How Self-Attention Revolutionizes Contextual Information Gain – A Comprehensive Approach

Patrick Gottschling
(1155173254@link.cuhk.edu.hk)
Department of Computer Science and Engineering
The Chinese University of Hong Kong
(January 2022)

Abstract

This work elaborates on the architecture and advantages of the self-attention mechanism in the Encoder of Transformer Neural Networks. Self-attention is a tool used to gain more contextualized information from raw input data like texts or images efficiently. It puts attention on relevant information in the data to rapidly extract the important contents comparable to humans, pulling specifically needed information from all the endless input data detected by their senses.

Introduction

In the past years, Transformer Neural Networks have been replacing Recurrent Neural Networks (RNN) and Long-Short-Term-Memory (LSTM) successively, improving upon Natural Language Processing (NLP) applications. Its main advantage is that it allows parallelization by processing input sequences simultaneously instead of sequentially, unlike LSTM networks where the next token can only be processed when the previous one has been processed completely. As modern GPUs are designed for parallel computing, this sequential approach appears to not use its hardware resources efficiently. Non-sequential approaches, like Transformers, also avoid the issue of the vanishing gradient, which occurs in RNN/LSTM models, that leads to difficulties in remembering information in long sequences.

State-of-the-Art Transformer models like BERT or GPT-3 implement several new technologies leading to them outperforming previous models. In most cases, these technologies try to use the concept of attention to retrieve relevant information and context efficiently from raw data. This concept stands in contrast to the previously known idea of trying to pass information from cell to cell as known from RNNs.

In this work, we further elaborate on the Multi-Head-Self-Attention mechanism applied in the encoder of the Transformer architecture, using machine translation as its application. The idea is to improve the word embedding vector of a sentence by considering contextualized information. Word embeddings encode a word from any language, such as English, to a numerical vector. Similar words are represented with similar vectors, making them numerically comparable in context and meaning. Transformer models usually use pretrained libraries as their baseline input embeddings. The goal is to adjust this initial vector by considering the specific, individual context.

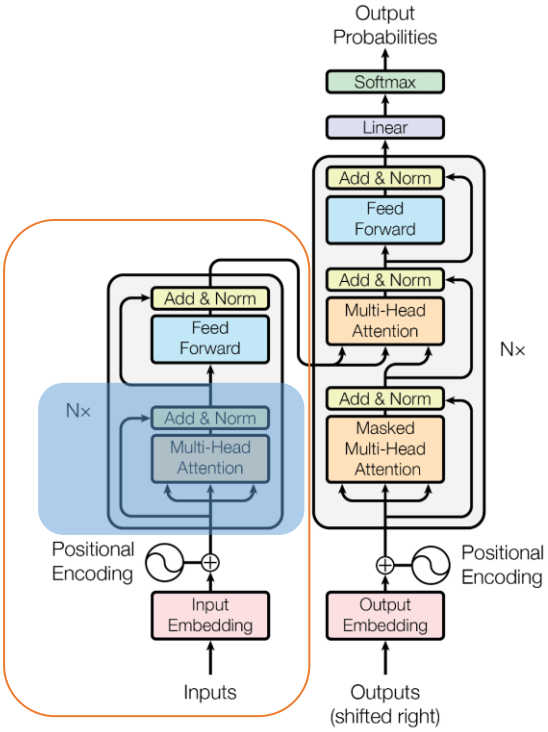

Fig.1 The Transformer architecture – The Encoder in the orange box and the Multi-Head-Self-Attention layers in the blue box. (from Vaswani et al.,2017,modified)

Theory and Background

The idea of self-attention is to recalculate the word embeddings of input tokens to contextualized word embedding vectors. Usually, word embedding vectors have a dimension of 100-300, which has been shown to be the best amount (Pennington et al.,2014). As the name reveals, this process only refers to itself, more precise, only to the tokens of the same sequence (e.g., the

sentence). The attention mechanism is built of a key, query, and value vector, all of which are initialized with the same initial input embedding vectors of a sentence, after positional encoding has been applied (Vaswani et al.,2017). Positional encoding adds information to the position of the word in the sentence (Kazemnejad, 2019) and are used as a first step to increase the contextual value of information.

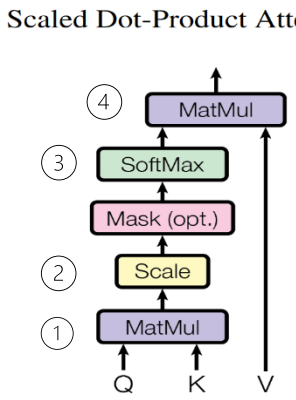

Fig.2 The self-attention architecture with Query Q, Key K and Value V (from Vaswani et al.,2017)

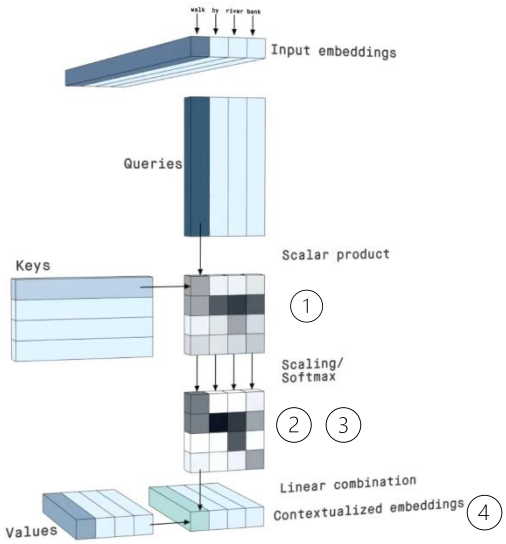

Fig.3 Visualization of the process (from "Pelarion",2020)

Fig.2 and Fig.3 visualize the self-attention process step-by-step. Each column represents the word embedding vector of a single token. This matrix, composed of the words embedding vectors of the sentence, is then multiplied with itself (Keys x Queries or $Q*K^T$ in the equation in Fig.4). As a result, this new matrix expresses the similarity of every token to all the other tokens of the sequence. Obviously, the diagonal elements of the matrix are high values, as seen in Fig.3. Next, the values are first scaled by taking the square root of the vector's dimension. Hence, by applying a softmax layer, the results are scaled between 0 and 1, summing up to 1.

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

Fig.4 Equation of self-attention in Transformer (from Vaswani et al.,2017)

The last step is to multiply the scaled matrix with the value vector which is once again only a copy of the input embedding. Subsequently, the final contextualized vector is received for every token, considering the contextual dependencies between the word in a sentence.

Up to this point, only the actions of a one single head have been mentioned. However, this architecture is called Multi-Head-Attention, as it is composed of several of these heads. For instance, one of the most famous Transformer models, called BERT, uses 8 of these heads with each head focusing its token comparison on different features.

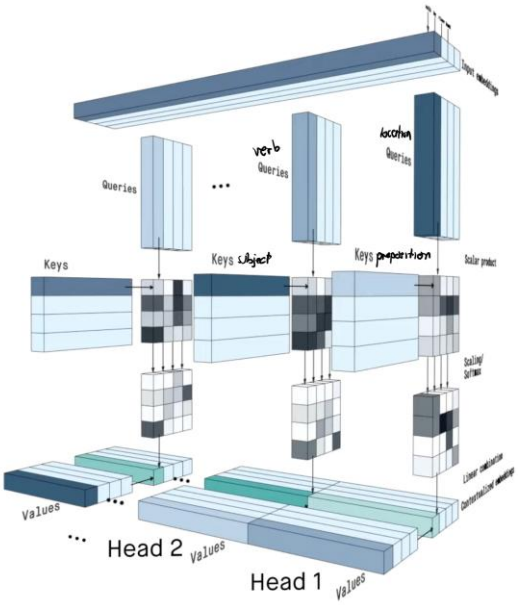

Fig.5 The multi-head-attention architecture (from "Pelarion",2020)

The result vectors are consequently concatenated as shown in Fig.5. Finally, a simple feed forward layer is applied (see architecture in Fig.1) to transform the concatenated results from all the heads into a form that can be used by the decoder. For example, one could just retrieve the initial word embedding vector size and interpret it as the contextualized word embedding vector. The result is subsequently, an adjusted word embedding vector for each word. In other words, a new

position in the 200–300-dimensional space for every word is retrieved, considering the individual contextual information of that specific sequence.

Proposed Implementations and Procedures

Once these complex models are pre-trained on excessive amount of data, they can be applied very easily. A good example of this is BERT's training corpus consisting of BooksCorpus, which holds 800 million words, and English Wikipedia, which holds 2.5 billion words (Devlin et al.,2019). Some fine-tuning may be used to help specify the task, however, different fine-tuned models and models using BERT or GPT-3 as baseline can easily be applied in Python with Tensorflow/Pytorch (Models available on Huggingface: https://huggingface.co/docs/transformers/index).

In another research work, we use the PEGASUS abstractive summarization model (Zhang et al.,2020) to summarize scientific papers using models provided in the Transformers library. Using different fine-tuned models, it is easy to create pretty accurate abstractive summaries of the input papers by just running the program on the local CPU. This underlines the fact that once the baseline is pretrained, the application of the model does not require too much computational cost. In our work we use the BERT-based PEGASUS model to retrieve several summaries with different focus and summarization technique, such as the length of the summary by using different fine-tuned models. Likewise, any other Transformer-based application could be implemented, leading to state-of-art results in machine translation, question answering, summarization, and many more.

Discussion and Conclusion

The Multi-Head-Self-Attention is often considered at the core innovation and improvement of Transformer Neural Networks. Using parallel computation by processing every token individually, the concept is highly suitable for today's GPUs. The amount of contextual data for relatively low computational cost makes it the State-of-Art approach in many applications.

The general idea is to put attention on relevant information of the input data.

In Transformer models this concept is used multiple times, not only for self-attention in the encoder. Another new technique called Masked Multi Head Attention is applied for the output sequence preprocessing in the decoder.

In Natural Language Processing (NLP) Transformers have replaced most of the other approaches, such as LSTM, however, Transformers are not only limited to language processing. It becomes a hot topic in many fields of Artificial Intelligence, far beyond Natural Language Processing.

References

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, 2017. https://arxiv.org/pdf/1706.03762.pdf

Jeffrey Pennington, Richard Socher, Christopher D. Manning. GloVe: Global Vectors for Word Representation, 2014. https://nlp.stanford.edu/pubs/glove.pdf

Amirhossein Kazemnejad. Transformer Architecture: The Positional Encoding, 2019. https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

Pelarion. How to get meaning from text with language model BERT | AI Explained, 2020. https://www.youtube.com/watch?v=-9vVhYEXeyQ

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, BERT, 2019. https://arxiv.org/pdf/1810.04805.pdf

Jingqing Zhang, Yao Zhao, Mohammad Saleh, Peter J. Liu. PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization, 2020. https://arxiv.org/pdf/1912.08777.pdf