Attention is All You Need

BERT and the Self-attention mechanism

At the core of BERT (Bidirectional Encoder Representations from Transformers) lies the self-attention mechanism, which enables the model to capture contextual relationships between all tokens in a sequence simultaneously, regardless of their distance. input sequence of token embeddings $X \in R\, n \times d$ $X \in R^{n \times d}$, where n is the sequence length and d is the embedding dimension, the self-attention mechanism computes three matrices via learned linear projections:

$$Q=XW_Q \ ,K=XW_K \ ,V=XW_V$$

Where $W_Q, W_K, W_V \in R^{dxdk}$ are learned parameter matrices. The attention output is computed as:

$$\text{Attention}(Q,K,V)=\text{softmax}\left(\frac{QK^T}{\sqrt{d^k}}\right)V$$

The BERT uses multi-head attention, allowing each token to attend to every other in the sequence, weighted by relevance, to capture diverse linguistic features by running multiple self-attention operations parallel with different learned projections. The outputs from all heads are concatenated and passed through a final linear layer:

$$\text{MultiHead}(Q,K,V)=\text{Concat}(\text{head}_1,\ldots,\text{head}_h)W_O$$

BERT's bidirectional self-attention allows tokens to simultaneously attend to all positions on both left and right, making it effective for tasks requiring deep contextual understanding, such as question answering and sentence classification.

Positional encoding

BERT, a bidirectional encoder representation from transformers, utilizes the Transformer architecture, which lacks inherent recurrence or convolution. Vaswani et al. introduced fixed sinusoidal positional encodings, which BERT authors adopted instead of learned positional embeddings, to encode the sequential order of tokens. For an input sequence of tokens

$X=[x_1,x_2,\ldots,x_n]$, BERT adds a corresponding learned positional embedding $P = [\, p_1, p_2 , \ldots , p_n]$ resulting in input representations:

$$z_i = x_i + p_i \text{ ,for } i=1,2,\ldots,n$$

BERT uses positional embeddings in its input sequence to distinguish between different sentences in tasks involving sentence pairs. These absolute embeddings are trained jointly with

the rest of the model, ensuring each position in the input sequence has a distinct embedding vector. However, absolute positional embeddings limit the model's ability to generalize across variable-length contexts or capture relative positions. This has led to the exploration of relative positional encoding in later models like RoBERTa and Transformer-XL.

Benefits of transformer architecture

BERT, originally introduced by Vaswani et al., leverages the transformer architecture to model deep bidirectional context in language, offering numerous benefits.

Bidirectional Contextualization: The transformer, unlike unidirectional models, employs the encoder component to simultaneously attend to both left and right contexts, enhancing understanding of entire sentence structure in tasks like question answering and natural language inference.

Self-Attention Mechanism: Traditional RNNs often overlook semantic relationships due to their lack of a self-attention mechanism, but the transformer model addresses this issue by dynamically weighting tokens in the input sequence, enhancing syntactic representation.

Scalability and Parallelism: Transformers significantly improve training efficiency by processing input sequences in parallel, unlike RNNs which operate sequentially. This architecture enables efficient training on large datasets using modern hardware accelerators.

Scaling laws

Devlin and Hoffmann's research on masked language models like BERT, which use a masked language modeling objective, has shown that increasing parameters, training data, and compute resources improves performance across downstream tasks, particularly on benchmarks like GLUE and SQuAD. Their work, including RoBERTa and ELECTRA, confirmed that scaling BERT-like models leads to substantial improvements, provided the pre-training task remains expressive that loss decreases approximately as a power-law with respect to model size N dataset size D and compute C:

$$L(N,D,C) \propto N^{-\propto}$$

The MLMs like BERT may require more careful calibration of training duration and batch size to fully utilize scaling benefits, especially due to the denoising nature of the pretext task.