Deep Learning-Transformer Architecture

Here's a summary of the Transformer architecture, along with an example to illustrate its key components:

Overview

The Transformer is a neural network architecture that was introduced in 2017 by Vaswani et al. (Attention is All You Need) as a way to process sequential data without relying on recurrence or convolution.

Components

1. **Encoder-Decoder Architecture**: The Transformer consists of an encoder and a decoder, similar to traditional sequence-to-sequence models.
2. **Self-Attention Mechanism**: The core component of the Transformer, self-attention allows the model to attend to all positions in the input sequence simultaneously and weigh their importance.
3. **Positional Encoding**: To preserve information about the position of each token in the input sequence, positional encoding is added to the input embeddings.
4. **Feed Forward Network (FFN)**: A fully connected network that processes the output of self-attention and adds non-linearity.

Example: English-Spanish Translation

Suppose we want to translate an English sentence "Hello, how are you?" into Spanish. We have a dataset of pairs of English sentences and their corresponding Spanish translations.

Step 1: Tokenization

We tokenize the input English sentence into individual words: ["Hello", ",", "how", "are", "you", "?"]

Step 2: Embedding

We embed each token in the vocabulary (including padding tokens) to create a fixed-size vector representation of the sequence. This is done using an embedding layer.

`English Sentence`: [0.5, -0.3, 0.2] (Embedding for "Hello"), [0.1, 0.7], [-0.4, -0.9] (Embeddings for other tokens)

Step 3: Positional Encoding

To preserve information about the position of each token in the sequence, we add a positional encoding vector to the input embeddings.

`English Sentence`: [0.5, -0.3, 0.2], [0.1, 0.7], [-0.4, -0.9], [0.3, -0.8] (Positional Encoding for each token)

Step 4: Self-Attention

The self-attention mechanism allows the model to attend to all positions in the input sequence simultaneously and weigh their importance.

`Self-Attention Weights`: Weight matrix of size (`num_heads`, `sequence_length`, `sequence_length`) that contains attention weights for each head, token pair

`Attended Outputs`: Output embeddings after applying self-attention, where attention is computed across all tokens for each position

Step 5: Feed Forward Network (FFN)

The FFN processes the output of self-attention and adds non-linearity.

`FFN Output`: Final output embedding after passing through FFN, which contains information from both input embeddings and positional encoding

This process is repeated in the decoder to generate a sequence of Spanish tokens that correspond to the English sentence "Hello, how are you?".

Key Takeaways

1. The Transformer uses self-attention to weigh importance across all positions in the input sequence.
2. Positional encoding helps preserve information about token position.
3. Feed Forward Network (FFN) adds non-linearity after self-attention output.
4. Encoder and decoder share similar architecture, but focus on different aspects of the task.

The Transformer has been widely adopted for many NLP tasks, including machine translation, question answering, text classification, and more.

Curated by Brajesh Kumar