Natural Language Processing
NLP_CLT_1c_June_8th_2025

Eng. Maytham Ghanoum

Artificial Intelligence & Deep Learning Specialist
MTN Syria – SCS – SVU CLT
+963947222064 - +963982018359
https://www.linkedin.com/in/maytham-ghanoum-697aa5207/
https://www.facebook.com/maytham.ghanoum

Attention Mechanism

Attention Mechanism:
The attention mechanism is a technique that allows the model to focus on different parts of the input sequence when generating each part of the output sequence. It addresses the limitation of traditional RNNs and LSTMs, which struggle with long-range dependencies.

Attention Mechanism

Why Attention is Important Improves Context Understanding:

In long sentences, important information might be far from the current word being processed. Attention helps the model weigh the importance of different words differently.

Parallelism: Unlike RNNs, attention mechanisms allow more parallelization, which speeds up training.

Attention Mechanism

How Attention Works

Score Calculation: For a given input, the model calculates a score for each word in the input sequence. This score represents the relevance of the input word to the current output word being generated.

Softmax: These scores are then normalized using a softmax function to get attention weights.

Context Vector: The context vector is computed as the weighted sum of the input words' embeddings, using the attention weights.

Combining Context with Output: The context vector is then used to generate the output word.

Attention Mechanism

Mathematically, the process can be described as:

$$Score(h_t, h_s) = h_t^T W_a h_s$$

$$a_{ts} = \frac{\exp(Score(h_t, h_s))}{\sum_s^t \exp(Score(h_t, h_s))}$$

$$context_t = \sum_s a_{ts} h_s$$

Attention Mechanism

Mathematically, the process can be described as:

h_t is the hidden state of the decoder at time t.

h_s is the hidden state of the encoder at time s.

W_a is a learnable weight matrix.

α_{ts} are the attention weights.

Attention Mechanism

Types of Attention

Global Attention: The model looks at all the hidden states of the input sequence.

Local Attention: The model focuses on a subset of the hidden states, typically around a specific point.

Attention Mechanism

Steps in Attention Mechanism

Alignment Scores: For each output word, calculate scores between it and each input word.

Attention Weights: Apply a softmax to these scores to get weights.

Context Vector: Compute a weighted sum of the input hidden states using these weights.

Output: Combine this context vector with the current output word's hidden state to generate the final output.

Attention Mechanism

Types of Attention

Global Attention: The model looks at all the hidden states of the input sequence.

Local Attention: The model focuses on a subset of the hidden states, typically around a specific point.

Transformer

Transformers are a type of neural network architecture designed for processing sequences of data, such as text. They were introduced in the paper "Attention is All You Need" by Vaswani et al. in 2017. The key innovation of Transformers is the self-attention mechanism, which allows the model to weigh the importance of different words in a sequence when encoding and decoding.

Transformer

Why Work with Transformers?

Efficiency and Parallelization:

Unlike RNNs and LSTMs, which process data sequentially, Transformers can process entire sequences in parallel. This parallelization significantly speeds up training and inference.

Long-range Dependencies:

Transformers can capture long-range dependencies in sequences better than RNNs and LSTMs because self-attention allows every word in a sequence to directly attend to every other word.

Scalability:

Transformers can be scaled to handle very large datasets and models with billions of parameters, making them suitable for state-of-the-art NLP tasks.

Performance:

They achieve superior performance on a wide range of NLP tasks, including machine translation, text summarization, and language modeling.

Transformer

How Were Transformers Created?

Transformers were developed to address the limitations of traditional sequence models like RNNs and LSTMs. The main issues with these models were:

Sequential Processing:

RNNs and LSTMs process tokens one by one, which prevents parallelization and makes them slow for long sequences.

Difficulty in Capturing Long-range Dependencies:

RNNs struggle with long-range dependencies due to issues like vanishing gradients, making it hard to capture relationships between distant words in a sequence.

To overcome these issues, the authors of "Attention is All You Need" proposed the Transformer architecture, which relies heavily on self-attention mechanisms.

Transformer

Key Components of the Transformer Architecture

Self-Attention Mechanism:

Self-attention allows the model to weigh the importance of different words in a sequence relative to each other. It calculates attention scores between all pairs of words in a sequence, enabling the model to capture dependencies regardless of distance.

Multi-Head Attention:

Instead of a single attention mechanism, the Transformer uses multiple attention heads. Each head learns different aspects of the relationships between words, providing richer representations.

Positional Encoding:

Since Transformers do not inherently capture the order of words, positional encodings are added to the input embeddings to provide information about the position of each word in the sequence.

Encoder-Decoder Architecture:

The Transformer consists of an encoder and a decoder, each composed of multiple layers. The encoder processes the input sequence and generates a representation, which the decoder then uses to produce the output sequence.

Transformer

Structure of a Transformer

Encoder:

- **Input Embedding**: Converts input tokens into dense vectors.
- **Positional Encoding**: Adds positional information to embeddings.
- **Stack of Encoder Layers**: Each layer consists of:
  - **Multi-Head Self-Attention**: Applies self-attention to the input.
  - **Feed-Forward Network (FFN)**: Applies a two-layer fully connected network to each position.
  - **Residual Connections and Layer Normalization**: Helps in training deep networks.

Transformer

Structure of a Transformer

Decoder:

Output Embedding: Converts output tokens into dense vectors.

Positional Encoding: Adds positional information to embeddings.

Stack of Decoder Layers: Each layer consists of:

- **Masked Multi-Head Self-Attention**: Applies self-attention with a mask to prevent attending to future positions.
- **Multi-Head Attention with Encoder Outputs**: Attends to the encoder's output.
- **Feed-Forward Network (FFN)**: Similar to the encoder.
- **Residual Connections and Layer Normalization**: Helps in training deep networks.

Transformer

Applications:

Machine Translation: Translating text between languages.

Text Summarization: Generating concise summaries of longer texts.

Language Modeling: Predicting the next word in a sequence.

Question Answering: Providing answers to questions based on context.

Text Generation: Generating coherent and contextually relevant text.

Transformer

Transformers use several key mathematical operations:

1. **Scaled Dot-Product Attention**:

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

- Q is the matrix of queries.
- K is the matrix of keys.
- V is the matrix of values.
- d_k is the dimension of the keys (or queries).

Transformer

Transformers use several key mathematical operations:

2. **Multi-Head Attention**:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)W^O$$

- Each head is computed as:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

- W_i^Q, W_i^K, W_i^V are projection matrices for the i-th head.

3. **Position-Wise Feed-Forward Networks**:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

Transformer

Hugging Face

Hugging Face is a company that provides a wide range of tools and libraries for Natural Language Processing (NLP). Their Transformers library is one of the most popular libraries for working with pre-trained transformer models.

Transformers Library:

Purpose: Simplifies the use of transformer models for various NLP tasks.

Models: Includes popular models like BERT, GPT-2, T5, RoBERTa, and many more.

Tasks: Supports tasks such as text classification, question answering, text generation, and translation.

Transformer

Hugging Face

Hugging Face is a company that provides a wide range of tools and libraries for Natural Language Processing (NLP). Their Transformers library is one of the most popular libraries for working with pre-trained transformer models.

Transformers Library:

Purpose: Simplifies the use of transformer models for various NLP tasks.

Models: Includes popular models like BERT, GPT-2, T5, RoBERTa, and many more.

Tasks: Supports tasks such as text classification, question answering, text generation, and translation.