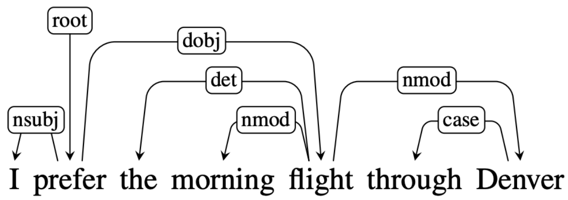
TTIC 31230 Fundamentals of Deep Learning, 2020

Problems For Language Modeling, Translation and Attention.

Problem 1. Transformers as Dependency Parsers. A dependency parse is a labeled directed graph on the words in a sentence. For example,



In this example the edges are labeled with **nsubj**, **dobj**, **det**, **nmod** and **case**. A dependency parse determines a tree with a root node labeled as **root** and with the other nodes labeled with the words of the sentence. This tree structure defines a set of phrases where each phrase consists of words beneath a given node of the tree.

(a) Let k range over the set of possible labels in a dependency parse. There is typically a small fixed set of such labels. If we interpret k has a transformer head, what attention $\alpha(\mathbf{dobj}, \mathbf{prefer}, w)$ over the words w attended from the word **prefer** by the head **dobj** corresponds to the above dependency parse?

(b) A dependency parse rarely has two edges leading from a given word that both have the same label. GTP-3 has 96 heads in each of 96 self-attention layers. It might be reasonable that, with so many heads, each head chould be encouraged to focus its attention on a small number of words (as would be typical in a dependency parse). Define a loss $\mathcal{L}_{\text{focus}}$ that can be combined with the log loss term of the language model such that $\mathcal{L}_{\text{focus}}$ encourages each head to focus on a small number of words. Write the total loss as a weighted sum of the language model loss $\mathcal{L}_{\text{LM}}$ and $\mathcal{L}_{\text{focus}}$. You do not need to define $\mathcal{L}_{\text{LM}}$, just $\mathcal{L}_{\text{focus}}$.

(c) Dependency edges tend to be between nearby words. Repeat part **(b)** but for a loss $\mathcal{L}_{\text{near}}$ which encourages the attention $\alpha[\ell, k, t_1, T_2]$ to be focused near t_1. The loss $\mathcal{L}_{\text{near}}$ should be "robust" in the sense that it has a maximum value that is independent of the length T of the transformer window. This should allow some "outlier" long distance attentions which are needed for coreference.

(d) State the "universalty assumption" under which the "loss shaping" terms of **(b)** and (**c**) above only hurts the langauge modeling performance. Also give a plausibility argument that these terms might help in practice.

Problem 2. Adjusting Temperature for Dimension. For a typical language model the softmax operation defining the probablility $P(w_{t+1} \mid w_1, \ldots, w_t)$ has the form

$$\alpha[\ell, t, w] = \underset{w}{\text{softmax}}\, h[\ell, t, I]e[w, I]$$

We now consider adding a "temperature parameter" β to this softmax.

$$\alpha[\ell, t, w] = \underset{w}{\text{softmax}} \;\; \beta \; h[\ell, t, I]e[w, I] \tag{1}$$

(a) Assume that the components of the vector $h[t, I]$ are independent with zero mean and unit variance. Also assume that that the word vectors have been initialized so that the components of the vector $e[w, I]$ are zero mean and unit variance. What initial value of β gives the result that the inner product $\beta h[t, I]e[w, I]$ has zero mean and unit variance. Explain your answer. (Use I to denote the dimension of the vectors $h[t, I]$ and $w[w, I]$.)

(b) Relate your answer to (a) to the equation used for the self attention $\alpha(k, t_1, t_2)$ computed in the tansformer.

Problem 3. Parameterizing Inner-Product This problem is on transformer self-attention. Modern classification problems tend to use a softmax operation of the form

$$P(y|h) = \underset{y}{\text{softmax}} \;\; h^\top e(y) \tag{2}$$

where h is a vector computed by the neural network and $e(y)$ is a vector embedding for the label y. Many early systems would insert a parameter matrix so that we have

$$P(y|h) = \underset{y}{\text{softmax}} \;\; h^\top W e(y) \tag{3}$$

However, it was generally observed that additional parameterization of the inner product operation does not improve the results. The vector h and the embedding $e(y)$ can be learned to be such that the standard inner product works well. However, the attention softmax of the transformer (3) does not use a naive inner product.

(a) Explain why we cannot replace (3) with a naive inner product of $L[\ell, t_1, J]$ and $L[\ell, t_2, J]$ as in (2).

(b) Rewrite the transformer self-attention equation (2) in the form of (3) where the matrix W in (3) is replaced by by a matrix defined in terms of W^K and W^Q.

Problem 4. Repetition at Low Temperatures. For low temperatures and modest-sized language models we tend to generate infinitely repeating infinite sentences. We can get insight into this phenomenon by considering a trigram model where each word is predicted from the two preceding words using a conditional probability $P_\Phi(w_{t+2}|w_t, w_{t+1})$. We will assume trained word embeddings $e(w)$ for each word w and a neural network predictor of the form

$$P(w_{t+1}|w_t, w_{t+1}) = \underset{w_{t+2}}{\text{softmax}} \, \beta \; h_\Phi(e(w_t), e(w_{t+1}))^\top \; e(w_{t+2})$$

where h_Φ is some arbitray neural network returnnig a quey vector and β is a temperature parameter. We will assume that the model has been trained with β

held fixed at 1 but that we will generate from this trigram model with different values of β. What degenerate behavior are we guaranteed to see if we sample at zero temperature? Explain your answer.

Problem 5. Eliminating the Key Matrix. The self-attention in the transformer is computed by the following equations.

$$
\begin{aligned}
\mathrm{Query}_{\ell+1}[k,t,i] &= W^Q_{\ell+1}[k,i,J]L_\ell[t,J] \\
\mathrm{Key}_{\ell+1}[k,t,i] &= W^K_{\ell+1}[k,i,J]L_\ell[t,J] \\
\alpha_{\ell+1}[k,t_1,t_2] &= \operatorname*{softmax}_{t_2}\left[\frac{1}{\sqrt{I}}\,\mathrm{Query}_{\ell+1}[k,t_1,I]\mathrm{Key}_{\ell+1}[k,t_2,I]\right]
\end{aligned}
$$

Notice that here the shape of W^Q and W^K are both $[K,I,J]$. We typically have $I < J$ which makes the inner product in the last line an inner product of lower dimensional vectors.

(a) Give an equation computing a tensor $\tilde{W}^Q[K,J,J]$ computed from W^Q and W^K such that the attention $\alpha(k,t_1,t_2)$ can be written as

$$\alpha_{\ell+1}(k,t_1,t_2) = \operatorname*{softmax}_{t_2}\left[L_\ell[t_1,J_1]\tilde{W}^Q[k,J_1,J_2]L_\ell[t_2,J_2]\right]$$

For a fixed k we have that $W^Q[k,I,J]$ and $W^K[k,I,J]$ are matrices. We want a matrix $\tilde{W}^Q[k,J,J]$ such that the attention can be written in matrix notation as $h_1^\top \tilde{W}^Q h_2$ where h_1 and h_2 are vectors and $\tilde{W}^Q$ is a matrix. You need write this matrix $\tilde{W}^Q$ in terms of the matrices for W^Q and W^K. But write your final answer in Einstein notation with k as the first index.

(b) Part (a) shows that we can replace the key and query matrix with a single query matrix without any loss of expressive power. If we eliminate the key matrix in this way what is the resulting number of query matrix parameters for a given layer and how does this compare to the number of key-query matrix parameters for a given layer in the original transformer version.