BERT, which stands for Bidirectional Encoder Representations from Transformers, marks a significant advancement in natural language processing (NLP). Its core innovation lies in its ability to pre-train deep bidirectional representations from unlabeled text. This means BERT learns by considering both the left and right context of a word simultaneously across all its layers, leading to a much richer understanding of language than previous models.

How BERT Works: Pre-training and Fine-tuning

BERT's success comes from its two-phase approach:

1. **Pre-training:** In this initial stage, the model is trained on vast amounts of unlabeled text data. This involves two distinct pre-training tasks:
    - **Masked LM (Language Model):** Intuitively, a deep bidirectional model is far more powerful than a simple left-to-right model or a shallow combination of left-to-right and right-to-left models. To achieve this, BERT randomly masks 15% of the input tokens. It then predicts these masked tokens, forcing it to learn context from both sides. To address the mismatch of the `[MASK]` token not appearing during fine-tuning, the training data generator chooses 15% of token positions for prediction. If a token is chosen, it's replaced with `[MASK]` 80% of the time, a random token 10% of the time, and remains unchanged 10% of the time. The final hidden vectors corresponding to the mask tokens are fed into an output softmax over the vocabulary, just like a standard language model. Unlike denoising auto-encoders, BERT only predicts the masked words, not the entire input.
    - **Next Sentence Prediction (NSP):** Many crucial downstream tasks like Question Answering (QA) and Natural Language Inference (NLI) rely on understanding the relationship between two sentences, which isn't directly captured by language modeling alone. For NSP, when choosing sentences A and B for each pre-training example, 50% of the time B is the actual next sentence that follows A (labeled as `IsNext`), and 50% of the time it is a random sentence from the corpus (labeled as `NotNext`).

2. **Fine-tuning:** Once pre-trained, the BERT model can be adapted to various NLP tasks with minimal effort. This involves initializing the BERT model with its pre-trained parameters and adding just one additional output layer. All parameters are then fine-tuned using labeled data from the specific downstream task. It's important to note that each downstream task typically gets its own separate fine-tuned model, even if they all start with the same pre-trained parameters.

BERT's Architecture and Key Features

A distinctive feature of BERT is its unified architecture across different tasks. Its model architecture is a multi-layer bidirectional Transformer encoder, based on the original implementation described in Vaswani et al. (2017) and released in the `tensor2tensor` library.

The paper reports results on two primary model sizes:

- BERTBASE: (L=12,H=768,A=12), with a total of 110 million parameters. This size was chosen to have the same model size as OpenAI GPT for comparison purposes.
- BERTLARGE: (L=24,H=1024,A=16), with a total of 340 million parameters.

A critical difference between BERT and OpenAI GPT is that the BERT Transformer uses bidirectional self-attention, allowing every token to attend to context to both its left and right. In contrast, the GPT Transformer uses constrained self-attention, where every token can only attend to context to its left.

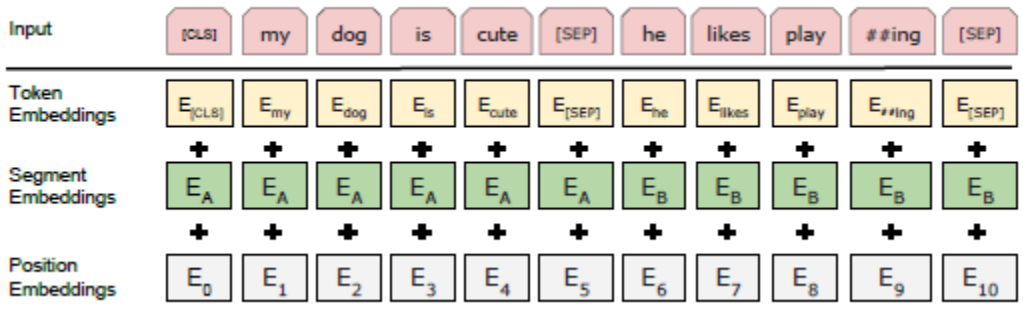
Input Representation

To enable BERT to handle a wide variety of downstream tasks, its input representation is designed to unambiguously represent both a single sentence and a pair of sentences (e.g., Question, Answer) within one token sequence. Throughout the work, a "sentence" can be an arbitrary span of contiguous text, not necessarily a linguistic sentence. A "sequence" refers to the input token sequence to BERT, which may be a single sentence or two sentences packed together.[3]

Key components of the input representation include:

- WordPiece embeddings: BERT uses WordPiece embeddings (Wu et al., 2016) with a 30,000-token vocabulary.
- Special tokens:
    - The first token of every sequence is always a special classification token ([CLS]). The final hidden state corresponding to this token[4] (RH_{C_2}) is used as the aggregate sequence representation for classification tasks.
    - Sentence pairs are packed together into a single sequence[5] and differentiated with a special separator token ([SEP]).
- Learned embeddings: A learned embedding is added to every token, indicating whether it belongs to sentence A or sentence B.

For any given token, its input representation (E) is constructed by summing the corresponding token, segment, and position embeddings. This construction is visualized in Figure 2 of the original paper. The final hidden vector for the i-th input token is denoted as RHTi.



State-of-the-Art Results

BERT achieved new state-of-the-art results on eleven natural language processing tasks, showcasing its remarkable ability to generalize across different challenges:

- GLUE score: Pushed to 80.5% (a 7.7% absolute improvement).
- MultiNLI accuracy: Improved to 86.7% (a 4.6% absolute improvement).
- SQuAD v1.1 question answering Test F1: Reached 93.2 (a 1.5 point absolute improvement).
- SQuAD v2.0 Test F1: Achieved 83.1 (a 5.1 point absolute improvement).

These impressive gains underscore BERT's transformative impact on NLP, demonstrating how pre-training deep bidirectional representations enables models to achieve superior performance with minimal task-specific modifications. BERT effectively democratized access to state-of-the-art NLP models by providing a powerful, adaptable pre-trained backbone that significantly reduces the need for extensive task-specific feature engineering.