Search Medium

Write

Overview: The Implemented Transformer

Hunter Phillips · Following

4 min read · May 9

269

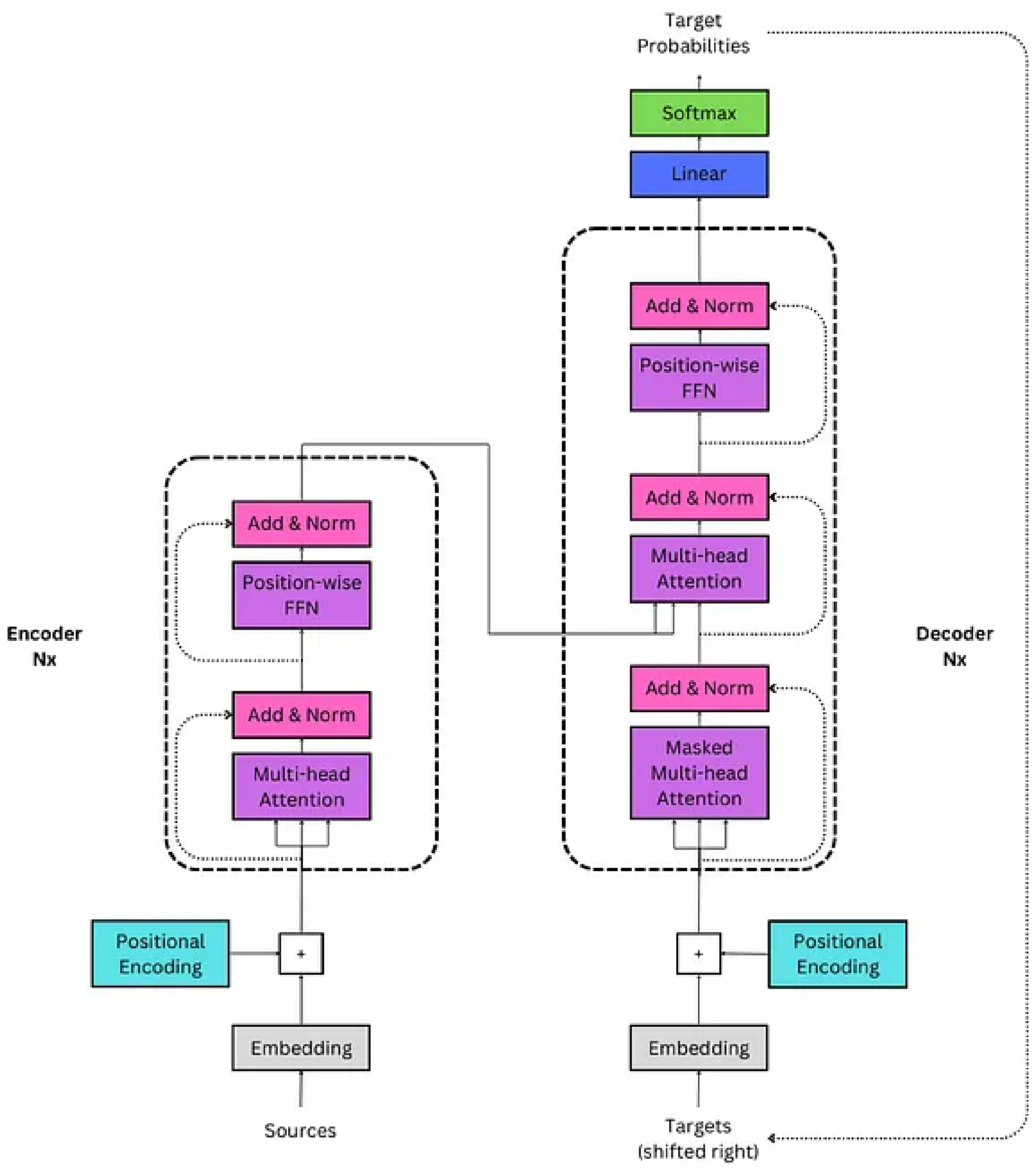


Image by Author

The transformer is a state-of-the-art model introduced in “Attention is All You Need” in 2017. There are great articles describing various components of transformers in varying amounts of detail.

My goal is to build off of these articles by explaining as many prerequisites as possible, and I plan to add more examples and intuition behind each implementation.

For these articles, I will be using PyTorch as my framework of choice, and I will explain each module as they are used.

Before digging into each of the components, it would be useful to understand the transformer's architecture.

The Transformer

The transformer is an encoder-decoder model. The encoder block is used to extract contextual features from a sequence, and the decoder block uses these features to generate an output sequence. The original paper uses the transformer to translate between languages, but it is also used for other natural language processing tasks and computer vision. This implementation will cover the translation task. On the in-depth image at the beginning of the article, the *Nx* on each side of the encoder and decoder demonstrate that multiple encoders and decoders are stacked upon each other to create the overall architecture. An elaborate, yet simplified, view can be seen below:

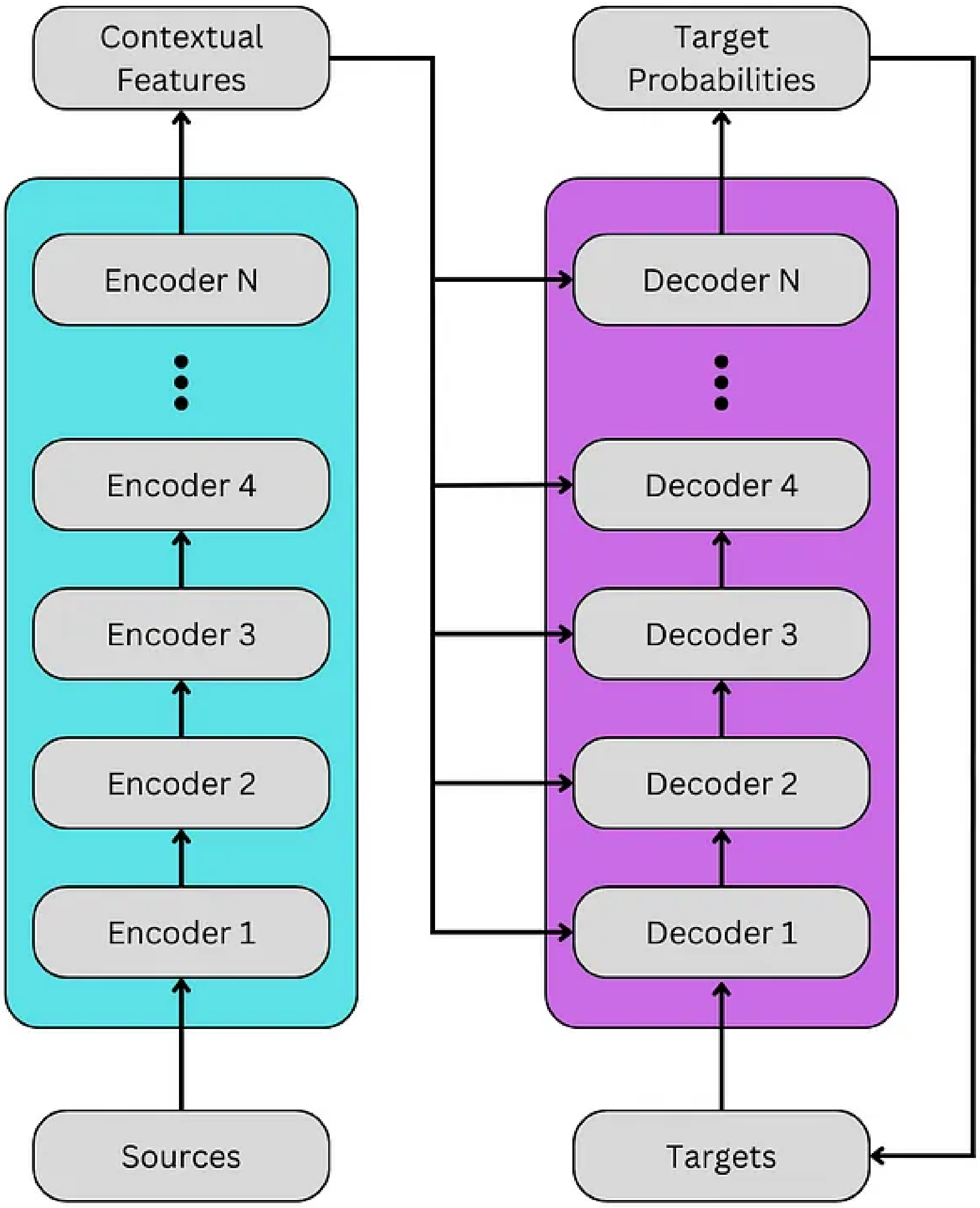


Image by Author

Essentially, the output of each encoder layer is passed to the next encoder layer, and then the output of the encoder block is passed to each decoder layer. Then, the output is generated using this information and the target inputs to the decoder block.

To train, the model requires a parallel corpus of the source and target languages, which will be German and English in this series. Both German and English sequences will be used during training to help the model learn the relationship between the languages.

At inference, an input to the model will be a German sentence, like "*<bos> Wie heißt du? <eos>"*, which will be encoded and passed to each layer of the decoder block. The decoder block will be prompted with a beginning-of-sequence token, "*<bos>"*, and it will use the encoded German sequence to predict the first English token. Ideally, this would be "*What"*. The model is then provided the German sequence, the "*<bos>"* token, and "*What"***.** These are used to predict the next English token, which should be "**is**". This process is repeated until the end-of-sequence token, "*<eos>"* is predicted. Since the sequence begins with "*<bos>*", the output is considered to be shifted right since "*<bos>"* token is not part of the prediction. A summary of this process can be seen below:

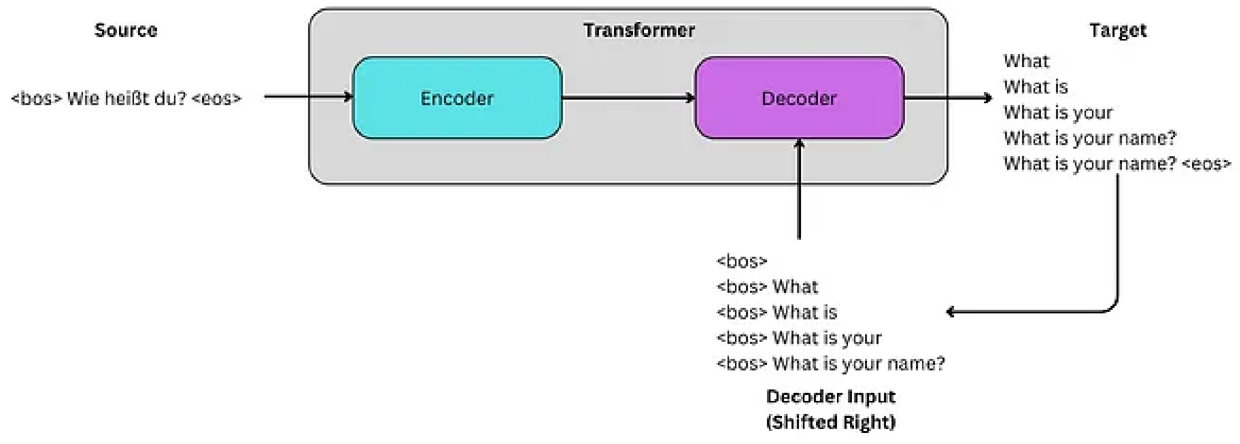


Image by Author

The Components

Before the encoder and decoder are used, the sequences are embedded and positionally encoded to provide as much information about each word and its context as possible. These sequences are passed through the multi-head attention layer and feed-forward network. They are responsible for extracting useful information from the source and target languages during training. After each of these sublayers, layer normalization and residual addition are also performed.

Multi-Head Attention

According to KiKaBeN, the multi-head attention layer is used to “enrich each token (embedding) vector with contextual information from the whole sentence.” In many languages, a single word can have many functions or meanings depending on its context. Therefore, KiKaBeN explains that the “self-attention mechanism employs multiple heads (eight parallel attention calculations) so that the model can tap into different embedding subspaces.”

Position-Wise Feed-Forward Network

The position-wise feed-forward network (FFN) consists of two linear layers with a ReLU activation function between them. It is also known as an expand-and-contract network because the first layer has a higher dimensionality, and the second layer returns to the original dimensionality. KiKaBeN proposes that the goal is to “introduce non-linearity (ReLU) without losing much information thanks to the intermediate dimensionality expansion.”

Residual Addition

Residual addition takes the embeddings before they were passed into the layer and adds them to the output. This enriches the embedding vectors with the information obtained from the multi-head attention and FFN layers.

Layer Normalization

Layer normalization maintains the mean and standard deviation of each embedding vector, or token, to help prevent issues with gradient descent.

The Implementation

As of now, the following articles will be created:

1. The Embedding Layer
2. Positional Encoding
3. Multi-Head Attention
4. Position-Wise Feed-Forward Network
5. Layer Normalization
6. The Encoder
7. The Decoder
8. Putting it Together: The Implemented Transformer

In these posts, I will also be referring to other articles that were fundamental to my learning. This list will continue to grow:

1. Dropout
2. Softmax
3. Tensors
4. Broadcasting
5. The Dot Product

Finally, as a reminder, each of these articles helped me learn the material. However, I acknowledge that I could have misunderstood or misrepresented certain topics, so please let me know if you have any recommendations for improvement!

References

1. KiKaBeN’s Transformer’s Encoder-Decoder: Let’s Understand The Model Architecture

Transformers Overview Machine Learning Translation Artificial Intelligence

More from the list: "NLP"

Curated by Himanshu Birla

Jon Gi... in Towards Data ...

Characteristics of Word Embeddings

· 11 min read · Sep 4, 2021

Jon Gi... in Towards Data ...

The Word2vec Hyperparameters

· 6 min read · Sep 3, 2021

Jon Gi... in

The Word2ve

· 15 min rea

View list

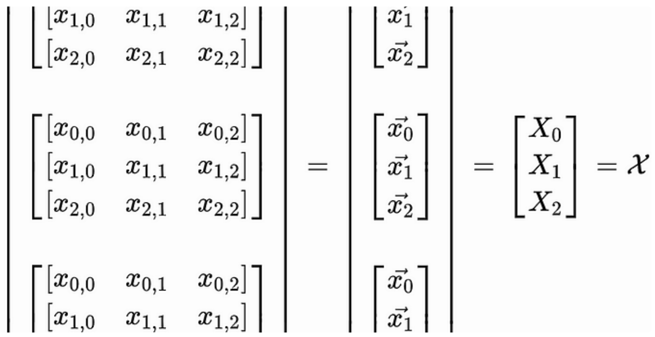
Written by Hunter Phillips

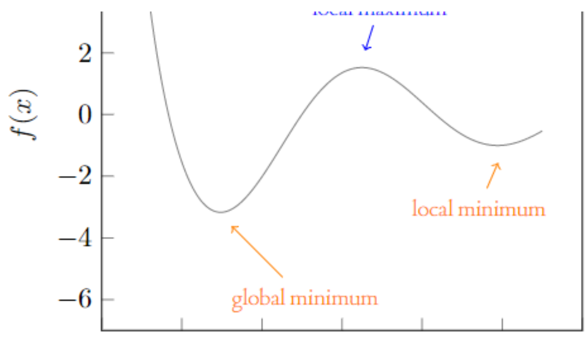
Following

219 Followers

Machine Learning Engineer and Data Scientist

More from Hunter Phillips





Hunter Phillips

A Simple Introduction to Tensors

Hunter Phillips

A Simple Introduction to Gradient Descent

A tensor is a generalization of vectors and matrices to n dimensions. Understanding ho...

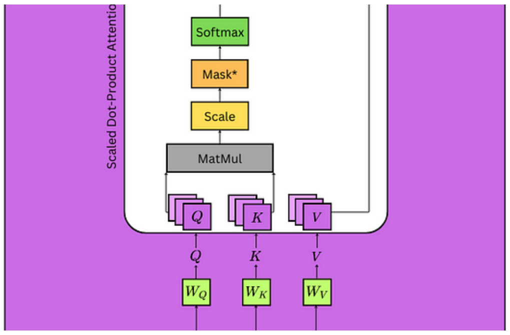
11 min read · May 10

273 5

Gradient descent is one of the most common optimization algorithms in machine learning....

10 min read · May 18

108



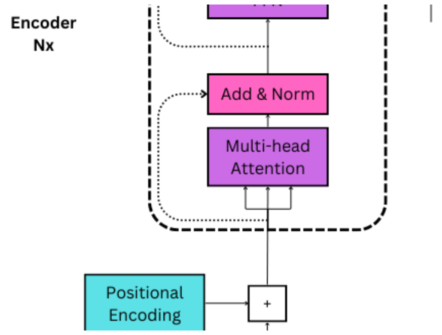
Hunter Phillips

Multi-Head Attention

This article is the third in The Implemented Transformer series. It introduces the multi-...

25 min read · May 9

167



Hunter Phillips

Position-Wise Feed-Forward Network (FFN)

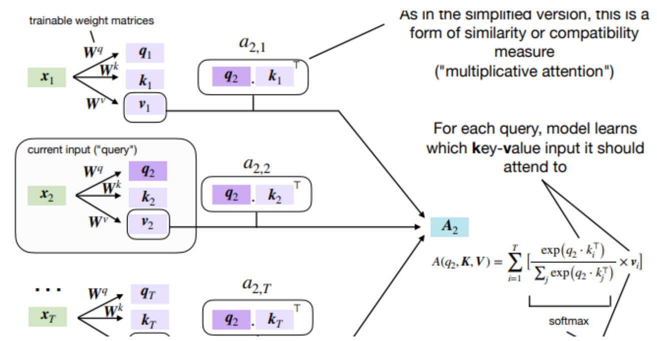
This is the fourth article in The Implemented Transformer series. The Position-wise Feed-...

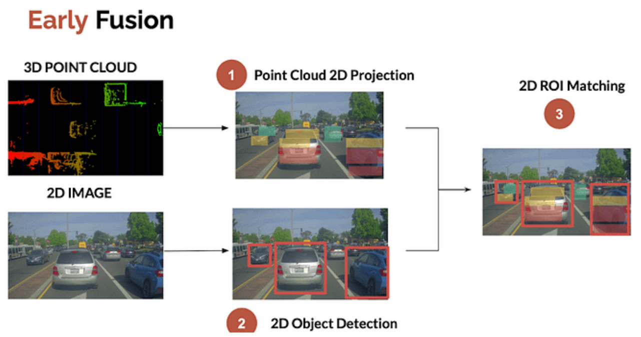
9 min read · May 9

155

See all from Hunter Phillips

Recommended from Medium


trainable weight matrices
As in the simplified version, this is a form of similarity or compatibility measure ("multiplicative attention")
For each query, model learns which key-value input it should attend to
current input ("query")
softmax



Zain ul Abideen

Attention Is All You Need: The Core Idea of the Transformer

An overview of the Transformer model and its key components.

6 min read · Jun 26

144

Chris Thaliyath

Deep Learning with PointCloud, Image and different kind of senso…

How to bring Lidar and Camera on the same coordinate frame and overlay the image on…

6 min read · Sep 6

21

Lists

Predictive Modeling w/ Python

20 stories · 452 saves

Natural Language Processing

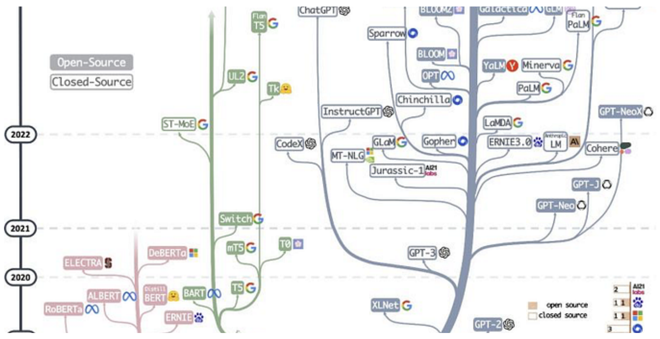
669 stories · 283 saves

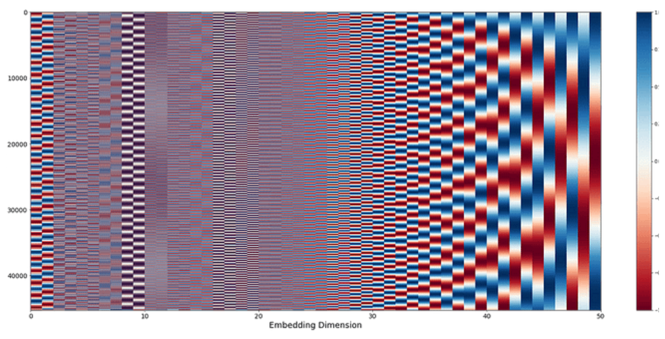
AI Regulation

6 stories · 138 saves

ChatGPT prompts

24 stories · 459 saves





Haifeng Li

A Tutorial on LLM

Generative artificial intelligence (GenAI), especially ChatGPT, captures everyone's...

15 min read · Sep 14

372

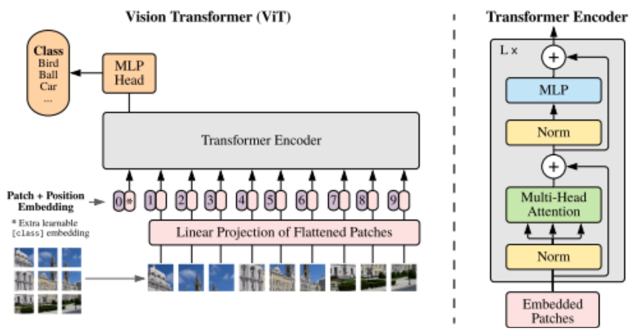
Eugene Ku

Transformer Architecture (Part 1— Positional Encoding)

Nowadays, arguably the most popular and influential model behind the hypes of deep...

4 min read · Aug 22

15



Walter Sperat

Using Optuna the wrong way

If you have lived under a rock for the last couple of years, Optuna is a Python library...

7 min read · Jun 8

23

Vikram Pande

CNNs and Vision Transformers: Analysis and Comparison

Exploring the effectiveness of Vision Transformers and Convolutional Neural...

6 min read · Apr 29

208 6

See more recommendations