EECS 182 Deep Neural Networks
Fall 2022 Anant Sahai

Homework 8

This homework is due on Monday, Nov 14, 2022, at 10:59PM.

Deliverables: Please submit the code/notebooks/saved model as a zip file to the code gradescope assignment. Submit your written answers in the written gradescope assignment, and attach a pdf printout of the notebooks.

1. Coding Question: Summarization

Implement the Summarization.ipynb notebook and answer the following questions below.

(a) In the notebook, visualize attention weights for randomly-chosen Q, K, and V matrices. Would you expect weights to sum to 1 if we sum over the queries dimenstion, or over the keys dimension?

(b) Run the notebook cell which plots attention weights where Q and K are identical random matrices. Explain why we see the attention weights and outputs we do.

(c) (Optional) In the notebook, we showed that positional encodings make it easy to attend to relative positions. Derive why the transformation matrix used achieves the desired effect - i.e. derive a matrix $M \in \mathbb{R}^{d \times d}$ such that $p_{t+1} = Mp_t$, where p_t is the d-dimensional positional encoding for timestep t.

(d) In this model, the encoder and decoder share the same dimensions. Is this necessary, or could we choose to make them different?

(e) Run the notebook cell which visualizes attention masks, and explain the patterns you see.

(f) How would you modify the summary sampling procedure if you wanted to generate multiple summaries per article rather than just one?

(g) (Optional) How does the Performer model's performance match the performance with standard softmax attention? Explain how you reached this conclusion (Did you test with multiple sets of hyperparameters? Include plots to support your answer.)

2. Coding Question: Visualizing Attention

Please run the cells in the Visualizing_BERT.ipynb notebook, then answer the questions below.

(a) Attention in GPT: Run part a of the notebook and generate the corresponding visualizations

  i. What similarities and differences do you notice in the visualizations between the examples in this part? Explore the queries, keys, and values to identify any interesting patterns associated with the attention mechanism.

  ii. How does attention differ between the different layers of the GPT model? Do you notice that the tokens are attending to different tokens as we go through the layers of the netowork?

(b) BERT pays attention: Run part b of the notebook and generate the corresponding visualizations.

  i. Look at different layers of the BERT model in the visualizations of part (b) and identify different patterns associated with the attention mechanism. Explore the queries, keys, and values to further inform your answer. For instance, do you notice that any particular type of tokens are attended to at a given timestep?

ii. Do you spot any differences between how attention works in GPT vs. BERT? Think about how the model architectures are different.

iii. For the example with syntactically similar but definitionally different sentences, look through the different layers of the two BERT networks associated with sentence a and sentence b, and take a look at the queries, keys, and values associated with the different tokens. Do you notice any differences in the embeddings learned for the two sentences that are essentially identical in structure but different in meaning?

iv. For the pre-training related examples, do you notice BERT's bi-directionality in play? Do you think pre-training the BERT helped it learn better representations?

(c) BERT has multiple heads!: Run part c of the notebook and generate the corresponding visualizations.

i. Do you notice different features being learned throughout the different attention heads of BERT? Why do you think this might be?

ii. Can you identify any of the different features that the different attention heads are focusing on?

(d) Visualizing untrained attention weights

i. What differences do you notice in the attention patterns between the randomly initialized and trained BERT models?

ii. What are some words or tokens that you would expect strong attention between? What might you guess about the gradients of this attention head for those words?

3. Kernelized Linear Attention

This is a continuation of Problem 3 on HW 6 (https://inst.eecs.berkeley.edu/~cs182/fa22/assets/assignments/hw6_sol.pdf). Please refer to this part for notation and context. In Part 1 of this problem, we considered ways to efficiently express the attention operation when sequences are long (e.g. a long document). Attention uses the following equation:

$$V_i' = \frac{\sum_{j=1}^{N} \text{sim}\left(Q_i, K_j\right) V_j}{\sum_{j=1}^{N} \text{sim}\left(Q_i, K_j\right)}. \tag{1}$$

We saw that when the similarity function is a kernel function (i.e. if we can write $\text{sim}\left(Q_i, K_j\right) = \Phi(Q_i)^T \Phi(K_j)$ for some function Φ), then we can use the associative property of matrix multiplication to simplify the formula to

$$V_i' = \frac{\phi\left(Q_i\right)^T \sum_{j=1}^{N} \phi\left(K_j\right) V_j^T}{\phi\left(Q_i\right)^T \sum_{j=1}^{N} \phi\left(K_j\right)}. \tag{2}$$

If we use a polynomial kernel with degree 2, this gives a computational cost of $\mathcal{O}\left(ND^2M\right)$, which for very large N is favorable to the softmax attention computational cost of $\mathcal{O}\left(N^2 \max\left(D, M\right)\right)$. ($N$ is the sequence length, D is the feature dimension of the queries and keys, and M is the feature dimension of the values. Now, we will see whether we can use kernel attention to directly approximate the softmax attention:

$$V_i' = \frac{\sum_{j=1}^{N} \exp(\frac{Q_i^T K_j}{\sqrt{D}}) V_j}{\sum_{j=1}^{N} \exp(\frac{Q_i^T K_j}{\sqrt{D}})}. \tag{3}$$

(a) Approximating softmax attention with linearized kernel attention

i. As a first step, we can use Gaussian Kernel $\mathcal{K}_{\text{Gauss}}(q,k) = \exp(\frac{-||q-k||_2^2}{2\sigma^2})$ to rewrite the softmax similarity function, where $\text{sim}_{\text{softmax}}(q,k) = \exp(\frac{q^T k}{\sqrt{D}})$. Assuming we can have $\sigma^2 = \sqrt{D}$, **rewrite the softmax similarity function using Gaussian Kernel.**. (*Hint: You can write the softmax* $\exp(\frac{-||q-k||_2^2}{2\sigma^2})$ *as the product of the Gaussian Kernel and two other terms.)*

ii. However, writing softmax attention using a Gaussian kernel does not directly enjoy the benefits of the reduced complexity using the feature map. This is because the feature map of Guassian kernel usually comes from the Taylor expression of $\exp(\cdot)$, whose computation is still expensive [1]. However, we can approximate the Guassian kernel using random feature map and then reduce the computation cost. (Rahimi and Recht, 2007)[2] proposed random Fourier features to approximate a desired shift-invariant kernel. The method nonlinearly transforms a pair of vectors q and k using a **random feature map** $\phi_{\text{random}}()$; the inner product between $\phi(q)$ and $\phi(k)$ approximates the kernel evaluation on q and k. More precisely:

$$\phi_{\text{random}}(q) = \sqrt{\frac{1}{D_{\text{random}}}}\left[\sin\left(\mathbf{w}_1 q\right), \ldots, \sin\left(\mathbf{w}_{D_{\text{random}}} q\right), \cos\left(\mathbf{w}_1 q\right), \ldots, \cos\left(\mathbf{w}_{D_{\text{random}}} q\right)\right]^\top. \tag{4}$$

Where we have D_{random} of D-dimensional random vectors $\mathbf{w}_i$ independently sampled from $\mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}_D)$,

$$\mathbb{E}_{\mathbf{w}_i}\left[\phi(q) \cdot \phi(k)\right] = \exp\left(-\|q-k\|^2 / 2\sigma^2\right). \tag{5}$$

Use ϕ_{random} to approximate the above softmax similarity function with Gaussian Kernel and derive the computation cost for computing all the V' here.

iii. However, we should note this approximation is not perfect, **use the HELPER.PY function to plot the approximation error of ϕ_{random} attention and softmax attention scales in terms of D and D_{random}, summarize what you find below.**

(b) (Optional) Beyond self-attention, an autoregressive case will be masking the attention computation such that the i-th position can only be influenced by a position j if and only if $j \leq i$, namely a position cannot be influenced by the subsequent positions.

Derive how this causal masking changes equation 1 and 2. After deriving the masked causal attention, using S_i and Z_i as follows,

$$S_i = \sum_{j=1}^{i} \phi\left(K_j\right) V_j^T, \;\; Z_i = \sum_{j=1}^{i} \phi\left(K_j\right), \tag{6}$$

to simplify the causal masking kernel attention and derive the computational complexity of this new causal masking formulation scheme.

4. Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student!
We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

[1] https://www.csie.ntu.edu.tw/~cjlin/talks/kuleuven_svm.pdf
[2] https://people.eecs.berkeley.edu/~brecht/papers/07.rah.rec.nips.pdf

(a) **What sources (if any) did you use as you worked through the homework?**

(b) **If you worked with someone on this homework, who did you work with?**
List names and student ID's. (In case of homework party, you can also just describe the group.)

(c) **Roughly how many total hours did you work on this homework? Write it down here where you'll need to remember it for the self-grade form.**

Contributors:

- Olivia Watkins.
- Sheng Shen.
- Hao Liu.
- Allie Gu.
- Anant Sahai.
- CS182 staff from past semesters.
- Shivam Singhal.
- Kevin Li.
- Bryan Wu.
- Shaojie Bai.
- Angelos Katharopoulos.
- Hao Peng.

