BERT

BERT (Bidirectional Encoder Representations from Transformers)

BERT is a transformer-based deep learning model introduced by Google in 2018. It is designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers. This allows BERT to capture rich contextual information and significantly improve performance on NLP tasks.

BERT Architecture in Detail

BERT is based on the Transformer encoder architecture introduced in the paper "Attention Is All You Need" (Vaswani et al., 2017). It consists of multiple stacked encoder layers and does not use the decoder part of the Transformer.

1. Input Representation

BERT's input representation is designed to handle both single-sentence and sentence-pair tasks.
It consists of:

- Token Embeddings: Each token is converted into an embedding using WordPiece tokenization.
- Segment Embeddings: Used to distinguish between sentences in sentence-pair tasks.
- Position Embeddings: Capture the order of tokens in a sequence.

The final input embedding is the sum of these three embeddings.

2. Transformer Encoder Layers

BERT consists of multiple Transformer encoder layers.
The main components of each encoder layer are:

a. Multi-Head Self-Attention Mechanism

BERT uses self-attention to compute contextualized representations of words. Multi-head attention allows the model to focus on different parts of the input simultaneously.

It calculates attention scores using the following formula: $\text{Attention}(Q, K, V) = \text{softmax}(QK^T / \sqrt{dk})\ V$

Where:
Q, K, V → Query, Key, and Value matrices (from input embeddings)
dk → Dimension of key vectors

b. Feedforward Network

Each encoder layer also contains a position-wise feedforward network (FFN), which consists of:

- A fully connected layer with ReLU activation.
- A second fully connected layer.

c. Layer Normalization and Residual Connections

Each sub-layer in the encoder (self-attention and feedforward network) is followed by:

- Layer normalization
- Residual connection (adds input to the output)

3. Output Representation

The final output from BERT's last encoder layer can be used for various NLP tasks:

- The [CLS] token representation is used for classification tasks.
- Each token's output embedding can be used for token-level tasks like Named Entity Recognition (NER).

BERT Model Comparison

Attribute	BERT-Base	BERT-Large
Layers	12	24
Hidden Size	768	1024
Attention Heads	12	16
Parameters	110M	340M

Pretraining and Fine-tuning

1. Pretraining

BERT is pretrained on two tasks:

- Masked Language Model (MLM): Randomly masks 15% of tokens and trains the model to predict them.
- Next Sentence Prediction (NSP): Determines whether one sentence follows another.

2. Fine-tuning

After pretraining, BERT can be fine-tuned on specific NLP tasks with additional task-specific layers.
BERT revolutionized NLP by introducing bidirectional context learning, leading to state-of-the-art results in various benchmarks.

How RoBERTa Was Trained: A Deep Dive

1. Data Collection & Preprocessing

RoBERTa was trained on a much larger dataset than BERT, improving its robustness and generalization.

Data Sources

- BOOKCORPUS: A dataset containing 11,038 freely available books.
- English Wikipedia: Processed and cleaned articles.
- Common Crawl News (CC-News): A 76GB dataset of news articles.
- OpenWebText: A dataset replicating the high-quality texts from Reddit.
- Stories Dataset: A collection of narrative texts from books and web sources.

Preprocessing Steps

- Tokenization: Applied Byte-Pair Encoding (BPE) to split text into subword units.
- Text Normalization: Converted to lowercase, removed special characters, and standardized punctuation.
- Data Filtering: Removed duplicate, low-quality, or noisy content.
- Sentence Splitting: Parsed text into meaningful sentence structures.

2. Model Architecture

RoBERTa shares the same architecture as BERT-large but introduces several optimizations.

Model Specifications

- 24 Transformer layers (same as BERT-large).
- 1024 hidden units per layer.
- 16 attention heads per layer.
- Total Parameters: 355 million.

Key Architectural Improvements

- Dynamic Masking: Unlike BERT, RoBERTa applies token masking dynamically at each training step.
- Removed Next Sentence Prediction (NSP): Found to be unnecessary and was discarded.
- Trained on Larger Batches: Used batch sizes of up to 8,000 samples for stability.
- Trained for More Steps: Pretrained with 500K additional training steps compared to BERT.

3. Pretraining Objective: Masked Language Modeling (MLM)

RoBERTa was trained using a single objective: Masked Language Modeling (MLM).

Training Process

- 15% of tokens were randomly masked.
- 80% of the masked tokens were replaced with "[MASK]".
- 10% of the masked tokens were replaced with a random word.
- 10% of the masked tokens remained unchanged.
- Optimized using cross-entropy loss to minimize prediction error.

4. Hardware & Training Setup

RoBERTa required massive computational resources to train effectively.

Training Configuration

- Trained on 1024 NVIDIA V100 GPUs.
- Used 16 GPUs per node with a batch size of 8,000.
- Applied Adam optimizer with weight decay.
- Utilized mixed-precision training (FP16) for efficiency.

5. Fine-Tuning on Downstream Tasks

After pretraining, RoBERTa was fine-tuned for various NLP benchmarks.

Benchmarks Used

- GLUE: General Language Understanding Evaluation.
- SQuAD: Question Answering dataset.
- RACE: Reading comprehension dataset.
- SuperGLUE: More advanced NLP tasks.

Fine-Tuning Process

- Added a classification head on top of the transformer.
- Trained with task-specific loss functions.
- Used transfer learning to adapt to new tasks.

6. Performance Improvements Over BERT

RoBERTa outperformed BERT on multiple NLP tasks.

Key Reasons for Improved Performance

- Better token masking strategy (dynamic masking).
- Larger training dataset (160GB vs. BERT's 16GB).
- Longer training with larger batch sizes.
- Removal of Next Sentence Prediction (NSP).

7. Key Takeaways

- RoBERTa is an optimized version of BERT with improved pretraining.
- It removed Next Sentence Prediction (NSP) for simpler training.
- It used dynamic token masking to improve generalization.
- It was trained on 10x more data than BERT for better language understanding.
- RoBERTa outperformed BERT on nearly all NLP benchmarks.

ELECTRA:
Efficiently Learning an Encoder that Classifies Token Replacements Accurately

ELECTRA is a transformer-based model introduced by Google Research in 2020. It introduces a more efficient pretraining strategy than BERT, achieving similar or better performance with fewer training resources.

1. Key Idea: Replacing MLM with a "Replacement Detection" Task

Unlike BERT, which masks some tokens and predicts them, ELECTRA trains the model to detect replaced tokens rather than predict masked ones.

Core Difference Between BERT and ELECTRA

BERT: Uses Masked Language Modeling (MLM)—randomly masks 15% of tokens and learns to predict them.

ELECTRA: Uses Replaced Token Detection (RTD)—replaces words with incorrect ones (generated by a small "Generator" model) and trains a "Discriminator" model to detect which words are fake.

2. ELECTRA Architecture

ELECTRA consists of two Transformer models:

a) Generator (G)

- Similar to BERT (smaller in size).
- Uses Masked Language Modeling (MLM) to generate fake words for replacement.

b) Discriminator (D)

- Learns to detect fake tokens rather than predicting masked ones.
- Instead of classifying words into a vocabulary, it outputs a binary classification (Real or Fake) for each token.

Example:

- Original: "The cat sat on the mat."
- Generator replaces "sat" → "ate": "The cat ate on the mat."
- Discriminator must classify "ate" as "Fake" and all other words as "Real".

3. Why is ELECTRA More Efficient?

- ✅ Full sentence learning: Unlike BERT, where only 15% of tokens contribute to learning, ELECTRA trains on every token, making learning more efficient.
- ✅ Smaller models, same accuracy: ELECTRA-Small (14M parameters) matches BERT-Base (110M parameters) in accuracy.
- ✅ Faster pretraining: Requires less compute power than BERT to achieve similar or better results.

4. ELECTRA Variants

Model Performance Comparison

Performance	ELECTRA-Small	ELECTRA-Base	ELECTRA-Large
Matches BERT-Base	14M	110M	N/A
Matches BERT-Large	N/A	N/A	335M
Outperforms BERT-Large	N/A	N/A	Yes

BERT vs. ELECTRA Comparison

Feature	BERT	ELECTRA
Pretraining Task	Masked Language Modeling (MLM)	Replaced Token Detection (RTD)
Efficiency	Uses only 15% of tokens	Uses all tokens for training
Compute Power	Requires more training steps	Learns faster, fewer steps needed
Performance	Strong, but expensive	More efficient, better in low-resource settings

5. Why Use ELECTRA?

- ✅ Faster and cheaper to train than BERT.
- ✅ Better accuracy for NER, sentiment analysis, and QA.
- ✅ Smaller models perform as well as larger BERT variants.