

# BERT

## Pre-training & Fine-Tuning



### Learning goals

- Know the pre-training tasks
- How to construct samples
- Understand the pre-training
- Gain understanding of the fine-tuning procedure
- Differences between token- and sequence classification

# MASKED LANGUAGE MODELING (MLM)

## First remark:

- Distinguish from *Masked Self-Attention*  
→ Masked Self-Attention is an architectural detail in the decoder of the Transformer, i.e. used by e.g. GPT
- Masked Self-Attention as a way to prevent causality issues in a Transformer decoder
- MLM is a self-supervised *modeling objective* introduced to couple Self-Attention and (deep) bidirectionality without violating causality

# MLM (2)

- **Training objective:**

Given a sentence, predict [MASK] ed tokens

- **Generation of samples:**

Randomly replace\* a fraction of the words by [MASK]

\*Sample 15% of the tokens; replace 80% of them by [MASK], 10% by a random token & leave 10% unchanged

- **Input:**

The	quick	brown	[MASK]	jumps	over	the	[MASK]	dog	.
-----	-------	-------	--------	-------	------	-----	--------	-----	---

- **Targets:**

(*fox, lazy*)

# MLM (3)

- ⚠ But 20% of the samples will look different!  
→ 10% replaced by a random token; 10% left unchanged
- **Input (first case):**

The	quick	brown	orange	jumps	over	the	is	dog	.
-----	-------	-------	--------	-------	------	-----	----	-----	---

- **Targets (first case):**

*(fox, lazy)*

- **Input (second case):**

The	quick	brown	fox	jumps	over	the	lazy	dog	.
-----	-------	-------	-----	-------	------	-----	------	-----	---

- **Targets (second case):**

*(fox, lazy)*

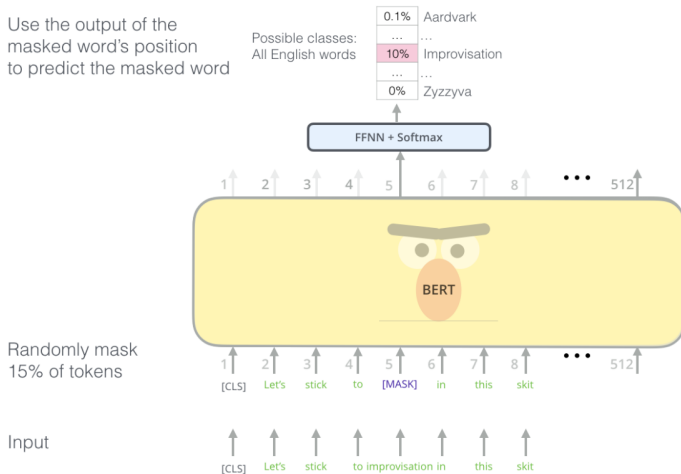
# MLM (4)

## Discrepancy between pre-training & fine-tuning:

- [MASK] -token as central part of pre-training procedure
- [MASK] -token does not occur during fine-tuning
- **Modified pre-training task:**  
Predict 15% of the tokens of which only 80% have been replaced by [MASK]
  - 80% of the selected tokens are actually [MASK] ed
  - 10% of the selected tokens: The model has to “understand” that the word needs to be *replaced*
  - 10% of the selected tokens: The model has to “understand” that the word needs to be *kept*
- This ensures overlap between the kind data seen during pre-training and during fine-tuning

# MLM (5)

Use the output of the masked word's position to predict the masked word



► Source: Jay Alammar

# NEXT SENTENCE PREDICTION (NSP)

- **Training objective:**

Given two sentences, predict whether  $s_2$  follows  $s_1$

- **Generation of samples:**

Randomly sample\* negative examples (cf. word2vec)

\*50% of the time the second sentence is the actual next sentence, 50% of the time it is a randomly sampled sentence

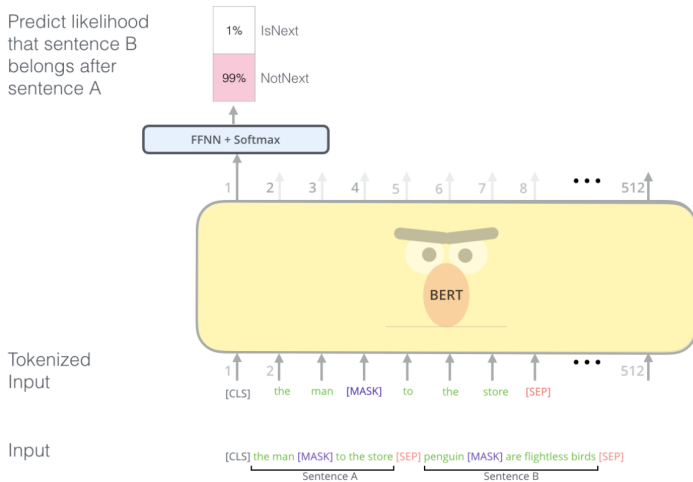
- **Full Input:**

[CLS]	The	[MASK]	is	quick	.	[SEP]	
It	jumps	over	the	[MASK]	dog	.	[SEP]

- [CLS] token as sequence representation for classification
- [SEP] token for separation of the two input sequences

# NSP (2)

Predict likelihood  
that sentence B  
belongs after  
sentence A



► Source: Jay Alammar



# PRE-TRAINING BERT

## Setup:

- 13 GB of text (BooksCorpus + Eng. Wikipedia)  
→ Back then: **huge**; Now: rather **small**
- Train for approximately\* 40 epochs
- 4 (16) Cloud TPUs for 4 days for the BASE (LARGE) variant
- Loss function:

$$Loss_{BERT} = Loss_{MLM} + Loss_{NSP}$$

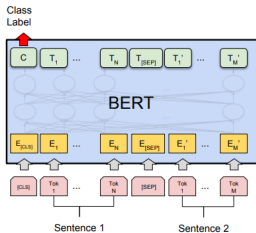
\*1.000.000 steps on batches of 256 sequences with a sequence length of 512 tokens

# PRE-TRAINING BERT

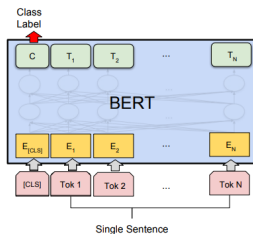
## Sequence lengths:

- For their experiments:
  - Pre-train w/ sequence length 128 for 90% of the steps
  - Pre-train w/ sequence length 512 for 10% of the steps
- *Reason:* Positional embeddings
  - Learned, not sinusoidal (as opposed to the original Transformer)
  - Training on long sequences necessary to learn them well
  - **But:** Training expensive, hence this “compromise”

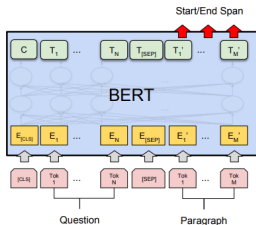
# FINE-TUNING BERT



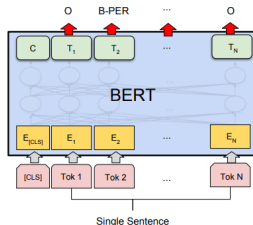
(a) Sentence Pair Classification Tasks:  
MNLI, QQP, QNLI, STS-B, MRPC,  
RTE, SWAG



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



(c) Question Answering Tasks:  
SQuAD v1.1



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

► Source: Devlin et al., 2019

# FINE-TUNING BERT

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
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
BERT <sub>BASE</sub>	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT <sub>LARGE</sub>	<b>86.7/85.9</b>	<b>72.1</b>	<b>92.7</b>	<b>94.9</b>	<b>60.5</b>	<b>86.5</b>	<b>89.3</b>	<b>70.1</b>	<b>82.1</b>

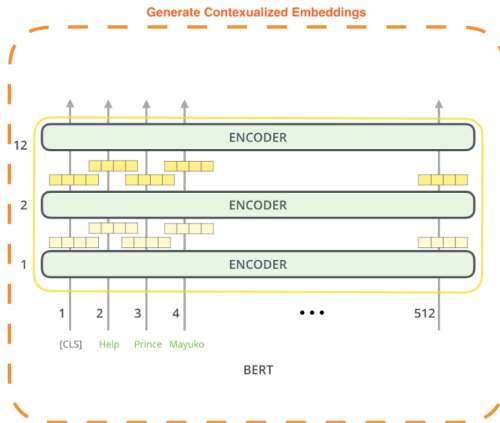
► Source: Devlin et al., 2019

- Performance of BERT on the GLUE Benchmark ► Wang et al., 2018
- Beats all of the previous state-of-the-art models
- In the meantime: Other models (way) better than BERT

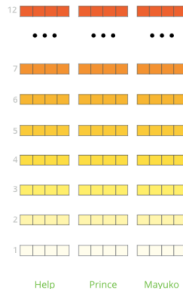
# FINE-TUNING DETAILS

- Relatively cheap compared to pre-training:
  - $< 1$  hour on a single Cloud TPU
  - "a few hours" on a GPU
- Recommendations for hyperparameters:
  - **Batch Size:** 16, 32
  - **Adam learning rate:**  $5e-5$ ,  $3e-5$ ,  $2e-5$
  - **#epochs:** 2, 3, 4
  - **Dropout probability:** 0.1
- Data sets w/  $> 100k$  labeled examples rather insensitive to hyperparameters

# FEATURE EXTRACTION FROM BERT



The output of each encoder layer along each token's path can be used as a feature representing that token.



But which one should we use?

► Source: Jay Alammar

# FEATURE EXTRACTION FROM BERT

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	<b>93.1</b>
Fine-tuning approach		
BERT <sub>LARGE</sub>	96.6	92.8
BERT <sub>BASE</sub>	96.4	92.4
Feature-based approach (BERT <sub>BASE</sub> )		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

Table 7: CoNLL-2003 Named Entity Recognition results. Hyperparameters were selected using the Dev set. The reported Dev and Test scores are averaged over 5 random restarts using those hyperparameters.

► Source: Devlin et al., 2019