Practice Exam Problems: Deep Learning

Natural Language Processing (JHU 601.465/665)

Prof. Jason Eisner

1. In the English language, almost every sentence contains a verb. But nowadays, it seems that almost every sentence also contains the word `Trump`! Perhaps the grammar of English has evolved recently? Let's call the new dialect "Trumpy English."

   This question is about building a language model that enforces this special property of Trumpy English.

   You have an RNN language model p that was pre-trained on an older corpus of text from the Wall Street Journal (WSJ). So $p(\mathbf{x})$ tends to be high if $\mathbf{x}$ is a probable string of WSJ English, and low otherwise.

   One thing you could do is to train the model further on some sentences of Trumpy English, using stochastic gradient descent.

   (a) Given a new minibatch of 30 sentences $(\mathbf{x}_1, \ldots, \mathbf{x}_{30})$, what function will you try to decrease by taking a step along its gradient?

   (b) When you train in this way, the RNN will learn about Trumpy English, but it may start to "catastrophically forget" what it knows about WSJ English. The longer you train on a small corpus of Trumpy English, the more your procedure reduces ______, but at the cost of greater ______.

   (c) To resist this "catastrophic forgetting," you could make use of a development corpus. How would you do this? And what kind of sentences should be in the development corpus?

2. Here are some of the equations that define the RNN in question 1:

$$p(w_{j+1} \mid w_1, \ldots, w_j) \propto \exp\left(\vec{e}_{j+1} \cdot [1; \vec{h}_j]\right) \tag{1}$$

$$\vec{h}_j = \sigma(V[1; \vec{h}_{j-1}; \vec{w}_j]) \tag{2}$$

All vectors here are column vectors, and the semicolon notation concatenates column vectors into a taller column vector.

$\vec{w}_j$ is the "input embedding" of word type w_j, and $\vec{e}_{j+1}$ is the "output embedding" of word type w_{j+1}; they can be regarded as rows of input and output embedding matrices.

Assume that each $\vec{h}_j$ is a d-dimensional vector with elements $\vec{h}_j[1], \ldots, \vec{h}_j[d]$. That is, $\vec{h}_j \in \mathbb{R}^d$.

(a) What are all of the parameters of this model?

(b) What is the dimensionality of the V matrix?

(c) Suppose you wanted to make a 2-layer stacked RNN. How does this change equations (1)–(2)? Write the new equations.

(d) Let's go back to the old equations (1)–(2). You would like to set some of the parameters of this neural network so that it explicitly keeps track of whether `Trump` has appeared in the sentence yet. You can keep these parameters constant while training the others.

To be specific, for all j, you want $\vec{h}_j[3] \approx 0$ when `Trump` $\in \{w_1, \ldots, w_j\}$, and otherwise $\vec{h}_j[3] \approx 1$.

Let $\vec{t}$ be the embedding of `Trump`. For the sake of this question, assume that the word embeddings are fixed, and that $\vec{t} \cdot \vec{t} > \vec{t} \cdot \vec{w}$ for all words $w \neq$ `Trump`. This means that then `Trump` is more similar to itself than to any other word, if we regard the dot product operator on two words' embeddings as measuring the similarity of those words,

Carefully describe how to set certain entries in V so that $\vec{h}_j[3]$ will behave in the desired way. *Hint:* You will need to discuss one row or one column of V. Say which row or column you will set, and how you will set it.

(e) What other parameters of the RNN should you set in order to ensure that random sentences generated by the RNN have a high probability of containing the word `Trump`, which is the case in Trumpy English? Say which parameters you will set, and how you will set them.

3. In class, we discussed a recursive method for encoding parse trees as vectors.

   (a) First, let's review that method. Suppose you have an NP subtree with encoding $\vec{h}_{\texttt{NP}}$, and a VP subtree with encoding $\vec{h}_{\texttt{VP}}$. You combine them into an S, using a learned matrix called $W_{\texttt{S}\to\texttt{NP VP}}$. What formula do you use for the encoding of that S?

   (b) In class, we allowed $\vec{h}_{\texttt{NP}}$ and $\vec{h}_{\texttt{VP}}$ to have different dimensionality. But for this problem, let's say that all subtrees are encoded as vectors in $\mathbb{R}^d$. Then what are the dimensions of the matrix $W_{\texttt{S}\to\texttt{NP VP}}$?

   (c) This question depends on the previous questions.

   Suppose you dropped the nonlinearity from your formula in question 3a. Interestingly, the encoding of a parse tree could then be described as a simple sum of vectors, where each vector corresponds to a different node in the parse tree. Try to convince yourself of this. Then, for the parse tree `(S (NP Papa) (VP (V ate) (NP (Det the) (N caviar))))`, describe how to get the summand corresponding to `(Det the)` in the representation of the whole tree. Let $\vec{h}_{\text{Det}}$ denote the encoding of `(Det the)`, and write your answer in terms of that. Invent some notation if this helps you explain.