ABBOUDI Mohammed Amine
mohammed-amine.abboudi@polytechnique.edu

Lab session # 7
ALTEGRAD 2020

01/22/20

1 Question 1

- In the case of an encoder-decoder framework, an attention of y over x entails that the queries would come from the decoder (output sequence) while the key-value pairs would come from the encoder (source sequence).
- In a self-attention framework all of the keys, values and queries come from the same place, in this case, the output of the previous layer of where x is situated (encoder or decoder). Each position in the encoder/decoder can attend to all positions in its previous layer.

2 Question 2

Proving the diagonal components (identities) of Table 1.
Axiom: The multiplication of $n \times m$ matrix by an $m \times p$ matrix is of complexity $\mathcal{O}(n \times m \times p)$.

- **Complexity of a Self attention layer:** each query matrix Q of size $n \times d$ is mutiplied by a matrix $K^\top$ of size $d \times n$ making this first dot-product of complexity $\mathcal{O}(n^2 \times d)$. A element wise division and a softmax are then carried out, counting for n^2 operations. Finally, the resulting $n \times n$ matrix is multiplied by $V \in \mathbb{R}^{n \times d}$, making the overall complexity $\mathcal{O}(n^2 \times d)$.
- **Complexity of a recurrent layer:** We have an input sequence of length n, we have a hidden state matrix of size $d \times d$, we carry out a matrix multiplication of complexity $\mathcal{O}(d^2)$ n times. The total complexity is therefore: $\mathcal{O}(n.d^2)$.
- **Sequential operations required in a recurrent layer:** A recurrent layer needs to be fed sequentially the entire sequence before it generates an output, making n its minimum number of sequential operations required.
- **Maximum Path Length for a convolutional layer:** A convolutional layer's ability to take into account dependencies between words is dictated by its kernel size k. Setting $k = 2$, for example, serves to capture the relationship that links each consecutive pair of words in a sentence. In this case the number of neurons that separate the first two words' dynamic and last two words' dynamic is $n/2$. In more general terms, the above-defined max path length is $\mathcal{O}(n/k)$ in the case of a convolutional layer of kernel size k.

3 Question 3

Having multiple heads of small dimension allows the model to jointly attend to information from different representation sub-spaces at different positions while making the computation parallelisable as well as cheaper per attention head, while a single attention head, some of the information is lost and the complexity is increased.

4 Question 4

Let PE be a $n \times d_{model}$ matrix where each row encodes position i onto an embedding of size d_{model}.
We set out to prove that there exists a linear transformation $A^{(k)} \in \mathbb{R}^{d_{model} \times d_{model}}$ s.t.

$$\mathbf{PE}_{i+k} = \mathbf{A}^{(k)}.\mathbf{PE}_i$$

holds for any positional offset k and for any valid position $i \in \{1, ..., n-k\}$.
In other terms, for each sine-cosine pair corresponding to a frequency $f_j = 10000^{2j/d_{model}}$ there exists a linear transformation $A^{(k)} \in \mathbb{R}^{2\times 2}$ independent of i that satisfies:

$$A^{(k)} \begin{bmatrix} \mathbf{sin}((i)/f_j) \\ \\ \mathbf{cos}((i)/f_j) \end{bmatrix} = \begin{bmatrix} \mathbf{sin}((i+k)/f_j) \\ \\ \mathbf{cos}((i+k)/fi) \end{bmatrix} \tag{1}$$

Proof:
Using the formulation in equation 1, we wish to find a, b, c and d such that:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} \textbf{sin}((i)/f_j) \\ \\ \textbf{cos}((i)/fi) \end{bmatrix} = \begin{bmatrix} \textbf{sin}((i+k)/f_j) \\ \\ \textbf{cos}((i+k)/fi) \end{bmatrix} \tag{2}$$

After applying the relevant trigonometric addition formulae and carrying out the matrix product, we are left with a 2 equation system:

$$\begin{cases} a.\textbf{sin}(i/f_j) + b.\textbf{cos}(i/f_j) = \textbf{cos}(k/f_j).\textbf{sin}(i/f_j) \ + \ \textbf{sin}(k/f_j).\textbf{cos}(i/f_j) \\ c.\textbf{sin}(i/f_j) + d.\textbf{cos}(i/f_j) = -\textbf{sin}(k/f_j).\textbf{sin}(i/f_j) + \textbf{cos}(k/f_j).\textbf{cos}(i/f_j) \end{cases}$$

Solving this system, we get:

$$a = cos(k/f_j), \qquad b = sin(k/f_j), \qquad c = -sin(k/f_j), \qquad d = cos(k/f_j)$$

$A(k)$ is therefore independent from the initial position i and satisfies equation 1. □
This method allows us to which eventually allows us to represent $\textbf{PE}_{i+k}$ as a linear function of (a rotation) of $\textbf{PE}_i$ for any k. This makes it easy for the model to learn to leverage relative positioning.

5 Question 5

The self-attention sub-layers with `maskout=True` in the decoder stack allow each position in the decoder to attend to all positions in the decoder up to and including that position. By setting to $-\infty$ the subsequent positions of the `SoftMax` input, the outputted corresponding entries are 0 and therefore excluded from affecting the backpropagation mechanism. They have no influence on the left-ward available words. In other words, it zeroes-out the similarities between words and the words that appear after the source words "in the future", preventing predictions from depending on knowing the answer before it gets predicted. Since we remove such information, it cannot be used by the model (especially during training), and we guarantee that only similarity to the preceding words is considered. In [2], the authors use a different kind of masking, one that is more related to the task to be performed. It masks only 15% of the sequence and asks the model to predict it (Language modeling). It therefore can have access to future and past values, hence the name "Bidirectional Transformers for Language Understanding".

6 Question 6

The model presented in the paper contains $6 \times 2 = 12$ identical feed-forward sub-layers and $6 \times 3 = 18$ multi-head attention identical sub-layers. Each **FFS** has $c_{FFS} = 512 \times 2048 \times 2 + 2048 + 512 = 2,099,712$ coefficients while each **MHAS** has $c_{MHAS} = coeffs(W_O) = 512 \times 512 = 262,144$ trainable parameters. The embedding layers map a vocabulary of $32,000$ tokens unto a 512 embedding space which makes their size : $c_{emb} = 32,000 \times 512 \times 2 = 32,768,000$. We can conclude that the total number of coefficients of the implemented model is : $c = 2,099,712 \times 6 \times 2 + 262,14 \times 6 \times 3 + 32,768,000 = 62,683,136$ parameters.
The authors used 8 Nvidia P100 GPUs and have trained the aforementioned model for 12 hours using a $32,000$ token vocabulary.
Assuming we have access to one GPU of the same model, due to the highly parallelisable nature of the architecture, we can approximate training time would take 8 times longer, making it a 4 day training session. This is not realistic for a local machine, and since platforms such as Google Colab limit continuous assignment of a **VM** to 12 hours, it seems quite improbable for the model to be trained with such limited infrastructure.

7 Question 7

Model takes too much time to train.

8 Question 8

BERT's [2] architecture makes it able to unambiguously represent multiple sentences separated by a certain token [SEP], it differs in the way it does so from [1] by adding more embedded information to the positional embedding and word embedding. Segmentation embeddings are learned emeddings that indicate which section the current token belongs to, as illustrated in figure 1.

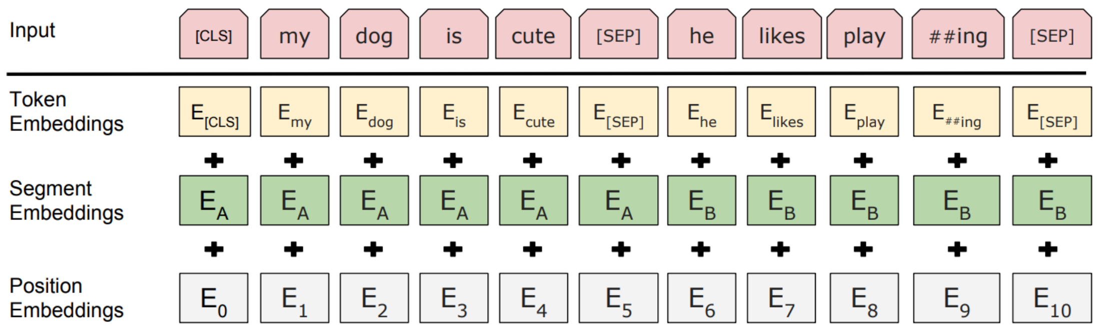

Figure 1: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

References

[1] Niki Parmar Jakob Uszkoreit Llion Jones Aidan N Gomez Łukasz Kaiser Ashish Vaswani, Noam Shazeer and Illia Polosukhin. Attention is all you need. 2017.

[2] Kenton Lee Jacob Devlin, Ming-Wei Chang and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018.