Report on *Attention Is All You Need*

Submitted in Partial Fulfillment of the Requirements for Seminars During the Hilary Term

Ion Lipsiuc

April 5, 2025

Abstract

This report provides an in-depth analysis of the influential paper *Attention Is All You Need*,[1] which introduces the transformer model - a groundbreaking architecture that relies entirely on attention mechanisms, thereby eliminating the need for recurrence. We detail the model's core components, including self-attention, multi-head attention, and positional encodings, and review experimental results that demonstrate its significant performance improvements on machine translation tasks.

Contents

[1] See (Vaswani, 2017).

1 Introduction

Recent advances in natural language processing (NLP) have been driven by the need to improve the way models handle sequential data, especially in tasks such as machine translation. Traditionally, recurrent neural networks (RNNs) and their variants, although effective, have encountered limitations in capturing long-range dependencies due to their sequential processing nature. These limitations have motivated researchers to explore more parallelizable and efficient approaches.

In their seminal work, *Attention Is All You Need*, the authors introduce the transformer - a novel architecture that completely abandons recurrence in favour of a self-attention mechanism. Unlike RNN-based models, the transformer is capable of attending to all parts of an input sequence simultaneously. This self-attention mechanism assigns dynamic weights to each token in the sequence, allowing the model to capture global dependencies regardless of the distance between words.

A key innovation of the transformer is its use of multi-head attention. By employing multiple attention heads, the model can focus on different parts of the sequence concurrently, which enhances its ability to learn complex relationships within the data. Additionally, the incorporation of positional encodings ensures that the model retains information about the order of the sequence, compensating for the absence of recurrence.

The architectural design of the transformer not only addresses the inherent limitations of RNNs but also significantly improves computational efficiency. With its high degree of parallelisation, the transformer reduces training time while achieving state-of-the-art performance on machine translation benchmarks. This breakthrough has not only advanced the field of machine translation but has also paved the way for numerous subsequent developments in NLP.

In this report, we provide an in-depth analysis of the transformer architecture as presented in *Attention Is All You Need*. We will examine its core components - including self-attention, multi-head attention, and positional encodings - and discuss how these innovations contribute to its superior performance. Through a detailed exploration of the experimental results and the model's overall impact, we aim to highlight the transformative influence of this work on modern NLP research.

2 Background and Related Work

2.1 Neural Sequence Modelling

Early approaches to sequence modeling in NLP relied heavily on RNNs and their variants, such as long short-term memory (LSTM) networks and gated recurrent units (GRUs). These models leverage sequential processing and maintain hidden states to capture contextual information. Despite their success, the inherent sequential nature of these architectures limits parallelisation and makes it challenging to learn long-range dependencies effectively, particularly in tasks that involve lengthy input sequences.

2.2 Attention Mechanisms

To overcome the limitations of traditional recurrent architectures, attention mechanisms were introduced as an innovative approach to dynamically focus on different parts of the input sequence. In neural machine translation, for example, the attention mechanism - first popularised by Bahdanau (2014) - enabled models to weight the importance of different input tokens when generating each output token. This dynamic reallocation of focus not only improved translation quality but also enhanced the interpretability of the models by providing insight into which parts of the input contributed most significantly to the output.

2.3 Advancements Leading to the Transformer

Subsequent developments explored alternative architectures that could better capture global dependencies while allowing for greater computational efficiency. Convolutional sequence models, such as ByteNet and ConvS2S, leveraged parallel computations by using convolutional layers to capture local context. However, while these models improved parallelisability, they still faced challenges in effectively modelling long-range relationships.

The transformer model represents a paradigm shift by dispensing with recurrence and convolutions entirely in favor of self-attention mechanisms. In this architecture, every token in the input sequence can directly interact with every other token, allowing for the efficient capture of both local and global dependencies. The model further employs multi-head attention to learn diverse representations of the data simultaneously, while positional encodings are used to preserve the order of the sequence. This combination of innovations has set a new standard for performance and efficiency in sequence modelling, influencing a broad range of applications in modern NLP.

Together, these advancements trace the evolution from traditional sequential models to the highly parallel and effective transformer architecture, which has now become a cornerstone of contemporary NLP research.

3 Transformer Architecture

3.1 Overview

In this section, we break down the components of the transformer model. We cover the overall architecture and key components such as self-attention, multi-head attention, positional encodings, and the feed-forward networks.

The transformer model is built on the idea of self-attention, which allows every position in the input sequence to interact with every other position. This design eliminates the need for sequential processing and enables parallel computation. The model is divided into two main parts, the encoder and the decoder, each consisting of multiple identical layers.

3.2 Encoder-Decoder Structure

The encoder transforms an input sequence $\mathbf{X} = (x_1, x_2, \ldots, x_n)$ into a set of continuous representations $\mathbf{H} = (h_1, h_2, \ldots, h_n)$. The decoder then uses $\mathbf{H}$ to generate an output sequence $\mathbf{Y} = (y_1, y_2, \ldots, y_m)$. Each encoder layer is comprised of the following:

1. A multi-head self-attention mechanism.
2. A position-wise fully connected feed-forward network.

In the decoder, each layer includes an additional multi-head attention sub-layer that attends over the encoder's output.

3.3 Self-Attention Mechanism

The self-attention mechanism allows the model to dynamically focus on different parts of the input sequence. The core operation is the scaled dot-product attention. For given query $\mathbf{Q}$, key $\mathbf{K}$, and value $\mathbf{V}$ matrices, the attention output is computed as

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}, \tag{1}$$

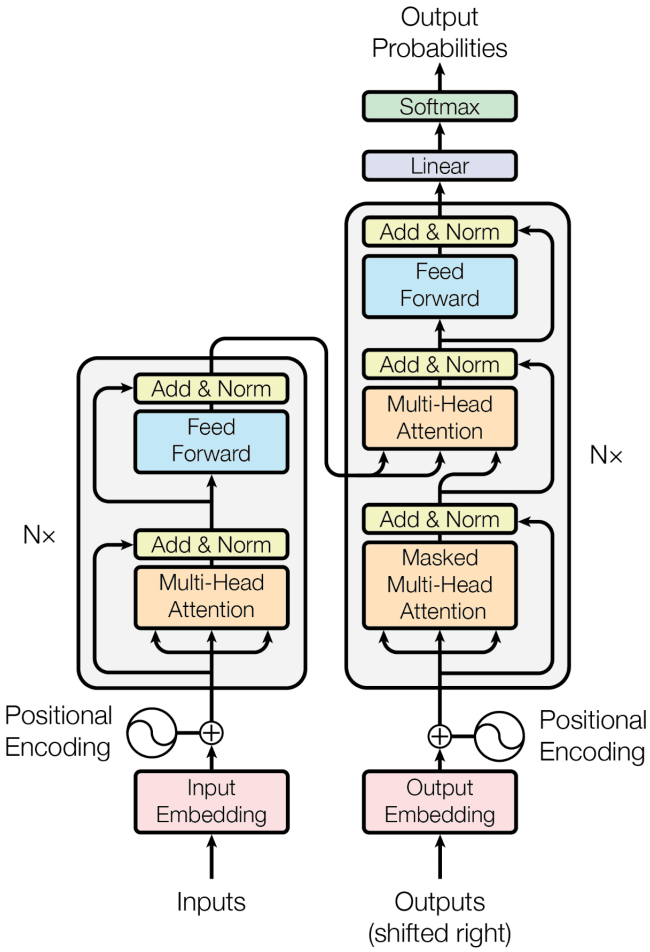

Figure 1: The transformer model architecture. The model consists of an encoder (left) and a decoder (right), each containing multiple stacked layers (denoted as N×). The encoder processes input tokens by passing them through embedding layers, positional encoding, multi-head self-attention, and feed-forward networks, with layer normalisation applied after each step. The decoder follows a similar structure but includes an additional masked multi-head attention mechanism to ensure autoregressive generation, preventing positions from attending to future tokens. The final output probabilities are computed through a linear layer and softmax activation.

where d_k is the dimensionality of the key vectors. The factor $\sqrt{d_k}$ helps to maintain stable gradients by scaling down the dot products.

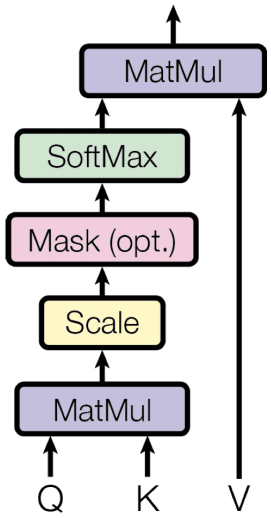

Figure 2: Illustration of the scaled dot-product attention mechanism. Given query $\mathbf{Q}$, key $\mathbf{K}$, and value $\mathbf{V}$ matrices, the attention scores are computed by performing a matrix multiplication between $\mathbf{Q}$ and $\mathbf{K}^T$, followed by a scaling factor $\frac{1}{\sqrt{d_k}}$ to stabilise gradients. An optional masking step is applied in the decoder to prevent positions from attending to future tokens. The scores are then passed through a softmax function to generate attention weights, which are used to weight the value $\mathbf{V}$ matrix. The final attention output is obtained via matrix multiplication between the attention weights and $\mathbf{V}$, allowing the model to focus on the most relevant parts of the input sequence.

3.4 Multi-Head Attention

To enhance the model's ability to capture various aspects of relationships between tokens, the transformer employs multiple attention heads. This approach involves projecting the queries, keys, and values into different subspaces and computing attention in parallel. It is given by

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\mathbf{head}_1, \ldots, \mathbf{head}_h)\mathbf{W}^O, \tag{2}$$

where each attention head is given by

$$\mathbf{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^{\mathbf{Q}}, \mathbf{K}\mathbf{W}_i^{\mathbf{K}}, \mathbf{V}\mathbf{W}_i^{\mathbf{V}}).$$

Here, $\mathbf{W}_i^{\mathbf{Q}}$, $\mathbf{W}_i^{\mathbf{K}}$, $\mathbf{W}_i^{\mathbf{V}}$, and $\mathbf{W}^{\mathbf{O}}$ are learned linear projection matrices.

3.5 Positional Encoding

Since the transformer does not have any recurrence or convolution to capture sequence order, positional encodings are added to the input embeddings. They inject information about the token positions using sine and cosine functions. The two functions used are

$$\text{PE}_{(pos,2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right), \tag{3}$$

$$\text{PE}_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right). \tag{4}$$

where *pos* represents the position in the sequence and i denotes the dimension index. These functions generate unique positional vectors that the model adds to the word embeddings.

3.6 Feed-Forward Networks

Each layer in both the encoder and decoder includes a feed-forward network applied to each position independently. This network consists of two linear transformations with a ReLU activation in between. It is given by

$$\text{FFN}(x) = \max(0, x\mathbf{W}_1 + b_1)\mathbf{W}_2 + b_2. \tag{5}$$

This component processes the output of the attention mechanisms and adds non-linearity to the model, aiding in the transformation of the attended information.

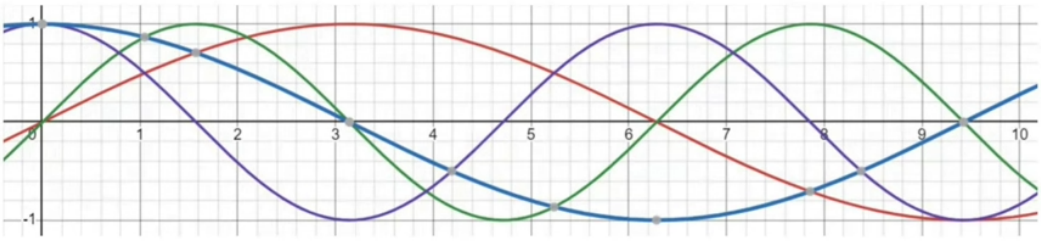

Figure 3: Visualisation of the sine and cosine functions used for positional encoding. Unlike recurrent models, which inherently process sequences step-by-step, transformers lack a built-in notion of order. To overcome this, they use positional encodings, which are added to the input embeddings to provide a unique representation for each position in the sequence. These encodings are generated using sine and cosine functions at different frequencies. The choice of these functions ensure that the encoding captures both absolute and relative positional information. Different frequencies allow the model to recognise repeating patterns and long-range dependencies, even in sequences longer than those seen during training. Since they produce smooth, continuous variations across positions, they provide the model with a structured way to differentiate words based on their position while maintaining meaningful distance relationships between them.

4 Experimental Setup and Results

4.1 Datasets and Evaluation Metrics

4.2 Training Details

4.3 Results Analysis

5 Discussion

6 Conclusion

References

Dzmitry Bahdanau. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.