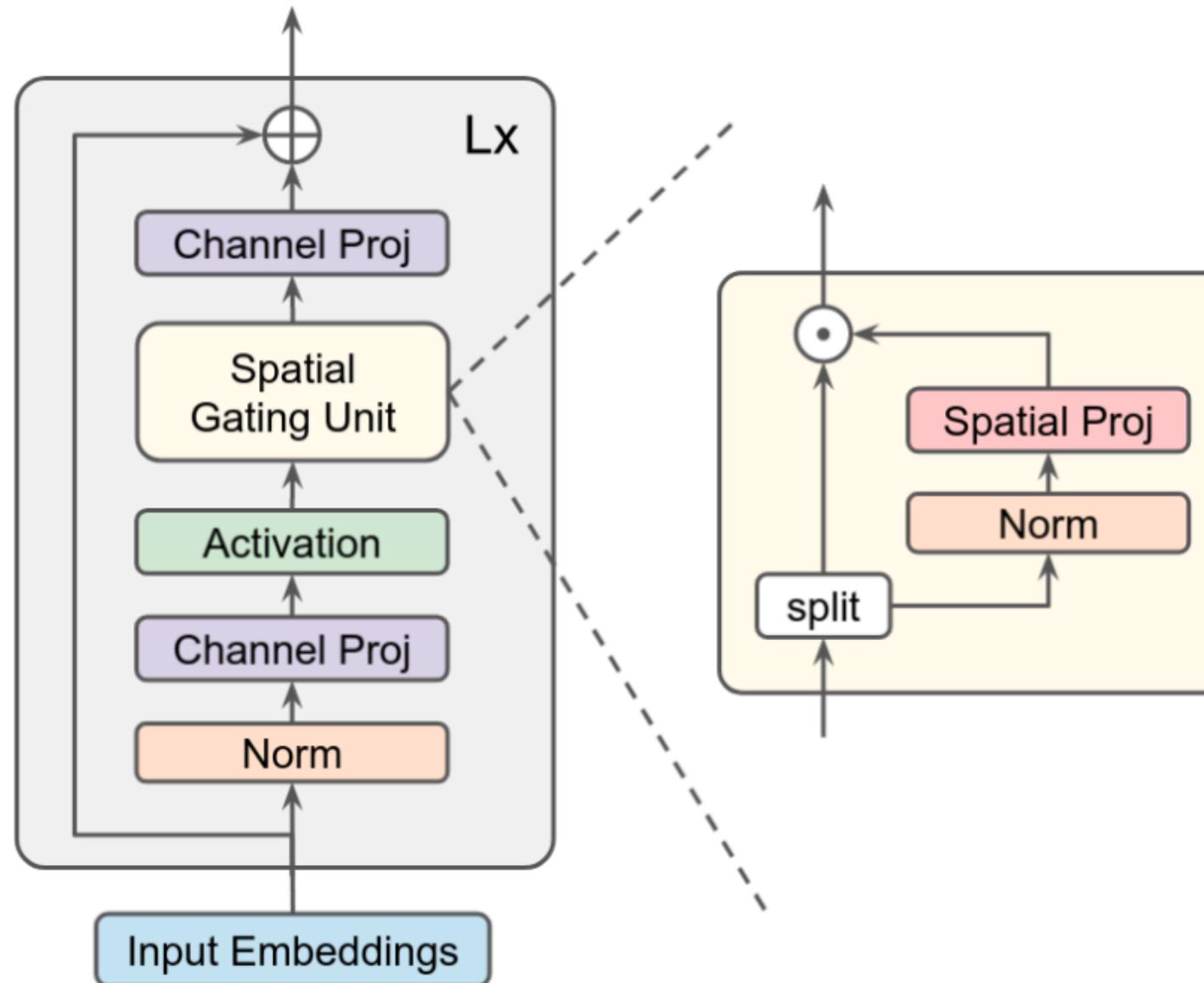
$$\mathcal{L}_{\text{MLM}}(x, \theta_G) = \mathbb{E} \left( \sum_{i \in m} -\log p_G(x_i | x^{\text{mask}}) \right)$$

$$\mathcal{L}_{\text{Disc}}(x, \theta_D) = \mathbb{E} \left( \sum_{t=1}^n -\mathbb{1}(x_t^{\text{corrupt}} = x_t) \log D(x^{\text{corrupt}}, t) - \mathbb{1}(x_t^{\text{corrupt}} \neq x_t) \log (1 - D(x^{\text{corrupt}}, t)) \right)$$



Boulder

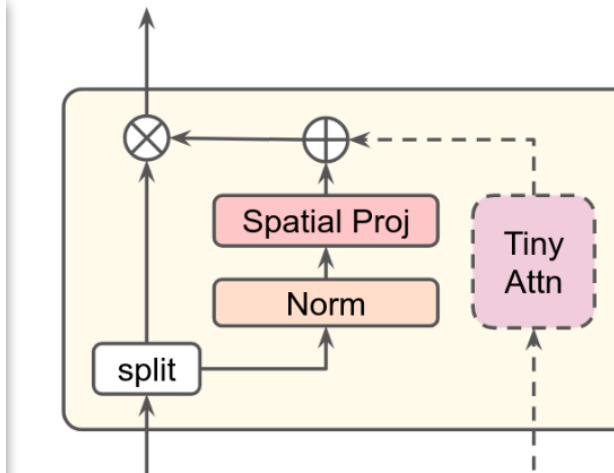
# Pay Attention to MLPs



Pseudo-code for the gMLP block

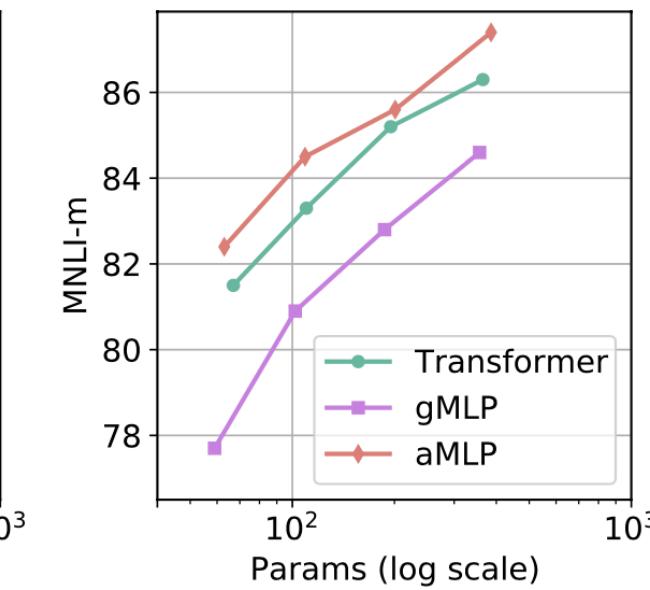
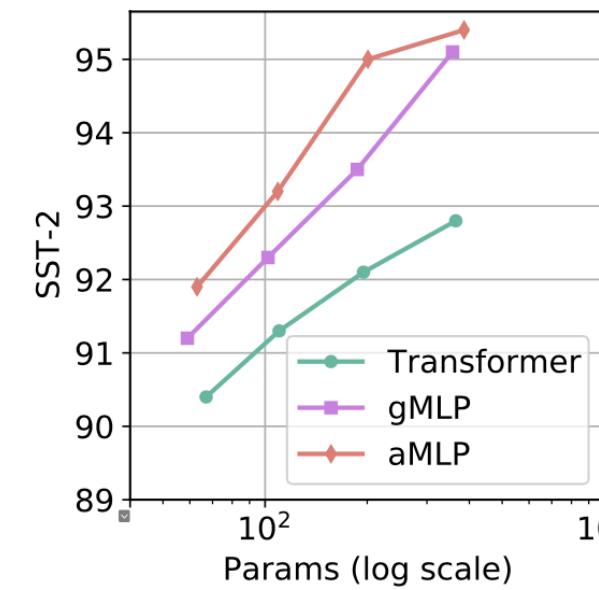
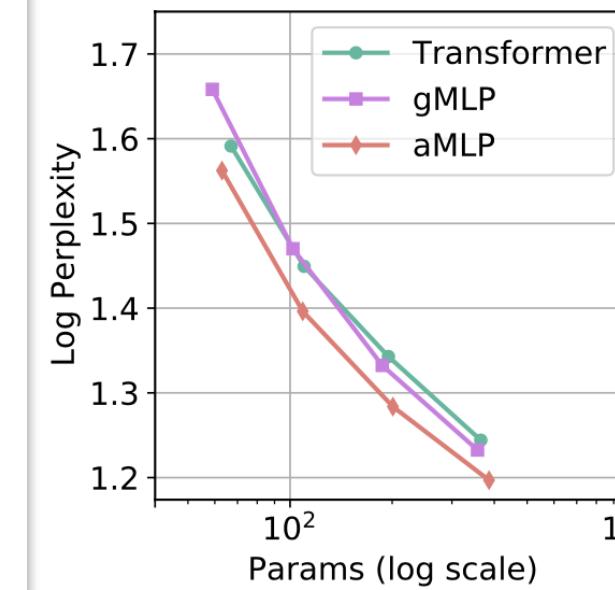
```
def gmlp_block(x, d_model, d_ffn):
    shortcut = x
    x = norm(x, axis="channel")
    x = proj(x, d_ffn, axis="channel")
    x = gelu(x)
    x = spatial_gating_unit(x)
    x = proj(x, d_model, axis="channel")
    return x + shortcut

def spatial_gating_unit(x):
    u, v = split(x, axis="channel")
    v = norm(v, axis="channel")
    n = get_dim(v, axis="spatial")
    v = proj(v, n, axis="spatial", init_bias=1)
    return u * v
```



Pseudo-code for the tiny attention module

```
def tiny_attn(x, d_ffn, d_attn=64):
    qkv = proj(x, 3 * d_attn, axis="channel")
    q, k, v = split(qkv, 3, axis="channel")
    w = einsum("bnd,bmd->bnm", q, k)
    a = softmax(w * rsqrt(d_attn))
    x = einsum("bnm,bmd->bnd", a, v)
    return proj(x, d_ffn, axis="channel")
```



$L$  blocks with identical size and structure

$X \in \mathbb{R}^{n \times d} \rightarrow$  token representations

$n \rightarrow$  sequence length

$d \rightarrow$  dimension

For brevity, let's omit shortcuts, normalizations and biases!

$Z = \sigma(XU)$ ,  $\sigma \rightarrow$  GeLU (Gaussian error Linear Unit)

$\tilde{Z} = s(Z) \rightarrow$  spatial interactions

$Y = \tilde{Z}V$

$U, V \rightarrow$  linear projections across the channel dimension

**Spatial Gating Unit (SGU)**

cross-token interactions

$f_{W,b}(Z) = WZ + b, W \in \mathbb{R}^{n \times n}$

$$Z = (Z_1, Z_2)$$

$$s(Z) = Z_1 \odot f_{W,b}(Z_2)$$

gMLP: MLP with a small attention  
aMLP: gMLP with a small attention

	Perplexity	SST-2	MNLI (m/mm)	SQuAD		Attn Size	Params (M)
				v1.1	v2.0		
BERT <sub>base</sub> [2]	–	92.7	84.4/-	88.5	76.3	768 (64 × 12)	110
BERT <sub>base</sub> (ours)	4.17	93.8	85.6/85.7	90.2	78.6	768 (64 × 12)	110
gMLP <sub>base</sub>	4.28	94.2	83.7/84.1	86.7	70.1	–	130
aMLP <sub>base</sub>	3.95	93.4	85.9/85.8	90.7	80.9	64	109
BERT <sub>large</sub> [2]	–	93.7	86.6/-	90.9	81.8	1024 (64 × 16)	336
BERT <sub>large</sub> (ours)	3.35	94.3	87.0/87.4	92.0	81.0	1024 (64 × 16)	336
gMLP <sub>large</sub>	3.32	94.8	86.2/86.5	89.5	78.3	–	365
aMLP <sub>large</sub>	3.19	94.8	88.4/88.4	92.2	85.4	128	316



Boulder

# Evaluating Large Language Models Trained on Code

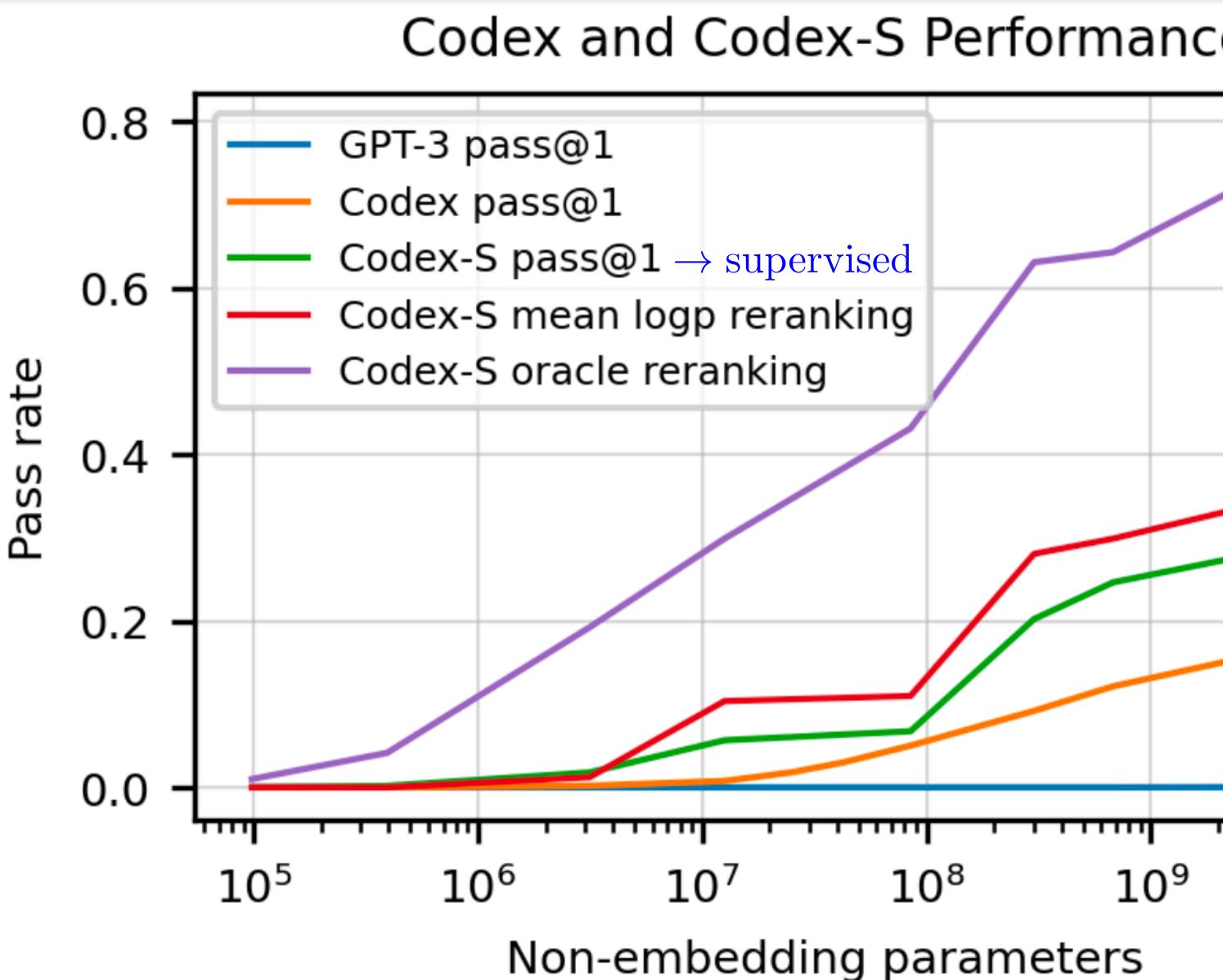
**Codex:** A GPT language model fine-tuned on publicly available code from GitHub

**GitHub Copilot:** Powered by a distinct production version of Codex

**Program synthesis:** Generating programs from docstrings

```
def solution(lst):
    """Given a non-empty list of integers, return the sum of all of the odd elements
    that are in even positions.

Examples
solution([5, 8, 7, 1]) =>12
solution([3, 3, 3, 3, 3]) =>9
solution([30, 13, 24, 321]) =>0
"""
    return sum(lst[i] for i in range(0, len(lst)) if i % 2 == 0 and lst[i] % 2 == 1)
```



## Evaluation Framework

HumanEval dataset: 164 hand-written programming problems

<https://www.github.com/openai/human-eval>

Functional correctness (i.e., passing a set of unit tests) v.s. match-based metrics  
(e.g., CodeBLEU score)

**pass@ $k$**  →  $k$  code samples are generated per problem, a problem is considered solved if any sample passes the unit tests, and the total fraction of problems solved is reported.

This could have high-variance. Instead, an unbiased estimator is used (see the paper!).

## Fine Tuning on Code

GPT-3 text tokenizer in addition to a set of tokens for representing whitespace runs of stop sequences: '\nclass', '\ndef', '\n#', '\nif', or '\nprint'

different lengths

Nucleus sampling instead of beam search

## Limitations

Codex is not sample efficient to train. It is trained on hundreds of millions of lines of code.

Codex can recommend syntactically incorrect or undefined code, and can invoke functions, variables, and attributes that are undefined or outside the scope of the codebase. Moreover, Codex struggles to parse through increasingly long and higher-level or system-level specifications. Codex can make mistakes binding operations to variables, especially when the number of operations and variables in the docstring is large.

```
def do_work(x, y, z, w):
    """ Add 3 to y, then subtract 4
    from both x and w. Return the
    product of the four numbers. """
    t = y + 3
    u = x - 4
    v = z * w
    return v
```

A system is **misaligned** if there's some task X that we want it to do, and it is "capable" of doing X but "chooses" not to. In contrast, if a system fails to do X because it does not have the ability to do so, then this system is not misaligned; it is just **incompetent**.



Boulder



# Questions?

[YouTube Playlist](#)

---