Google AI

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

Ming-Wei Chang, Google AI Language

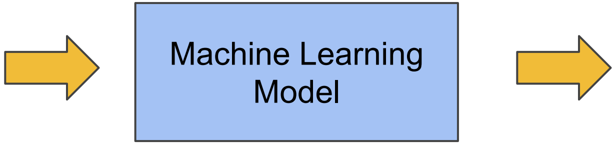
INTRODUCTION

Natural Language Processing (NLP)

- Enabling computers to process natural language
  - E.g. sentiment analysis, question answering, ...
- Example: Question Answering

Q: The traveling salesman problem is an example of what type of problem?

P: A function problem is a computational problem ... Notable examples include the traveling salesman problem and the integer factorization problem.



A: A function problem is a computational problem where a single output …. Notable examples include the traveling salesman problem and the integer factorization problem.

The Language Representation Problem

- Q: How to represent text for machine learning models?

- Many machine learning models (such as neural networks) expect continuous vectors as input

- The representations should capture the semantic meaning

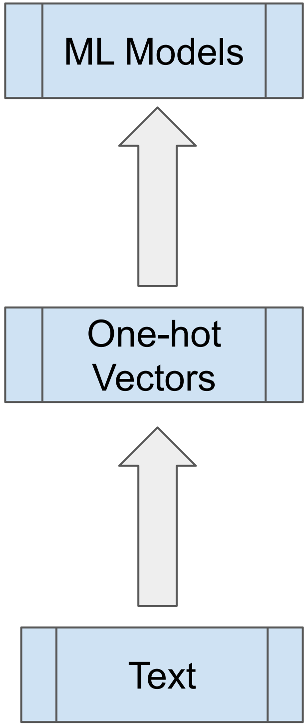
How Should We Represent Words?

Can we do better?

- A one-hot vector? Traditional NLP

```
king                      queen
 ↓                          ↓
[0, 1, 0, 0, …]           [0, 0, 1, 0 …]
```

- Distances between any two words …
  - are always the same!
  - However, "queen" should be more related to "king" compared to "headphone"



Continuous Representations of Words

Can we do better?

- Representing words with continuous values

`king` → `[-0.5, -0.9, 1.4, …]`

`queen` → `[-0.6, -0.8, -0.2, …]`

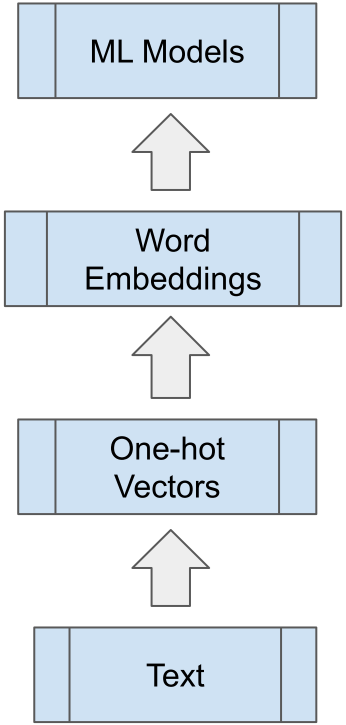
- **Word embeddings** (`word2vec`, `GloVe`) are often *pre-trained* on **unlabeled** text corpus from co-occurrence statistics

Inner Product

`the king wore a crown`

Inner Product

`the queen wore a crown`



Word2Vec [mikolov et-al, 13], PLSI [Hoffman, 99], GloVe [Pennington et-al, 14],

Contextual Representations

- **Problem** of word embeddings: context-independent

```
open a bank account                    on the river bank

                 [0.3, 0.2, -0.8, …]
```

- Ideally, representations should be contextual

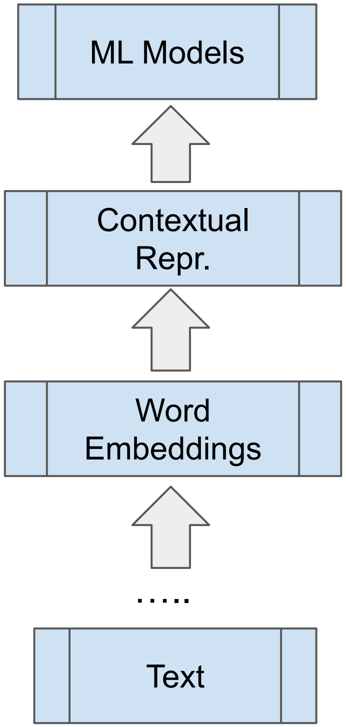
```
[0.9, -0.2, 1.6, …]                    [-1.9, -0.4, 0.1, …]

open a bank account                    on the river bank
```

Training Contextual Representations

Can we do better?

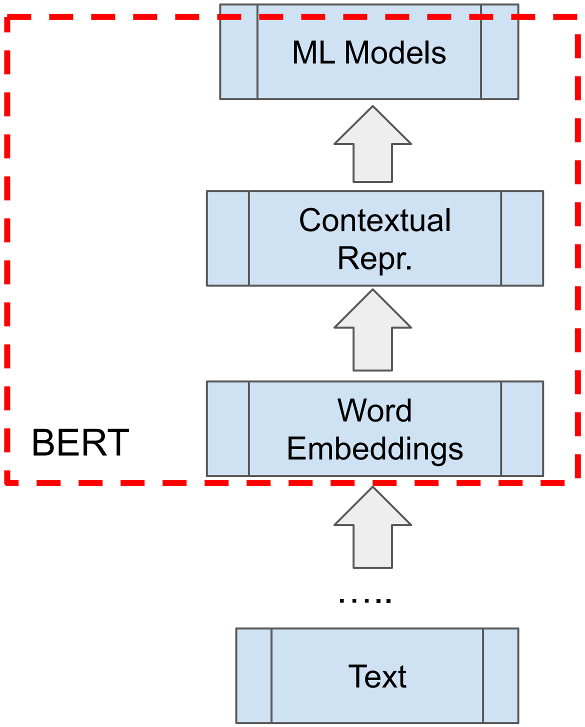
- Pre-training contextual representations:
  - Semi-Supervised Sequence Learning, Google, 2015
  - ELMo, AI2, 2017
  - Generative Pre-Training, OpenAI, 2018
  - ULMFit, fast.ai, 2018
- Training language models on text corpus:
  - Generate contextual representations. Single direction.

```
P(open a bank account) = P(open)*P(a|open)*P(bank|open, a)*
                         P(account|open, a, bank)
```



BERT!

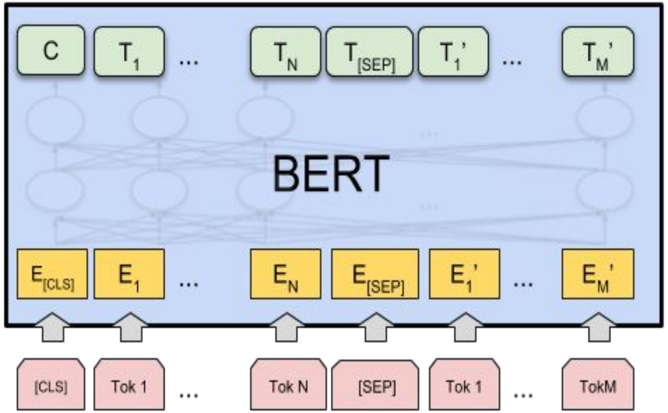
- Deep Bi-directional Pre-training
  - Using Transformer Blocks
- Learning contextual representations
  - With unlabeled data
  - Not just embeddings; Model initialization.
- Pre-training is very powerful
  - State-of-the-art performance for 11 tasks
  - With little task-specific engineering



Input and Output for BERT

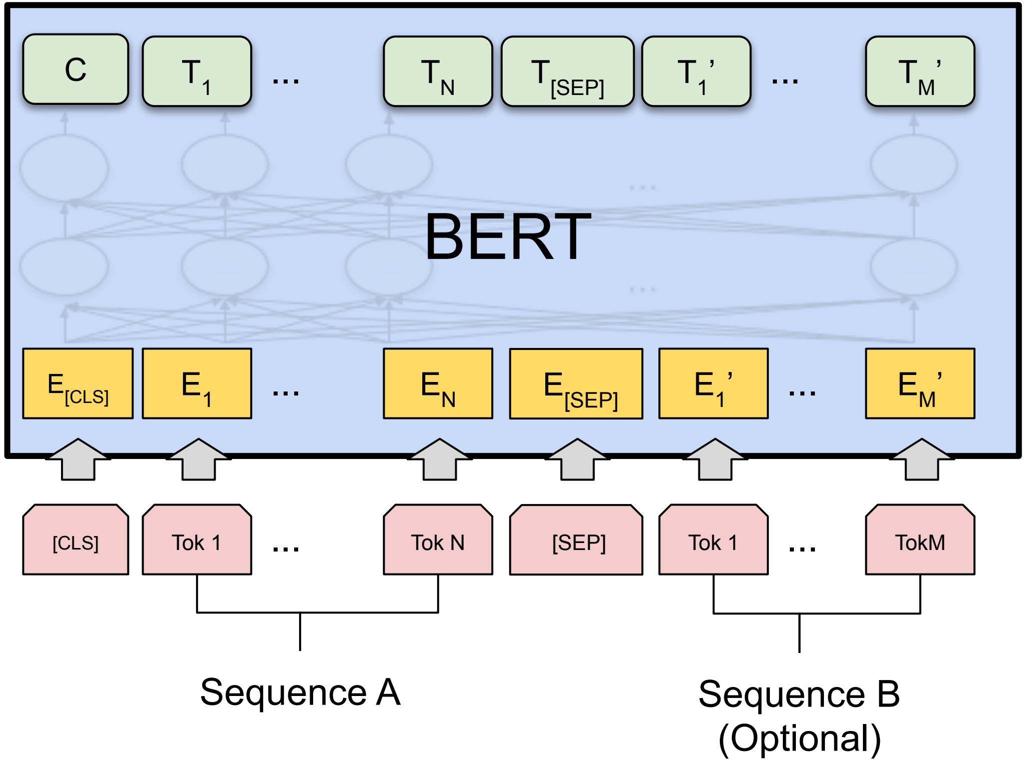
BERT Model

- Every token will be translated into a representation vector
- Bi-directional Transformer [Vaswani et. al 17]
- What is the input/output for BERT?



Transformer

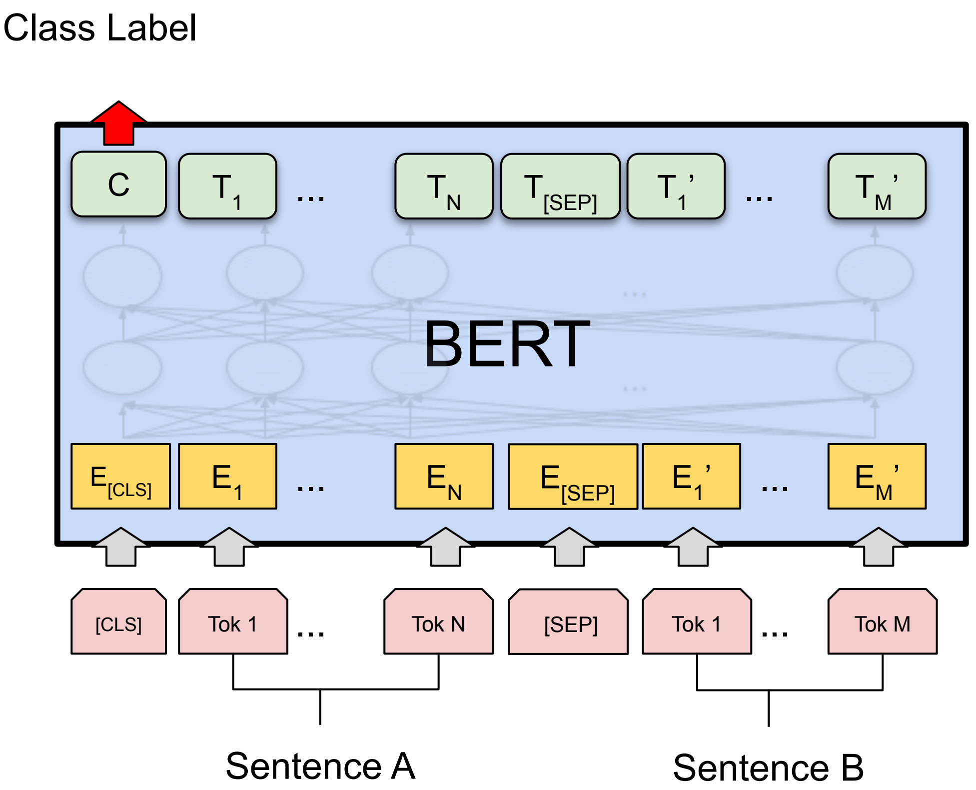
Unified Input Representation



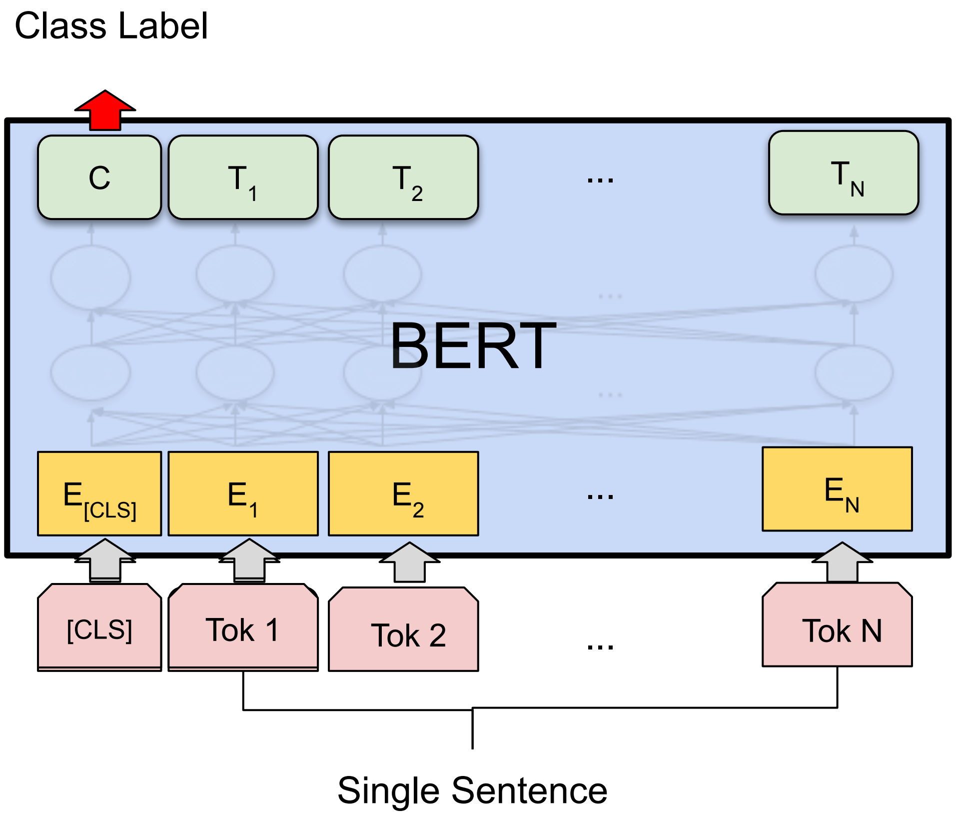
Always represent the input as a long sequence

- For text pair, pack two sentences in one sequence
- For single text task (such as) classification or tagging, pack one sentence in one sequence

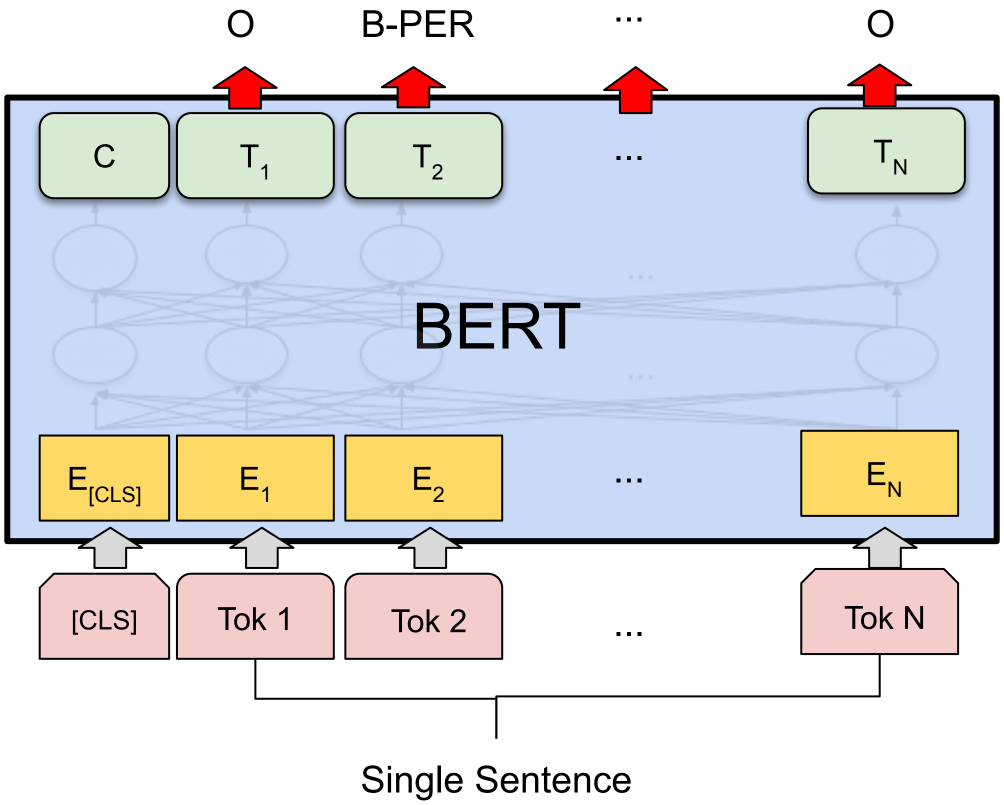
Sentence Pair Classification Tasks



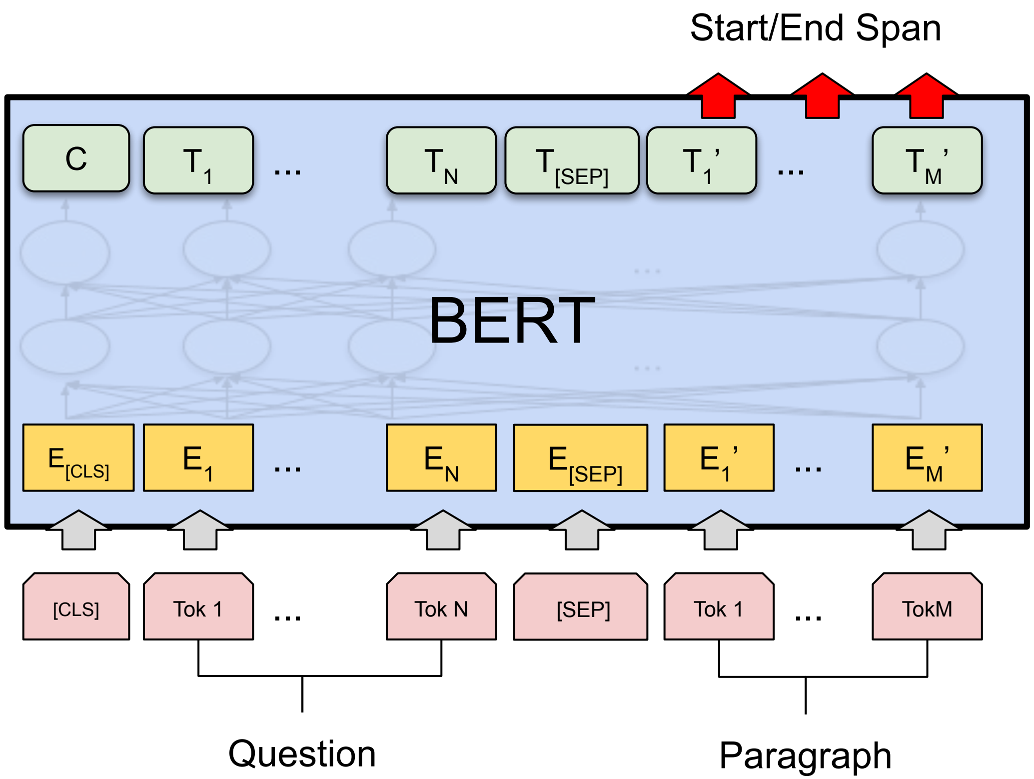
Single Sentence Classification Tasks



Sequence Tagging



BERT for Question Answering



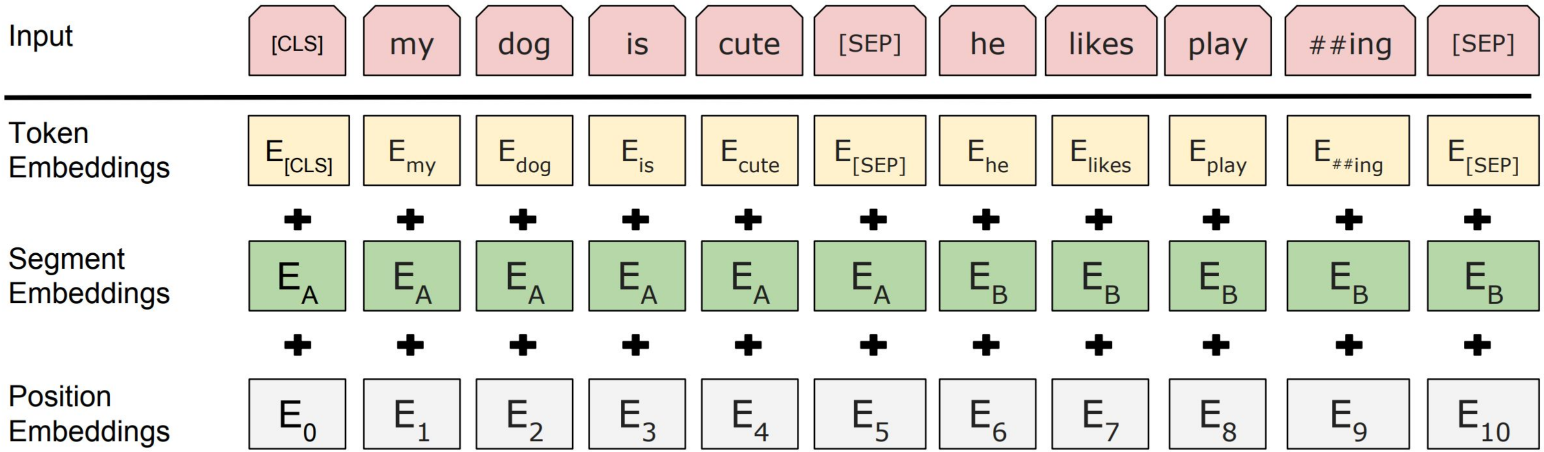
Task List

- Different tasks have different input/output

Input \ Output	Classification	Token-level Output
Single Text Sequence	e.g. Sentiment Classification	e.g. Named Entity Recognition
Text Sequence Pairs	e.g. Entailment	e.g. Question Answering

- The [CLS] and [SEP] tokens and the unified input format make it is possible to use the same architecture

Input Representation Details

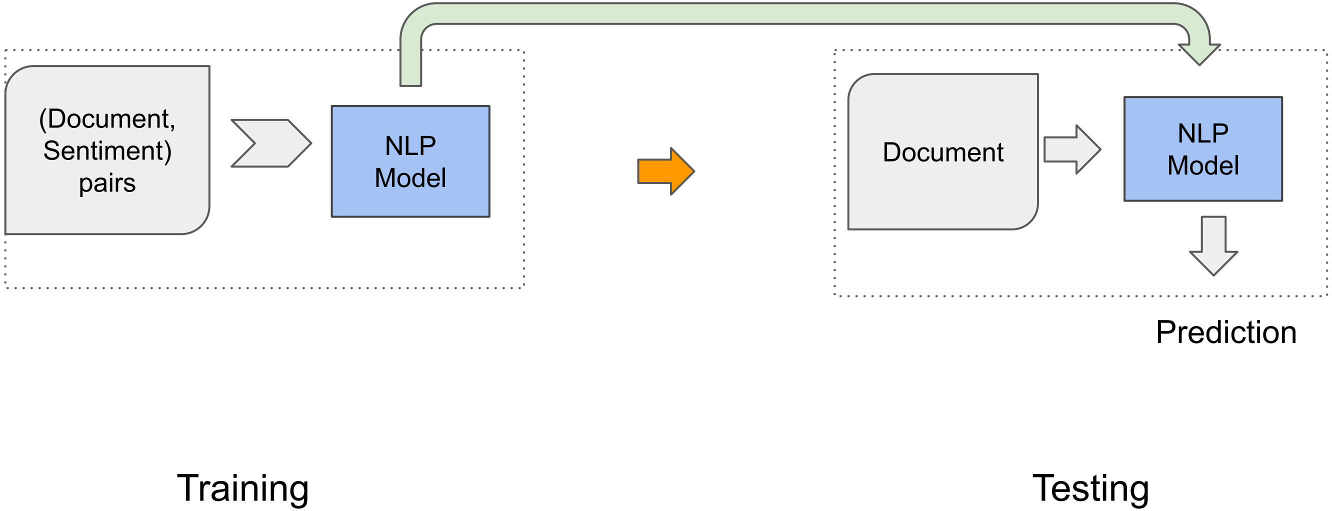


- Use 30,000 WordPiece vocabulary on input.
- Each token is sum of three embeddings

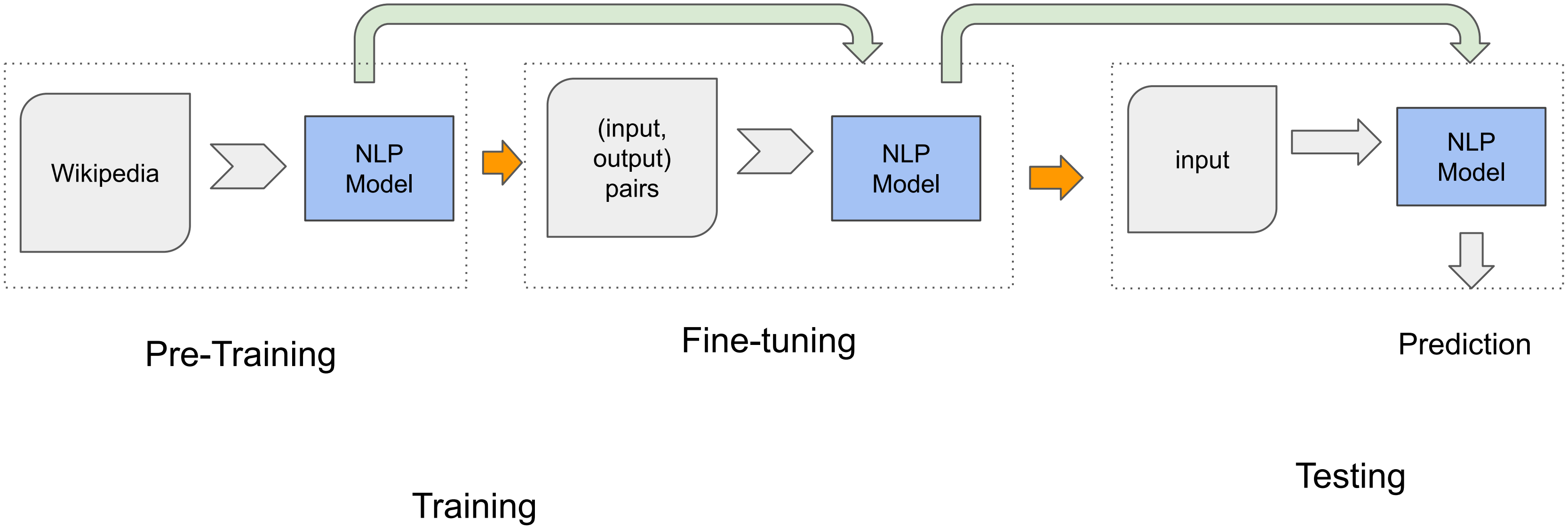
Pre-training BERT

Review: Supervised Training

- Labeled data: (input, output) pairs



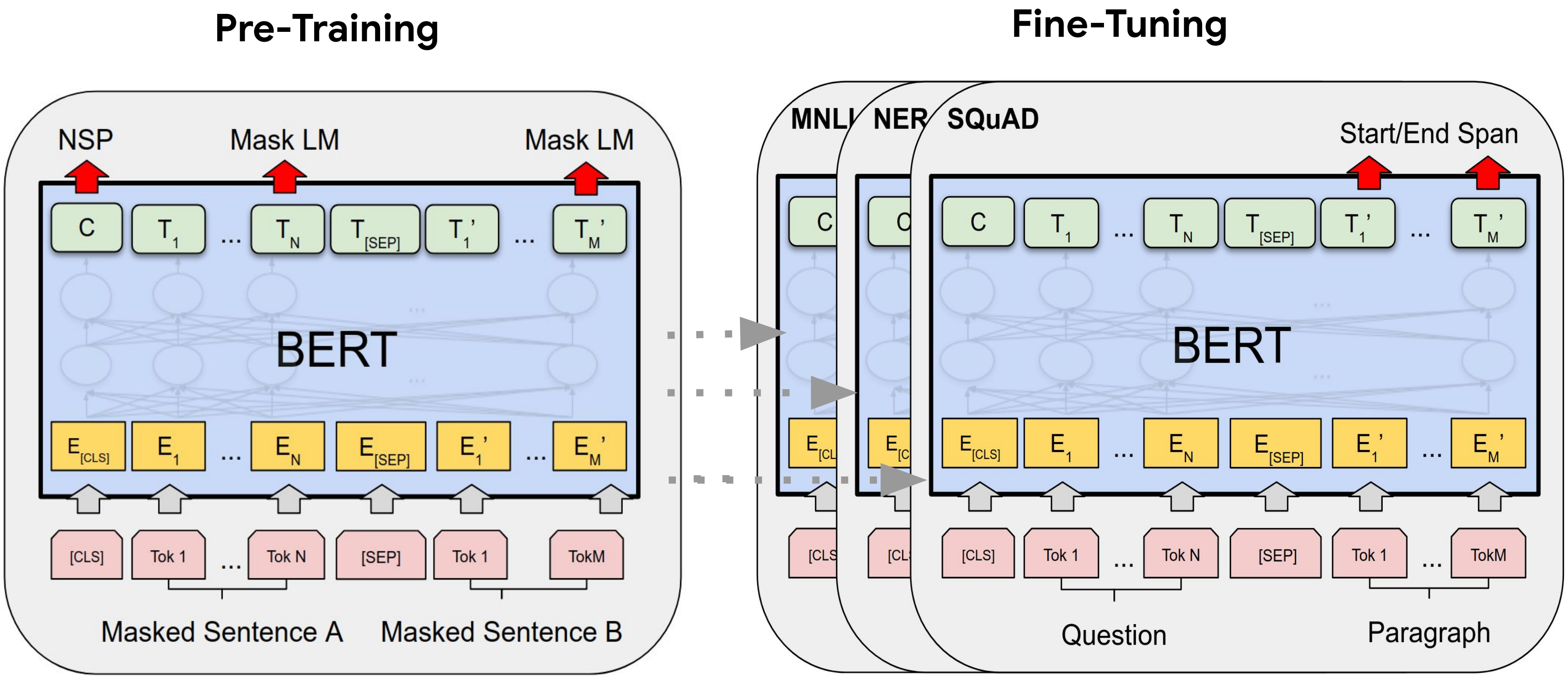
Pre-training for BERT



Terminologies

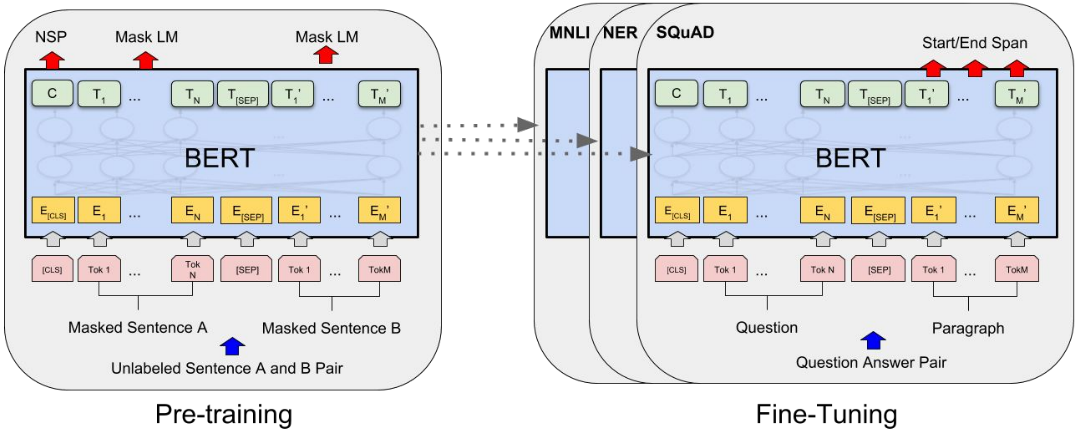
- Pre-training
  - Training BERT with a large amount of unlabeled data
- Fine-tuning
  - Training BERT with a small amount of task-specific labeled data
- In BERT, we use pre-training as a way to initialize the whole model. This is different from just using fixed word embeddings.

Basic BERT Overall Framework



Notes in the BERT Framework

- We always use the same model architecture
- We initialize all parameters from pre-training model
- We fine-tune all parameters in the fine-tuning stages



Pre-training: Masked LM

- **Solution**: Mask out *k*% of the input words, and then predict the masked words
    - We always use k = 15%

```
                             store              gallon
                               ↑                  ↑
the man went to the [MASK] to buy a [MASK] of milk
```

- Too little masking: Too expensive to train
- Too much masking: Not enough context

Pre-training: Next Sentence Prediction

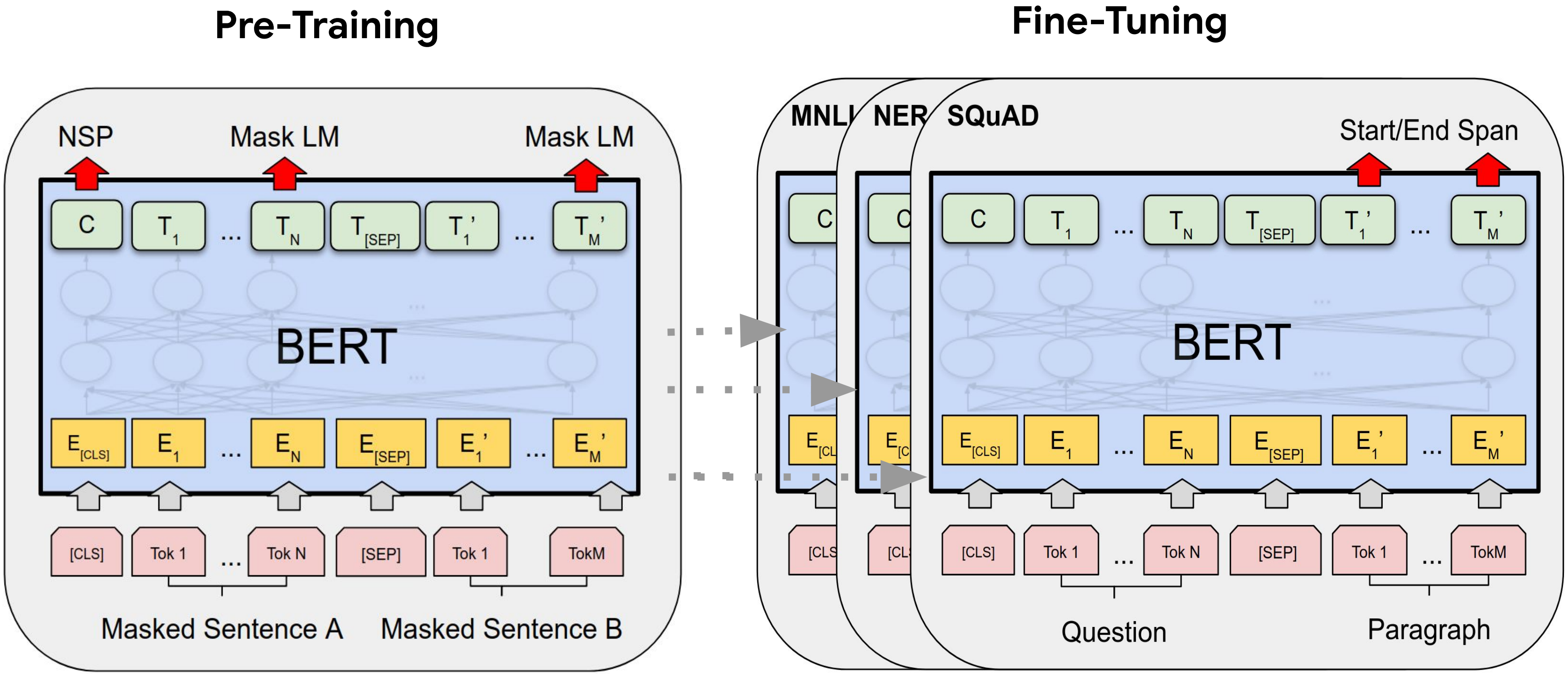
- To learn *relationships* between sentences, predict whether Sentence B is actual sentence that follows Sentence A, or a random sentence

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`

Google AI

Transfer Learning; Not Multi-task Learning



Model Details

- Public BERT: Train on 3.3B words for 40 epochs
- `BERT-Base`: 12-layer, 768-hidden, 12-head
- `BERT-Large`: 24-layer, 1024-hidden, 16-head
- Trained on TPU for 4 days
- Pre-Trained models released and ready to use!

RESULTS

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:
At the other end of Pennsylvania Avenue, people began to line up for a White House tour.
Hypothesis: People formed a line at the end of Pennsylvania Avenue.
Label: Entailment

CoLA

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

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

SQuAD 1.1 Results

What was another term used for the oil crisis?
Ground Truth Answers: first oil shock shock shock first oil shock shock
Prediction: shock

The 1973 oil crisis began in October 1973 when the members of the Organization of Arab Petroleum Exporting Countries (OAPEC, consisting of the Arab members of OPEC plus Egypt and Syria) proclaimed an oil embargo. By the end of the embargo in March 1974, the price of oil had risen from US$3 per barrel to nearly $12 globally; US prices were significantly higher. The embargo caused an oil crisis, or "shock", with many short- and long-term effects on global politics and the global economy. It was later called the "first oil shock", followed by the 1979 oil crisis, termed the "second oil shock."

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar et al. '16)	82.304	91.221
1 Oct 05, 2018	BERT (ensemble) *Google AI Language* https://arxiv.org/abs/1810.04805	**87.433**	**93.160**
2 Oct 05, 2018	BERT (single model) *Google AI Language* https://arxiv.org/abs/1810.04805	85.083	91.835
2 Sep 26, 2018	nlnet (ensemble) *Microsoft Research Asia*	85.954	91.677
5 Sep 09, 2018	nlnet (single model) *Microsoft Research Asia*	83.468	90.133
3 Jul 11, 2018	QANet (ensemble) *Google Brain & CMU*	84.454	90.490

SWAG Results

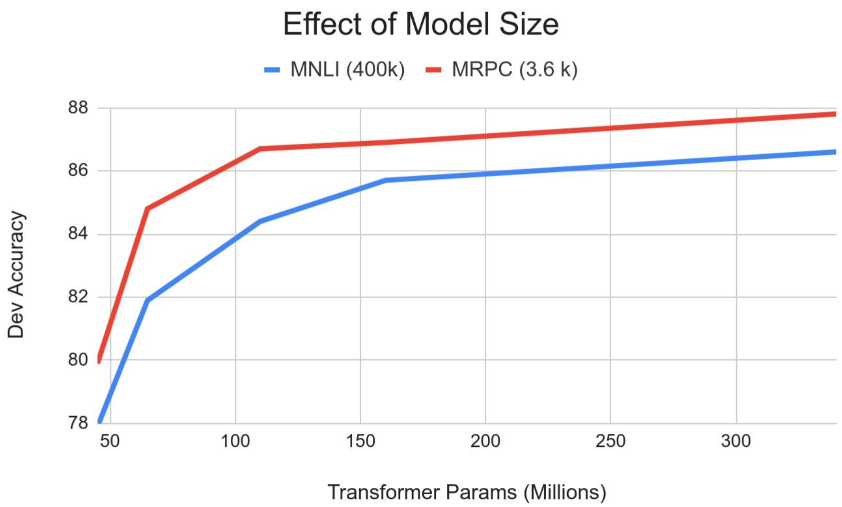
```
A girl is going across a set of monkey bars.  She
(i) jumps up across the monkey bars.
(ii) struggles onto the bars to grab her head.
(iii) gets to the end and stands on a wooden plank.
(iv) jumps up and does a back flip.
```

Leaderboard

Human Performance (88.00%)
Running Best
Submissions

Rank	Model	Test Score
1	**BERT (Bidirectional Encoder Representations from Transfo...** *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova* 10/11/2018	**86.28%**
2	**OpenAI Transformer Language Model** *Original work by Alec Radford, Karthik Narasimhan, Tim Salimans, ...* 10/11/2018	**77.97%**
3	**ESIM with ELMo** *Zellers, Rowan and Bisk, Yonatan and Schwartz, Roy and Choi, Yejin* 08/30/2018	**59.06%**
4	**ESIM with Glove** *Zellers, Rowan and Bisk, Yonatan and Schwartz, Roy and Choi, Yejin* 08/29/2018	**52.45%**

Effect of Model Size



- Big models help *a lot*
- Going from 110M -> 340M params helps even on datasets with 3,600 labeled examples
- Bigger model might even help more!

BERT is Open Sourced

- Both code and pre-trained models are available
  - Over 14,000 github stars
  - Multilingual models are also released
- Many articles and blog posts about BERT
- Large impact on other NLP tasks
  - ARC, CoQA, SQuAD 2.0, MSMARCO, OpenBookQA, SciTail, parsing

https://github.com/google-research/bert

Using BERT on TfHub

Install and Config BERT

- Install BERT

```
pip install bert-tensorflow
```

- Import BERT

```
import bert
from bert import run_classifier
from bert import optimization
from bert import tokenization
```

CoLab

Prepare Input for BERT

- For this task, we have only one text sequence

```
bert.run_classifier.InputExample(guid=None,
                                 text_a = x[DATA_COLUMN],
                                 text_b = None,
                                 label = x[LABEL_COLUMN])
```

Input Format

- Construct BERT input

```
bert.run_classifier.convert_examples_to_features(train_
InputExamples, label_list, MAX_SEQ_LENGTH, tokenizer)
```

```
INFO:tensorflow:tokens: [CLS] the performances were fault ##less and outstanding . [SEP]
INFO:tensorflow:input_ids: 101 1996 4616 2020 6346 3238 1998 5151 1012 102
INFO:tensorflow:input_mask: 1 1 1 1 1 1 1 1 1 1
INFO:tensorflow:segment_ids: 0 0 0 0 0 0 0 0 0 0
INFO:tensorflow:label: 1 (id = 1)
```

Use BERT TfHub Module

- Load BERT Tf-Hub Module

```
bert_module = hub.Module(BERT_MODEL_HUB, trainable=True)
bert_inputs = dict(input_ids=input_ids,
                   input_mask=input_mask,
                   segment_ids=segment_ids)
bert_outputs = bert_module(inputs=bert_inputs,
                           signature="tokens",
                           as_dict=True)
```

Specify the Output Layer

- Add the standard output layer for classification

```
with tf.variable_scope("loss"):
  logits = tf.layers.dense(output_layer, num_classes)
  loss = tf.losses.sparse_softmax_cross_entropy(labels, logits)
  predictions = tf.argmax(logits, -1)
```

Training and Evaluating with BERT

- Getting results

```
estimator.train(input_fn=train_input_fn, max_steps=num_train_steps)
…

estimator.evaluate(input_fn=test_input_fn, steps=None)
```

```
{'auc': 0.86659324,
 'eval_accuracy': 0.8664,
 'f1_score': 0.8659711,
```

Conclusion

- BERT is a novel language representation model
- The pretrained models and code are released!
  - https://github.com/google-research/bert
- Questions?