Details about the Transformer-style Network

Yuhao Cheng

https://yuhaocheng.github.io/

December 20, 2023

Outline

Contents

Attention is the metaphorical term that describes the network or system recognizing some critical part in the input stuff, just like human beings. And the transformer network is just one of the successful forms.

Before the transformer, we have two ways of attention:

- Hard Attention: Using the one-hot coding to indicate which part is important
- Soft Attention: Using the softmax to provide softer values

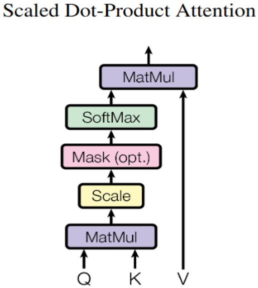
Contents

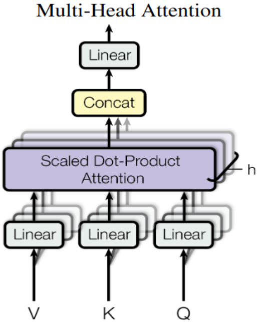
Attention Mechanism

Attention computation

The bloom of the attention, or in the other words, the transformer architecture, is from [4].

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (1)$$





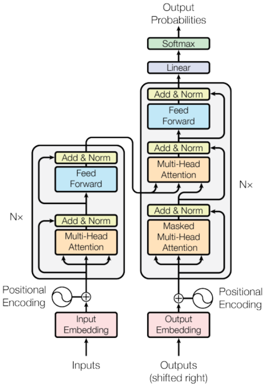
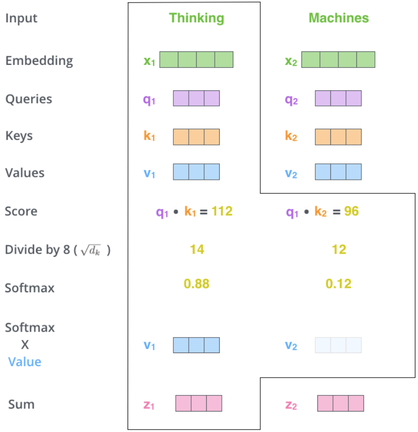


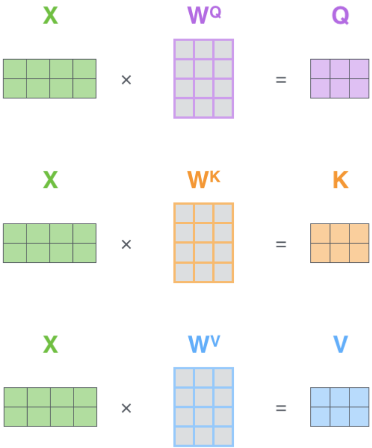
Figure 1: The architecture of the transformer

Intuitive Description
One Token

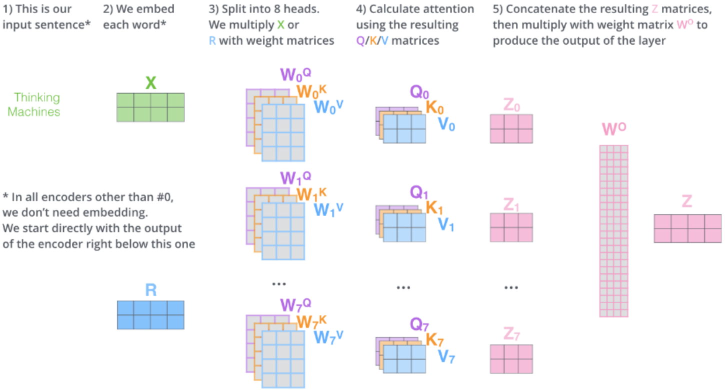


The example of one word(token) in one sequence. And in this figure, the word is "Thinking".

- The "Thinking" comes to the embedded feature, x_1. For this part, we does not consider what the difference between the tokenizer and encoder.
- Meanwhile, the q_1, k_1 and v_1 will be generated by using different weight matrices.
- For the each token, we will get self-attention result z_1.



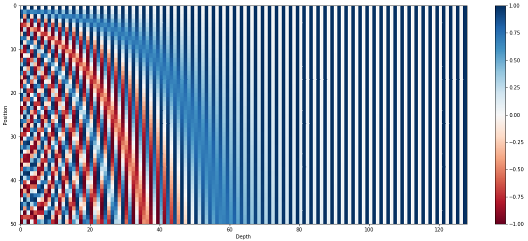
And for many tokens, the Transformer can stack the words and use matrices to get the values



1) This is our input sentence*
2) We embed each word*
3) Split into 8 heads. We multiply X or R with weight matrices
4) Calculate attention using the resulting Q/K/V matrices
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer
Thinking Machines
X
* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one
R
W0Q W0K W0V
W1Q W1K W1V
W7Q W7K W7V
Q0 K0 V0
Q1 K1 V1
Q7 K7 V7
Z0
Z1
Z7
...
WO
Z

Only having the self-attention will make the model not know the sequential information of each position. So in order to solve this problem, researchers use the **Positional Embedding** to make the model get the sequential information.

$$
\begin{aligned}
PE_{(pos,2i)} &= \sin\left(pos/10000^{2i/d_{\mathrm{model}}}\right) \\
PE_{(pos,2i+1)} &= \cos\left(pos/10000^{2i/d_{\mathrm{model}}}\right)
\end{aligned}
\tag{2}
$$



Generative Pre-trained Transformer(GPT)

Generative Pre-trained Transformer(GPT)[2] is proposed by OpenAI in 2018. It proposes using the self-supervised learning to train the large Language model.

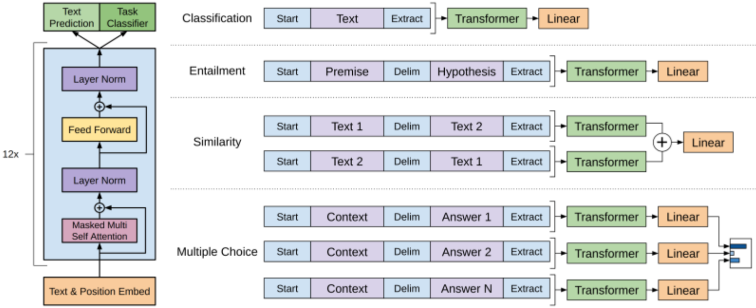


Figure 2: (left)Transformer architecture and training objectives used in this work. (right) Input transformations for fine-tuning on different tasks. We convert all structured inputs into token sequences to be processed by our pre-trained model, followed by a linear+softmax layer.

GPT-series

From the GPT-1, they gradually improve it to develop the GPT-2[3], GPT-3[1] and so on. However, they do not have so much update on the core technologies. They just use more data and more weights to train. And the performance's increasing also proves the advantages of the large model.

	Release	#Parameters	Model Store	Pre-trained Data
GPT-1	2018.6	0.117B	0.468GB	$\sim$ 5GB
GPT-2	2019.2	1.5B	6GB	40GB
GPT-3	2020.5	175B	700GB	45TB

Table 1: The comparison of GPT-series. The model size is computed if the weights are store in Float32. $1B = 10^9.10^9 \times 4bits = 4 \times 10^9 bytes = 4 \times 10^6 KB = 4 \times 10^3 MB = 4GB$

Contents

Tokenizer

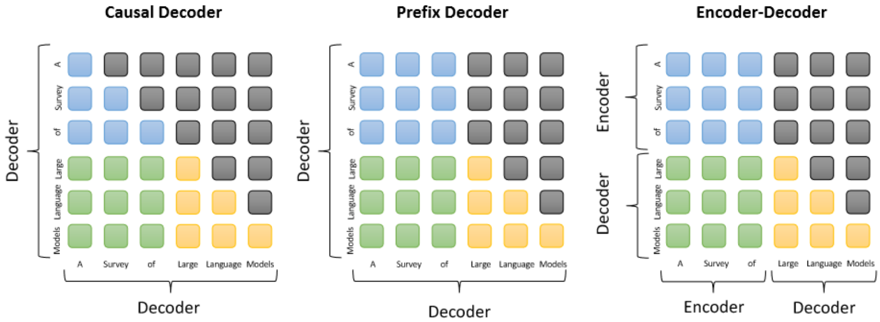
- Tokenizer is more a model trained through likelihood
- One or several words generate one tokenizer

Embedding

- Embedding is like the Encoder part
- A sequence to generate one vector

Supplement Knowledge

Current Network Architecture,Why only decoder?



Here, we use a example that the input is "**A Survey of**" and the Language model will output "**Large Language Models**". Blue: the attention between prefix tokens. Green: attention between prefix and target tokens. Yellow: attention between target tokens. Grey: masked attention

Configuration	Method	Equation
Normalization position	Post Norm [22]	$\text{Norm}(\mathbf{x}+\text{Sublayer}(\mathbf{x}))$
	Pre Norm [26]	$\mathbf{x}+\text{Sublayer}(\text{Norm}(\mathbf{x}))$
	Sandwich Norm [255]	$\mathbf{x}+\text{Norm}(\text{Sublayer}(\text{Norm}(\mathbf{x})))$
Normalization method	LayerNorm [256]	$\frac{\mathbf{x}-\mu}{\sigma}\cdot\gamma+\beta,\quad \mu=\frac{1}{d}\sum_{i=1}^{d}x_i,\quad \sigma=\sqrt{\frac{1}{d}\sum_{i=1}^{d}(x_i-\mu))^2}$
	RMSNorm [257]	$\frac{\mathbf{x}}{\text{RMS}(\mathbf{x})}\cdot\gamma,\quad \text{RMS}(\mathbf{x})=\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}$
	DeepNorm [258]	$\text{LayerNorm}(\alpha\cdot\mathbf{x}+\text{Sublayer}(\mathbf{x}))$
Activation function	ReLU [259]	$\text{ReLU}(\mathbf{x})=\max(\mathbf{x},\mathbf{0})$
	GeLU [260]	$\text{GeLU}(\mathbf{x})=0.5\mathbf{x}\otimes[1+\text{erf}(\mathbf{x}/\sqrt{2})],\quad \text{erf}(x)=\frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$
	Swish [261]	$\text{Swish}(\mathbf{x})=\mathbf{x}\otimes\text{sigmoid}(\mathbf{x})$
	SwiGLU [262]	$\text{SwiGLU}(\mathbf{x_1},\mathbf{x_2})=\text{Swish}(\mathbf{x_1})\otimes\mathbf{x_2}$
	GeGLU [262]	$\text{GeGLU}(\mathbf{x_1},\mathbf{x_2})=\text{GeLU}(\mathbf{x_1})\otimes\mathbf{x_2}$
Position embedding	Absolute [22]	$\mathbf{x}_i=\mathbf{x}_i+\mathbf{p}_i$
	Relative [82]	$A_{ij}=\mathbf{W}_q\mathbf{x}_i\mathbf{x}_j^T\mathbf{W}_k^T+r_{i-j}$
	RoPE [263]	$A_{ij}=\mathbf{W}_q\mathbf{x}_i\mathbf{R}_{\Theta,i-j}\mathbf{x}_j^T\mathbf{W}_k^T=(\mathbf{W}_q\mathbf{x}_i\mathbf{R}_{\Theta,i})(\mathbf{W}_k\mathbf{x}_jR_{\Theta,j})^T$
	ALiBi [264]	$A_{ij}=\mathbf{W}_q\mathbf{x}_i\mathbf{x}_j^T\mathbf{W}_k^T-m(i-j)$

Thanks

Reference I

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al.
Language models are few-shot learners.
Advances in neural information processing systems, 33:1877–1901, 2020.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al.
Improving language understanding by generative pre-training.
2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al.
Language models are unsupervised multitask learners.
OpenAI blog, 1(8):9, 2019.

Reference II

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin.
Attention is all you need.
Advances in neural information processing systems, 30, 2017.