A gist of Attention is all you need

Introduction

Recurrent neural networks (RNNs), including long short-term memory (LSTM) and gated recurrent units (GRUs), have been the standard for sequence modeling tasks like language modeling and machine translation. These models process sequences step-by-step, which limits parallelization and efficiency, especially with longer sequences. Attention mechanisms have improved sequence modeling by allowing dependencies to be modeled regardless of their distance in the sequence. However, these mechanisms are usually combined with recurrent networks.

The Transformer model, introduced in this work, eliminates recurrence entirely and relies solely on attention mechanisms to capture global dependencies between inputs and outputs. This architecture allows for greater parallelization and achieves state-of-the-art performance in translation tasks with significantly reduced training time.

Background

The extract explains that models like Extended Neural GPU, ByteNet, and ConvS2S use convolutional neural networks to compute hidden representations in parallel, but they struggle with learning dependencies between distant positions due to the increasing number of operations required. The Transformer model addresses this by using self-attention, which reduces the number of operations to a constant, though it initially reduces resolution. This is counteracted with Multi-Head Attention.

Self-attention, also known as intra-attention, relates different positions within a sequence to compute its representation and has been effective in various tasks like reading comprehension and summarization. Unlike previous models, the Transformer relies entirely on self-attention without using recurrent neural networks (RNNs) or convolution.

Model Architecture

Most advanced neural sequence transduction models utilize an encoder-decoder framework. In this setup, the encoder transforms an input sequence of symbol representations $(x_1,\ldots,x_n)$ into a sequence of continuous representations $z = (z_1,\ldots,z_n)$. The decoder then produces an output sequence $(y_1,\ldots,y_m)$ one symbol at a time, using previously generated symbols as additional input at each step in an auto-regressive manner. The Transformer model adheres to this architecture but employs stacked self-attention and point-wise, fully connected layers for both the encoder and decoder.

The encoder consists of 6 identical layers, each with two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. Residual connections and layer normalization are applied around each sub-layer, with all sub-layers and embedding layers producing outputs of dimension $d_{model} = 512$.

The decoder consists of 6 identical layers, each with three sub-layers: a multi-head self-attention mechanism, a position-wise fully connected feed-forward network, and a multi-head attention mechanism over the encoder's output. Residual connections and layer normalization are applied around each sub-layer. The self-attention sub-layer is modified to prevent positions from attending to subsequent positions, ensuring predictions for position *i* depend only on known outputs at positions less than *i*.

Attention Function

An attention function maps a query and a set of key-value pairs to an output, with all elements being vectors. The output is a weighted sum of the values, where the weights are determined by a compatibility function between the query and the corresponding key.

Scaled Dot-Product Attention

In Scaled Dot-Product Attention, the input consists of queries and keys of dimension d_k, and values of dimension d_v. It computes the dot products of the query with all keys, divide each by $\sqrt{d_k}$, and apply a SoftMax function to obtain the weights on the values.

Multi-Head Attention

To enhance the attention mechanism, instead of using d-dimensional keys, values, and queries, the model linearly projects them h times into d_k, d_k, and d_v dimensions, respectively. The attention function is then performed in parallel on these projected versions, resulting in d_v-dimensional output values.

The Transformer model employs multi-head attention in three distinct ways:

1. **Encoder-Decoder Attention**: Queries come from the previous decoder layer, while keys and values come from the encoder's output. This allows each position in the decoder to attend to all positions in the input sequence.
2. **Encoder Self-Attention**: Keys, values, and queries all come from the same source, the previous layer's output in the encoder. Each position in the encoder can attend to all positions in the previous encoder layer.
3. **Decoder Self-Attention**: Each position in the decoder can attend to all positions up to and including itself. To maintain the auto-regressive property, leftward information flow is prevented by masking out illegal connections in the scaled dot-product attention.

The need of Self Attention

This section compares self-attention layers with recurrent and convolutional layers for mapping variable-length sequences. The comparison focuses on three aspects:

1. Computational Complexity: Self-attention layers connect all positions with a constant number of sequential operations, whereas recurrent layers require (O(n)) sequential operations. Self-attention layers are faster than recurrent layers when the sequence length (n) is smaller than the representation dimensionality (d).
2. Parallelization: Self-attention allows for more parallelization compared to recurrent layers, which is beneficial for computational efficiency.
3. Path Length for Long-Range Dependencies: Self-attention layers have shorter path lengths between any two positions in the input and output sequences, making it easier to learn long-range dependencies. Recurrent layers have longer path lengths, which can hinder learning long-range dependencies.

Additionally, self-attention layers can be restricted to a neighborhood of size r to improve computational performance for very long sequences, increasing the maximum path length to O(n/r). Convolutional layers require a stack of layers to connect all input and output positions, increasing the path length and computational complexity. Separable convolutions reduce complexity but still have higher complexity compared to self-attention layers combined with point-wise feed-forward layers.

Self-attention also offers more interpretable models, with individual attention heads learning different tasks and exhibiting behavior related to the syntactic and semantic structure of sentences.