CS7150.37338.202130 Day 12 Preparation Questions

Aishwarya Vantipuli, Harshita Ved, Aveek Choudhury

TOTAL POINTS

5 / 5

QUESTION 1

1 Question 2 **1 / 1**

✓ **- 0 pts** Correct

- 0.1 pts Click here to replace this description.

QUESTION 2

2 Question 3 **1 / 1**

✓ **- 0 pts** Correct

- 0.1 pts Click here to replace this description.

- 0.25 pts provide more details

- 0.5 pts don't copy from the paper

- 0.25 pts Click here to replace this description.

- 0.5 pts Some inaccuracies and missing key points

QUESTION 3

3 Question 4 **1 / 1**

✓ **- 0 pts** Correct

- 0.25 pts Click here to replace this description.

- 1 pts don't copy from the paper

QUESTION 4

4 Question 5 **1 / 1**

✓ **- 0 pts** Correct

QUESTION 5

5 Question 6 **1 / 1**

✓ **- 0 pts** Correct

- 0.2 pts Interpretation and observations

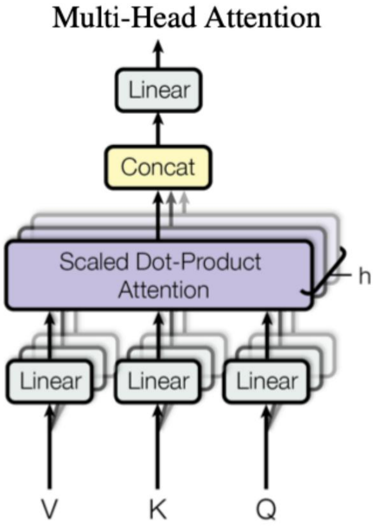
💬 very good

Question 2. *Explain multi-head attention.*

Response: Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. Hence, instead of performing a single attention function with d_{model}-dimensional keys, values and queries, it is more beneficial to linearly project the queries, keys and values h times with different, learned linear projections to d_k, d_k and d_v dimensions, respectively. On each of these projected versions of queries, keys and values, an attention function is performed in parallel, yielding d_v-dimensional output values. These are concatenated and once again projected, resulting in the final values, as depicted in Figure. With a single attention head, averaging inhibits this for multi-head attention.

$$\begin{aligned}\mathrm{MultiHead}(Q, K, V) &= \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O \\ \text{where } \mathrm{head_i} &= \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)\end{aligned}$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.



1 Question 2 **1 / 1**

✓ **- 0 pts** Correct

- 0.1 pts Click here to replace this description.

Question 3. *Explain the positional encoding that was used.*

Response: Positional encodings are used as the model contains no recurrence and no convolution, hence, in order for the model to make use of the order of the sequence, some information about the relative or absolute position of the tokens in the sequence was needed.
Positional encodings are added to the input embeddings at the bottoms of the encoder and decoder stacks. They have the same dimension d_{model} as the embeddings, so that the two can be summed.

sine and cosine functions of different frequencies are used:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to a sinusoid. The wavelengths form a geometric progression from 2π to $10000 \cdot 2\pi$.
This function was chosen because of hypothesis that it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, PE_{pos+k} can be represented as a linear function of PE_{pos}.

2 Question 3 **1 / 1**

✓ **- 0 pts** Correct

- 0.1 pts Click here to replace this description.

- 0.25 pts provide more details

- 0.5 pts don't copy from the paper

- 0.25 pts Click here to replace this description.

- 0.5 pts Some inaccuracies and missing key points

Question 4. *Explain the transformer architecture.*

Response: Transformer consists of six encoders and six decoders. Each layer of Encoder has two sub-layers: a multi-head self attention layer which helps the encoder look at other words in the input sentence as it encodes a specific word and a feed forward layer.

Each layer of Decoder has three sub-layers: in addition to the two sub-layers in Encoder, the Decoder inserts a third sub-layer, which performs multi-head attention over the output of the last encoder stack.

The word embeddings of the input sequence are passed to the first encoder. These are then transformed and propagated to the next encoder. The output from the last encoder in the encoder-stack is passed to all the decoders in the decoder-stack.

Self-attention is computed multiple times in the Transformer′s architecture, in parallel and independently. It is therefore referred to as Multi-head Attention. The outputs are concatenated and linearly transformed. Self-attention mechanism calculates only the relevance among the inputs or the outputs.

In addition to the self-attention and feed-forward layers, the decoders also have one more layer of Encoder-Decoder Attention layer. This helps the decoder focus on the appropriate parts of the input sequence.

Another important step on the Transformer is to add positional encoding when encoding each word. Encoding the position of each word is relevant, since the position of each word is relevant to the translation.

3 Question 4 1 / 1

✓ - **0 pts** Correct

- **0.25 pts** Click here to replace this description.

- **1 pts** don't copy from the paper

Question 5. *Compare and contrast transformers with recurrent neural networks.*

Response: RNN/CNN handle sequences word-by-word sequentially which is an obstacle to parallelize. Transformer achieves parallelization by replacing recurrence with attention and encoding the symbol position in the sequence. This, in turn, leads to significantly shorter training time.
RNNs need to propagate the error back in time through words one word at a time. Transformer sees all words simultaneously - so there is no back propagation through time.
Although Transformer is proved as the best model to handle really long sequences, the RNN and CNN based model could still work very well or even better than Transformer in the short-sequences task.

Directions: Read 'Language Models are Few-Shot Learners' (GPT-3).

- Read Section 1, 2, 3.0, 3.1

Question 6. *Explain Figure 1.2*

Response:

Context:

The GPT-3 autoregressive language model with varying number of parameters (1.3B, 13B and 175B) was trained to remove extraneous symbols from a word with and without "in-context" natural language prompt as well as varying number of examples, in order to compare the model performances.

What is plotted:

The accuracy of the models with 1.3B, 13B and 175B parameters is plotted as a function of the number of examples in context given, K for both variants of the models. The solid line represents the model with a natural language prompt whereas the dashed line indicates that no prompt was given.

What we observe:

The larger models with 175B parameters show a steeper curve in terms of accuracy improvement with increasing number of examples shown during training. When a natural task description prompt is provided, the 175B model curve shows considerably higher accuracy even with very few examples and eventually having similar accuracy to the model with no prompt as more examples are shown. The trend for a steeper curve when natural language prompt is provided can be seen in the cases of smaller models as well.

Interpretation:

Th steep "in-context learning curves", especially for the larger model, demonstrates the ability to learn a task from contextual description. It also demonstrates a general trend of improved performance with increase in model size and number of examples provided, though the latter might not be extremely beneficial in terms of generalizability.

4 Question 5 **1 / 1**

✓ **- 0 pts** Correct

Question 5. *Compare and contrast transformers with recurrent neural networks.*

Response: RNN/CNN handle sequences word-by-word sequentially which is an obstacle to parallelize. Transformer achieves parallelization by replacing recurrence with attention and encoding the symbol position in the sequence. This, in turn, leads to significantly shorter training time.
RNNs need to propagate the error back in time through words one word at a time. Transformer sees all words simultaneously - so there is no back propagation through time.
Although Transformer is proved as the best model to handle really long sequences, the RNN and CNN based model could still work very well or even better than Transformer in the short-sequences task.

Directions: Read 'Language Models are Few-Shot Learners' (GPT-3).

- Read Section 1, 2, 3.0, 3.1

Question 6. *Explain Figure 1.2*

Response:

Context:

The GPT-3 autoregressive language model with varying number of parameters (1.3B, 13B and 175B) was trained to remove extraneous symbols from a word with and without ”in-context” natural language prompt as well as varying number of examples, in order to compare the model performances.

What is plotted:

The accuracy of the models with 1.3B, 13B and 175B parameters is plotted as a function of the number of examples in context given, K for both variants of the models. The solid line represents the model with a natural language prompt whereas the dashed line indicates that no prompt was given.

What we observe:

The larger models with 175B parameters show a steeper curve in terms of accuracy improvement with increasing number of examples shown during training. When a natural task description prompt is provided, the 175B model curve shows considerably higher accuracy even with very few examples and eventually having similar accuracy to the model with no prompt as more examples are shown. The trend for a steeper curve when natural language prompt is provided can be seen in the cases of smaller models as well.

Interpretation:

Th steep ”in-context learning curves”, especially for the larger model, demonstrates the ability to learn a task from contextual description. It also demonstrates a general trend of improved performance with increase in model size and number of examples provided, though the latter might not be extremely beneficial in terms of generalizability.

5 Question 6 **1 / 1**

✓ **- 0 pts** Correct

- 0.2 pts Interpretation and observations

💬 very good