Intro to Transformer

Hi everyone, in this video I will explain to you all the concepts and features of Transformer, an AI model that has been revolutionizing not only the NLP space but also the whole AI world.

The Transformer model is proposed in the famous paper: ‘Attention Is All You Need’, published by Vaswani and co-authors in 2017 at Google Brain and Google Research.

But why this publication is a game changer? The answer is that it solved the problems remaining in NLP for a long time till 2017, and then open the way to the super AI model that can understand and generate human-like language.

So first, let’s take a look which models have dominated NLP before Transformer.

Until 2017, the common architectures like Recurrent Neural Networks (RNNs), and its variants Long Short-Term Memory (LSTM) networks, Gated Recurrent Units (GRU) were the mainstream in NLP tasks.

RNNs offer several benefits, including but not limited to:

* Efficient handling of sequential data types such as text, speech, and time series.
* Ability to process inputs of variable lengths, a feature lacking in feedforward neural networks.
* Enhanced training efficiency due to weight sharing across different time steps.

Now let’s see some major limitations of RNNs.

Take language modeling task as an example. Given a sentence starts with some words, like “She stands up and opens the …” ,the model needs to predict the next word. By using RNN, the model processes one word at a time step, to generate the hidden state for computation in the next time step, meaning that it is a sequential computation. As a consequence, there are less rooms for parallel computation. So longer sequence, longer computation time.

Next, due to its nature of sequential computation to product the hidden states, the contributions of initial states or information to the final state or the prediction are very small for long sequence context. This leads to the loss of information for long-range dependencies.

Another major limitation of RNNs is that they are prone to vanishing or exploding gradient problems. RNN use Backpropagation Through Time to updates the weights. For example, to calculate the gradient of loss function L with respect to the parameter weights of the network, it uses the chain rule, to calculate the product of gradients across time steps.

If the gradients are less than 1, each multiplication operation leads to a decrease in the magnitude of the gradient. If this process continues over many time steps, the gradient eventually diminishes to zero. As a result, the updates to the weights during optimization become insignificant, hindering the learning process, especially for long sequences or deep architectures.

Conversely, the exploding gradient problem occurs when the gradients at each time step are greater than 1, leading to exponential growth of gradients as they propagate backward through time, as a result the product of large gradients across time steps can result in extremely large gradient values. The large gradient values can cause instability during optimization, leading to weight updates that oscillate or diverge, making the training process highly unstable.

Both issues hinder the training of RNNs, affecting their ability to effectively capture long-term dependencies in sequential data. These problems make training RNN unstable and extremely hard.

To sum up, there are some major disadvantages of RNNs:

* + Sequential computation, hard to parallel computation with GPU
  + Loss of information for long-term dependencies
  + Vanishing or exploding gradient problems

And then, the arrival of Transformer in 2017 proposed by the paper “Attention Is All You Need”.

This paper proposed Transformer architecture for machine translation task, for example translate from one language to another, like from English to French. At the high level overview, its architecture includes encoder and decoder. In this video, we will deep dive into the two main blocks of Transformer.

What makes transformer so innovative at that time?

Transformer with Self-Attention mechanisms allow parallel computation, meaning that we can leverage GPU to accelerate the training process. Furthermore, It is able to capture long-range dependencies. Last but not least, its architecture allows the model less prone to vanishing or exploding gradient problems.

Followed by the success of Transformers, since 2018 onwards, there is a new trend in NLP to develop Pre-trained language models based on the Transformer architecture, to name a few: BERT, GPT, T5, Llama, Mistral, Phi, Falcon, OLMo.

----

Now let deep dive into transformer. Here is our plans of attack. I am going to explain to you

* What are Transformer’s **block-by-block**?
* What are the **input-output-process** of each block?
* Talk about input/output **shape, type**
* How to **Implement** and **playground** with **notebook** for each block

First of all, Input block