Diving into the Revolutionary paper of Google, Attention is all you need

---

Title of the paper: Attention Is All You Need
Authors: Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

Intro

In this homework, I investigated the above-mentioned paper. I divide the paper concept into the following topics. So, the reader could follow the complex concept of the paper with peace of mind.

1. Prerequisites

In this section I will provide basic concepts that we should know before going further.

Artificial Neuron

Generally speaking, the Artificial neuron (AN) is a mathematical function that receives some input, applies some mathematical operation on them and then pass them into the output. So, each AN is constituted from three parts: 1- Input, 2 Process 3- Output [1]. (Figure 1)

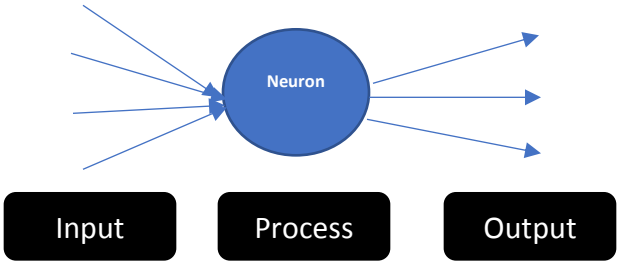

Figure 1: Shows a simple Artificial Neuron's part. It gets number as input as after processing pass them into its output

Since the essence of the ANs is coming from the Biological Neuron concept, we call the neuron process part, Activation function (Same as biological neural neuron that should be activated to transfer a massage.). [1]

Regarding the Bio neurons something interesting is that those can selectively transfer a massage and manipulate it. Similarly, in the ANs, we have input parameters (Named neuron's weights and biases) that controls and manipulate massage passing. In the below I demonstrated the mathematical model which is correspond to the concept of the ANs. See Figure 2.[2]

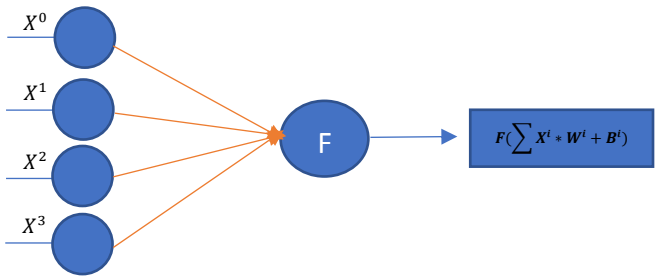

Figure 2: Mathematical Model of an Artificial Neuron

2. Artificial neural network

Artificial neural networks (ANNs) are an interconnection of a stack of neurons which are concatenated to form a layer of neurons. It can have a variety of neuron counts. The neurons in ANNs can be grouped together in a single layer, or they can be divided into two, three, or even more layers. The difference between a single-layer and a multilayer ANN construction is seen in Figure 3.[2]

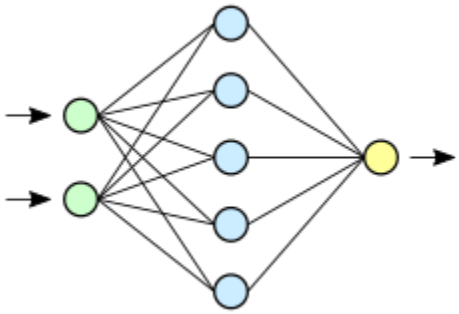

Figure 3: shows a neural network with one hidden layer (blue), one input layer (green) and one output layer

(Single neuron)

3. Learning

Generally speaking, the Learning process in ANNs is adjusting the neurons parameters in order to the network deliver what wee expected. To do so, usually there exist a trainset that we pass

its data to the network with randomly initialize parameters. Then, we compare the result of the network with the result that we would expect.

Then, by changing the network parameters we will try to modify the result of the network to be close to the expected answer. To do so, we define a function that measures the network error (distance between the expected result and and the network current result) we named this function the loss function. Next, we will try to find the minimum of the loss function through the computing the gradient of the loss function and solving the $\nabla Loss\ function(trainset) = 0$. The answer of the previous equation is the network training point.[2]

4. Learning limits (Problematic)

As we mentioned above, In the neural network training process we use the gradient concept in mathematics. It is worth noting that in solving $\nabla Loss\ function(trainset) = 0$, is not feasible in the real word since it is too much complex. Therefore, to solve the above equation we have to use an approximative algorithm which its name is gradient descent.

The Gradient Descent algorithm is a recursive algorithm and need to be done multiple times to be able to gives us an appropriate answer. The big problem is when if there are lots of hidden layer in a neural network, the gradient descent value will be vanished or exploded after some iteration and it cannot give us a good answer. As a result, the training process will be failed. [3]

5. Different ANNs architectures

To Overcome the aforementioned issue, which is one the biggest challenges in the ANNs, the researchers have designed many different neural network architectures such as: Convolutional neural networks, RNNs, LSTMS, and …

However, all the architectures have their own training limit. As a result, those are not able to be trained on the datasets that have complex logical relationships between their elements. For example, none of them cannot gives us a good performance to learn a textbook and answers our questions from the textbook.

Therefore, the authors of the paper introduced a novel neural network that can learn long term dependencies between dataset elements and realizes complex logical relationships camper to the conventional neural architectures.

Before going through the details, we need to become familiar with attentions mechanism that authors have used in their neural network design. In the following we will explain the Attention Mechanism.[4]

6. Attention Mechanism

According to to our brain's cognitive function, when we want to do a task such as reading a sentence, observing a picture and … our brain do it in a very intelligent way. First, it searchs for useable data then categorizes them and does not pay attention to the rest of the data.

For instance, when somebody gives us a picture of a red car in a jungle and he wants us to see whether there is a driver in the car, probably we do not pay attention to the trees at all. This simple brain method gives the brain a very powerful cognitive power. Because using the attention it can manage how to optimize the brain power allocation on useful data and forgeting the rest.

Similarly, In the ANNs we can implement such an attention mechanism so that the neural network does not need to consume too much processing power in order to do a specific task.

In the Figure 4, we can see that the ANN allocates more parameters to what it needs for car recognition.[5]

Figure 4: Simple attention mechanism for car recognition task. As shown, the ANNs is trained to allocate more parameters to what it needs to recognize the car.

Till now, we learned what was the problem due to conventional neural networks and what are the answers to cover the weakness of the networks. If we would summarize it in a couple of paragraphs, we can say:

To do complex tasks on complex dataset we have to use neural architectures that have lots of hidden layers. However, increasing the hidden layers will result in the gradient descent vanishing/exploding problem. To overcome the issue, we can change the interconnections of the hidden layers and therefore, change the architecture.

But still, this technique cannot compensate the gradient descent vanishing/exploding affect when it comes to train a network on an extremely depended dataset such as a textbook to build a Question answering System for example.

In the following we will explain a more complex and efficient version of the attention by which the authors designed their architecture that outperform the conventional architectures.

7. Multiheaded Attention (State of the art part 1)

So far, we discussed the simple attention mechanism. Now, let see how we can learn complex dependencies more efficiently. To do so, imagine that we have a dataset that its elements have lots of features. Suppose we have formed those features union set named F. Therefore, each

dataset elements feature is a member of the $F = \{f_0, f_1, \dots, f_n\}$. Now, theoretically in order to extract the $f_i s$ we should design a ANNs such that it is trainable on the $F_{Trainset}$, where $F_{Trainset}$ is the feature set of the trainset.

To do so, one amazing technique that the authors used is to run a stack of ANNs which have simple attention mechanism in parallel but each ANN will train on an specific subset of the F set. To put this in an other way, let's define $ANN = \{ANN_0, ANN_1, \dots, ANN_k\}$ is the stack of the ANNs where the ANN_i is the neural network that just pay attention to the specific subset of the F. That is, the attentions scope of the ANN_i is $\{f_{t0}, f_{t1}, \dots, f_{ts}\} \subset F$. Using this technique, the $ANN = \{ANN_0, ANN_1, \dots, ANN_k\}$ will have an spectacular power to extract the features.

For instance, imagine that we want to design a car recognizer network. Instead of the training an ANN plus attention mechanism on a dataset of car pictures we can train an stack of ANN such as the $ANN = \{ANN_0, ANN_1, \dots, ANN_k\}$ where, ANN_0 just looks for tire, ANN_1 looks for headlight of car, ANN_2 looks for the car logo and so on[6]. See Figure 5.

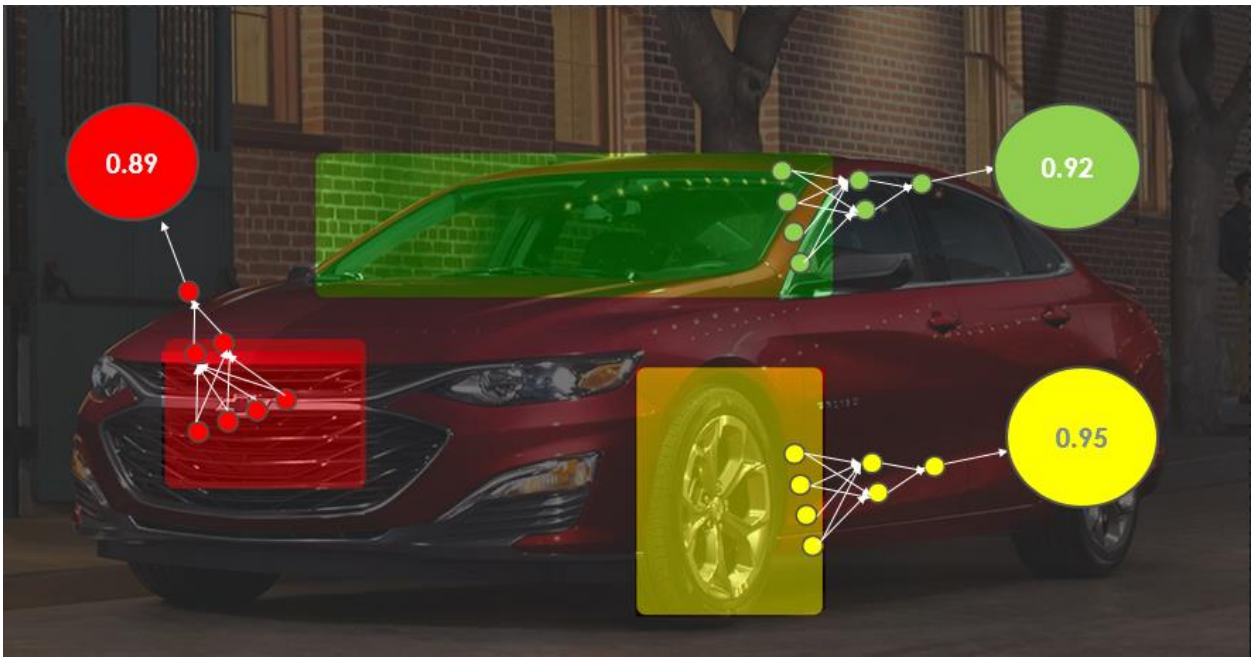

Figure 5: Shows multiheaded attention mechanism. Each ANNs (the red, green, and blue one) is trained to look for a specific feature subset. For example, the red, green, and yellow are looking for car logo, windshield, and tire respectively.

In the following we will explain a specific type of the neural network the Auto Encoder. That is used for unsupervised feature extraction. The Authors used the Auto Encoder along with the multiheaded attention to build a novel neural architecture the transformer. It uses the multiheaded attention to be trained on large text corpus dataset and building a Language translators or question answering systems.

8. Autoencoders

Autoencoders are constituted of an Encoder and a Decoder part where the encoder and decoder are an ANN. Generally speaking, the Encoder part take the data, compress them and pass them

to the decoder. Then, the decoder takes the compressed data and decompress them into their initial form. One interesting thing is happening that through this compressing and decompressing process the autoencoder learns how to generate data from a randomized input. In other words, it learns the knowledge to generate data itself. As a result, this type of the neural networks can be used for machine translation for instance[7]. See Figure 6.

Figure 6: shows the Schematic of a machine translator autoencoder.

In the following we will explain the transformer architecture which is the state of the art of the paper.

9. Transformers (State of the art part 2)

So far, we have learned autoencoder and multiheaded attention systems. Now, we have learned all the stuff to realizes what is a transformer. Basically, A transformer is just an auto encoder that equipped with many layers of multiheaded attention. As a result, the transformer can easily learn the large dataset such as text corpus and can extract the concepts from text.

Speaking in more detail, in a transformer, the encoder part is constituted from a multiheaded attention layer that takes the data and then pass it into a feedforward neural network. Also, there are some residues connections between different parts of the encoder to prevent the gradient vanishing/exploding problem[6]. See Figure 7.

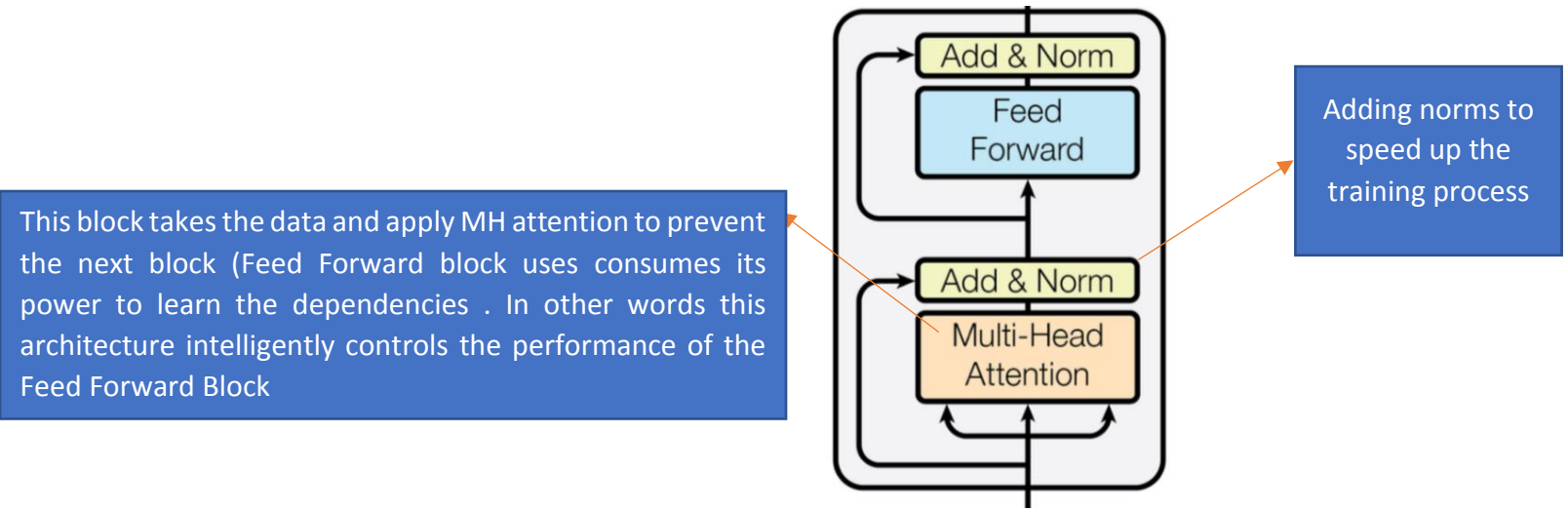

Figure 7: Shows the encoder block of the transformer. As it is shown, the data passes through the MH attention block and then using some residue connection passes through the feed forward neural net.

The mechanism of the decoder part is same as the decoder part.

10. References

1. Hecht-Nielsen, R., *Theory of the backpropagation neural network*, in *Neural networks for perception*. 1992, Elsevier. p. 65-93.
2. Hornik, K., M. Stinchcombe, and H. White, *Multilayer feedforward networks are universal approximators.* Neural networks, 1989. **2**(5): p. 359-366.
3. Du, S., J. Lee, H. Li, L. Wang, and X. Zhai. *Gradient descent finds global minima of deep neural networks*. in *International Conference on Machine Learning*. 2019. PMLR.
4. Shahsavar, A., S. Khanmohammadi, D. Toghraie, and H. Salihepour, *Experimental investigation and develop ANNs by introducing the suitable architectures and training algorithms supported by sensitivity analysis: measure thermal conductivity and viscosity for liquid paraffin based nanofluid containing Al2O3 nanoparticles.* Journal of Molecular Liquids, 2019. **276**: p. 850-860.
5. Yao, K., G. Zweig, and B. Peng, *Attention with intention for a neural network conversation model.* arXiv preprint arXiv:1510.08565, 2015.
6. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, *Attention is all you need.* arXiv preprint arXiv:1706.03762, 2017.
7. Wang, Y., H. Yao, and S. Zhao, *Auto-encoder based dimensionality reduction.* Neurocomputing, 2016. **184**: p. 232-242.