# BERT

What’s BERT?

- Full name
  - Bidirectional Encoder Representation from Transformers
- What does it do?
  - Pre-train deep bidirectional representation from unlabeled text
  - By jointly conditioning on both left and right context in all layers.
  - BERT model can be fine-tuned with just one additional output layer
- What is it good at?
  - State-of-the-art models for a wide-range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.
- How did it do it?
  - Use a masked language model, where it randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked word based only on its context.
  - It utilizes both the left and the right context, which in turn allows us to pre-train a deep bidirectional transformer.
  - “Next sentence prediction” task the jointly pre-train text-pair representation.

In the paper

- Demonstrate the importance of bidirectional pre-training for language representation.
- Shows that pre-trained representations reduce the need for many heavily-engineered task-specific architectures.
- Advances the state of the art for eleven NLP tasks.

Background

- Unsupervised feature-based approaches
  - these approaches have been generalized to coarser granularities such as sentence embeddings or paragraph embeddings.
- Unsupervised fine-tuning approaches
  - sentence or document encoders which produce contextual token representations have been pref-trained from unlabled text and fine-tuned for a supervised downstream task.

- Transfer learning from supervised data
- BERT
    - There are two steps in BERT framework: pre-training and fine-turning.
    - Pre-training
        - Model is trained on unlabeled data over different pre-training tasks
    - Fine-tuning
        - First initialized with the pre-trained parameters
        - Then fine-tuned all parameters using labeled data from the downstream tasks.
    - Features
        - Unified architecture across different tasks. There is minimal difference between the pre-trained architecture and the final downstream architecture.
    - Model architecture
        - Multi-layer bidirectional transformer encoder
        - Implementation is identical in Vaswani (2017)
- Input/Output representation
    - Input representation is able to unambiguously represent both a single sentence and a pair of sentences in one token sequence.
    - Use WordPiece embeddings with a 30,000 token vocabulary.
- Pre-training BERT
    - Use two unsupervised tasks
    - Task #1: Masked LM
        - It's also referred to as Cloze task.
        - Mask 15% of all WordPiece tokens in each sequence at random
        - The downside is that the mismatch between pre-training and fine-tuning. To mitigate this, masked words are not always replaced with the actual token
    - Task #2: Next Sentence Prediction
        - For NLP tasks such as question answering, or language inference, it's important to understand the relationship between two sentences. But it's not directly captured by language modeling.
        - To fix this, they pre-train for a binarized "next sentence prediction" task that can be trivially generated from any monolingual corpus.
- Pre-training data

    - BooksCorpus (800M words)
    - English Wikipedia (2,500M words)
        - Only extract text passages and ignore lists, tables and headers
    - It is critical to use a document –level corpus rather than a shuffled sentence-level corpus in order to extract long contiguous sequences.

Fine-tuning BERT

- It is straightforward because of the self-attention mechanism in the Transformer
- BERT can model many downstream tasks by swapping out the appropriate inputs and outputs.
- Plug in the task-specific inputs and outputs into BERT and fine-tune all the parameters end-to-end.
- Compared to pre-training, fine-tuning is relatively inexpensive.