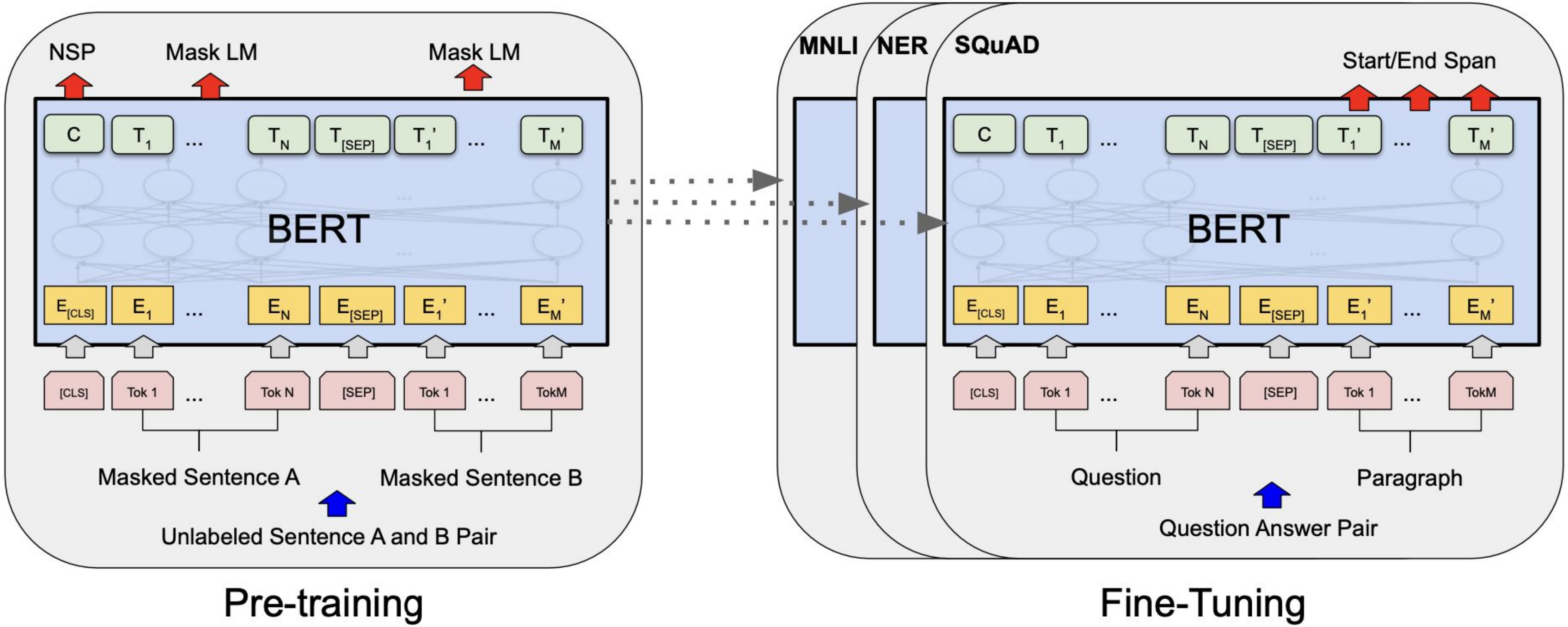
BERT Model in a Nutshell

Ashley Kumar

What it Looks Like:



Basically, it’s an encoder that represents words and sentences and attempts to “understand” relationship between sequences.

Bert vs GPT

- BERT, or Bidirectional Encoder Representations from Transformers, is an encoder ONLY.
    - Processes input sequence, constructs an “encoder hidden state” as an output
    - Think of as a representational map
- Trained on BooksCorpus (800M words) and Wikipedia (2.5B words), with batch size of 128,000 words
- Learns the separator/CLS tokens and A/B embeddings (see later) in pre-training.
- Task-specific fine-tuning learning rates selected

- GPT, or Generative Pre-Trained Transformer, is a decoder ONLY.
- Only left-to-right
    - Takes in encoder hidden state and has output layer and generates output sequence by using self-attention
- Trained on BooksCorpus (800M words) with a batch size of 32,000 words and same number of steps (100M)
- Uses the separator/CLS tokens and A/B embeddings during fine-tuning
- Same learning rate for all fine-tuning (5e-5).

Tokens

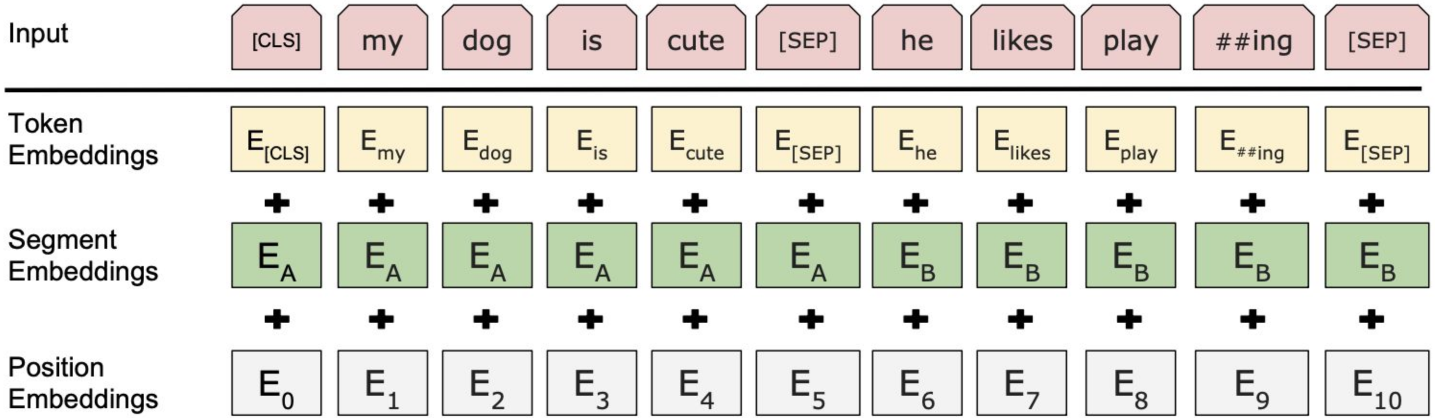
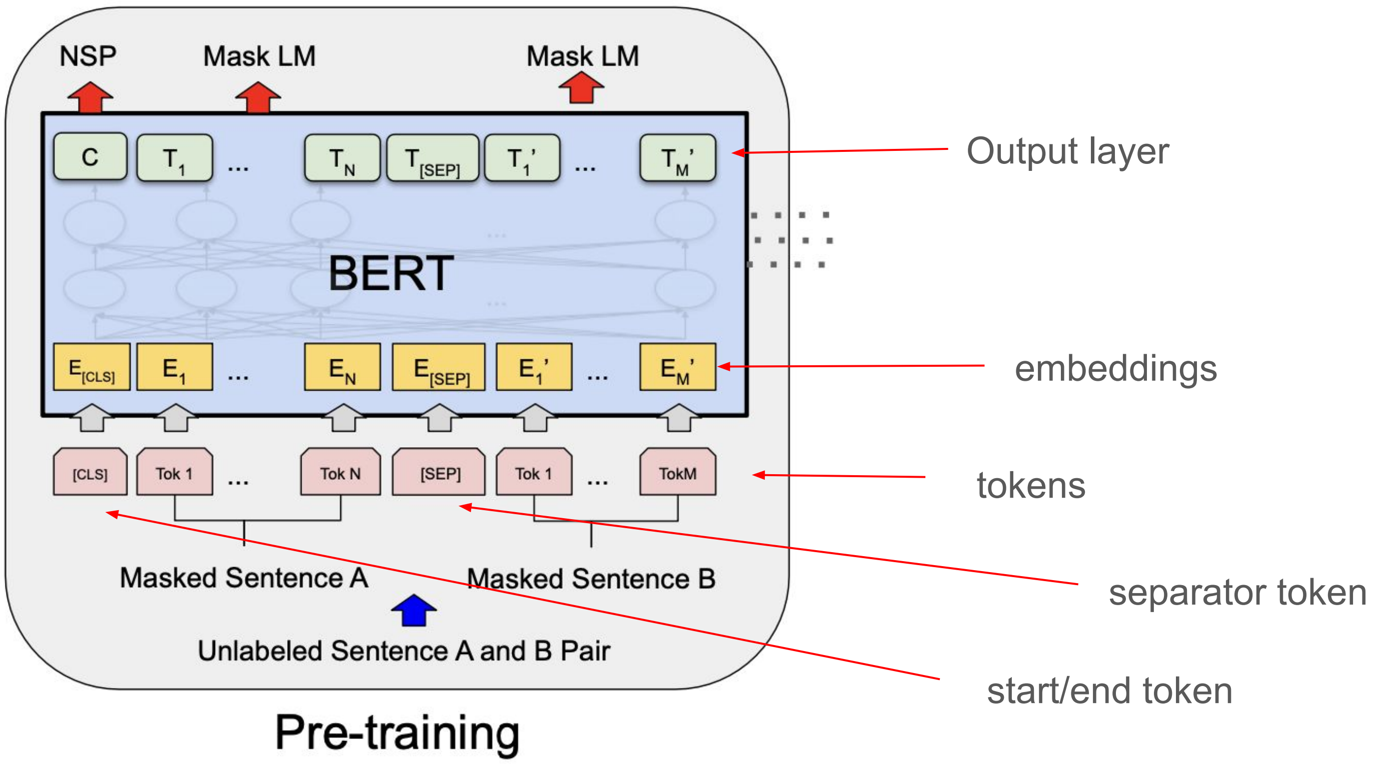


Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.


NSP
Mask LM
Mask LM
C
T1
...
TN
T[SEP]
T1'
...
TM'
BERT
E[CLS]
E1
...
EN
E[SEP]
E1'
...
EM'
[CLS]
Tok 1
...
Tok N
[SEP]
Tok 1
...
TokM
Masked Sentence A
Masked Sentence B
Unlabeled Sentence A and B Pair
Pre-training
Output layer
embeddings
tokens
separator token
start/end token

Well how does it work?

Pretraining!

- involves two tasks: masking (MLM) and next sentence prediction
- Accounts for the bidirectional aspect → instead of reading left-to-right only, it takes the surrounding words in consideration to construct a context.
- Text modeled as “tokens” which are then represented by vectors. We’ll get more into that in the following slides.

Fine-tuning!

- Trained on very very specific tasks:
  - MLI
  - NLI (natural language inference)
  - QA (question-answering)
- Only add a classification layer to the pretrained model
- “We hypothesize that when the model is fine-tuned directly on the downstream tasks and uses only a small number of randomly initialized parameters, the task-specific models can benefit from larger, more expressive pre-trained representations…”

Pre-Training

Task #1 - Masking: Predicting the Masked Tokens

Masked LM and the Masking Procedure Assuming the unlabeled sentence is `my dog is hairy`, and during the random masking procedure we chose the 4-th token (which corresponding to `hairy`), our masking procedure can be further illustrated by

- 80% of the time: Replace the word with the `[MASK]` token, e.g., `my dog is hairy` → `my dog is [MASK]`
- 10% of the time: Replace the word with a random word, e.g., `my dog is hairy` → `my dog is apple`
- 10% of the time: Keep the word unchanged, e.g., `my dog is hairy` → `my dog is hairy`. The purpose of this is to bias the representation towards the actual observed word.

Task #2 - Next Sentence Prediction

Next Sentence Prediction The next sentence prediction task can be illustrated in the following examples.

Input = `[CLS] the man went to [MASK] store [SEP]`

`he bought a gallon [MASK] milk [SEP]`

Label = `IsNext`

Input = `[CLS] the man [MASK] to the store [SEP]`

`penguin [MASK] are flight ##less birds [SEP]`

Label = `NotNext`

Fine-Tuning

- Utilizes self-attention mechanism to determine the importance of different parts of the input
  - Also helps emphasize this bi-directionality by encoding the concatenated text pair with this self-attention
- Simply plug in task-specific input/output pairs to make minor adjustments
  - Similar to the Next Sentence Prediction in the pre-training stage, but modified so that the model can adjust based on the task

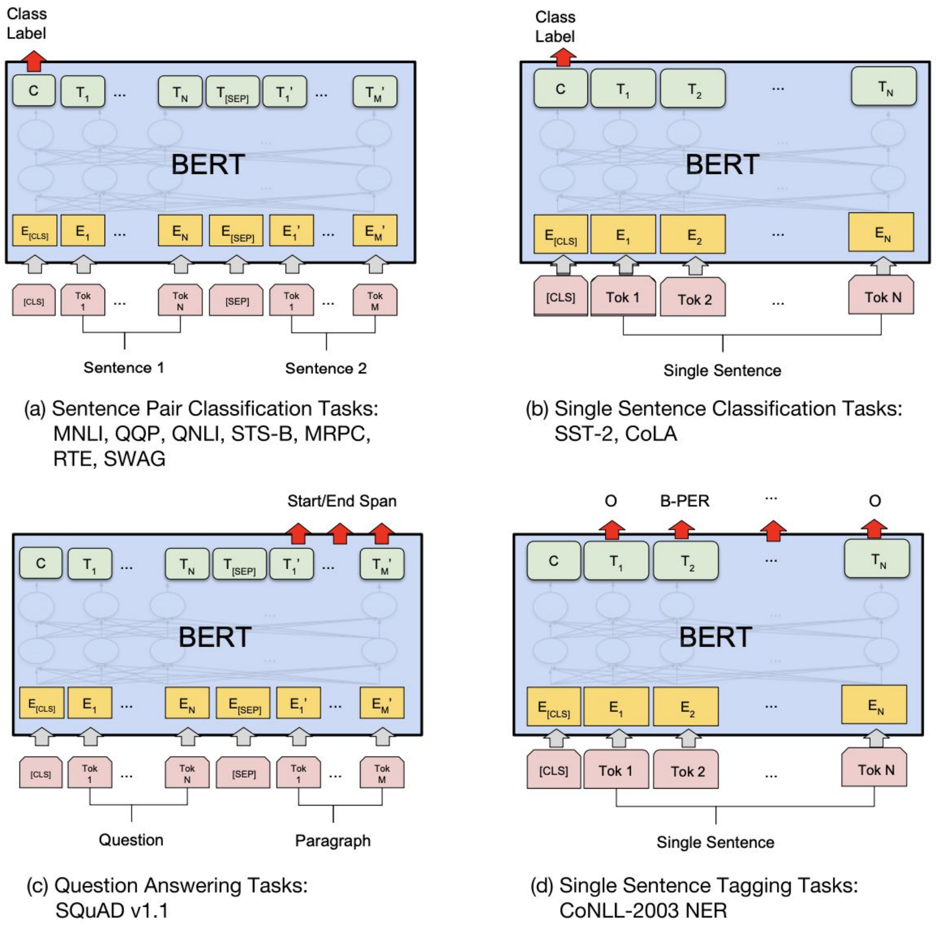


Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

Stats:

Batches = 256 sequences

Tokens <= 512

Hence: 256 sequences * 512 tokens = 128,000 tokens/batch

Across 1M steps, about 40 epochs over 3.3B wods

In fine-tuning:

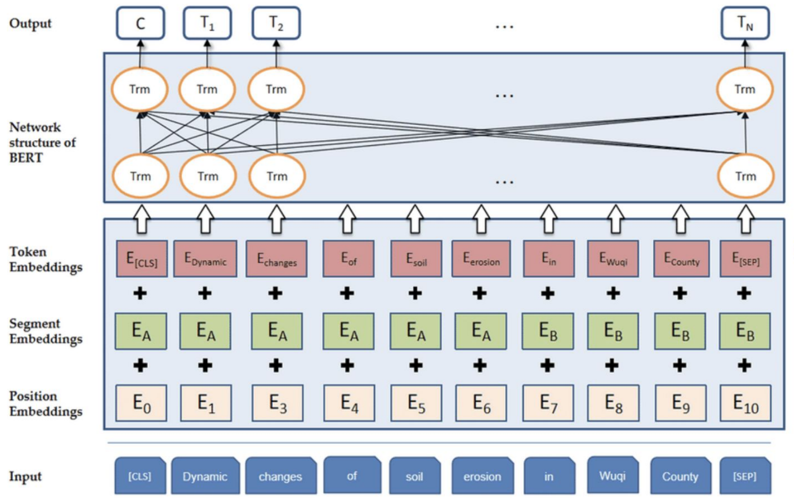
Batch size - 16, 32

Learning rate: 5e-5, 3e-5, 2e-5

Number of epochs: 2, 3, 4

BERT-Base: 110M parameters

BERT-Large: 340M parameters



Sources

- https://www.youtube.com/watch?v=OR0wfP2FD3c
- BERT: Pre-training of Deep Bidirectional Transformers for ...arXivhttps://arxiv.org › cs
- BERT 101 - State Of The Art NLP Model ExplainedHugging Facehttps://huggingface.co › blog › bert-101
- https://h2o.ai/wiki/transformer-architecture/