Machine Learning for Graphs and Sequential Data Exercise Sheet 08

Neural Network Approaches for Sequential Data

Problem 1: Word2vec defines a mapping from a single word to a single fixed vector. Explain and provide an example why this will not be expressive enough regarding homographs (i.e., words with the same spelling but having more than one meaning). Propose an alternative solution.

Word2vec assigns the same vector representation to all words with the same spelling. Hence, it will fail to disambiguate the words that have the same spelling but different meaning, e.g. *duck* – an animal and *duck* – to lower head.

One possible solution is to use RNNs, e.g. LSTMs, to get contextually aware embeddings that will be different for different contexts. Given a sentence, each word will get a hidden state as an embedding. The training task is to predict the next word based on all the previous. Another example is attention based models, e.g., BERT, GPT-3 etc.

Problem 2: Given a previous hidden state $\boldsymbol{h}^{(t-1)} \in \mathbb{R}^D$ and a current input $\boldsymbol{x}^{(t)} \in \mathbb{R}^N$, the recurrent neural network equations to update the hidden state and produce the output are:

$$\begin{aligned}
\boldsymbol{z}^{(t)} &= \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)} \\
\boldsymbol{h}^{(t)} &= \tanh\left(\boldsymbol{z}^{(t)}\right) \\
\boldsymbol{o}^{(t)} &= \boldsymbol{V}\boldsymbol{h}^{(t)} \\
\hat{\boldsymbol{y}}^{(t)} &= \operatorname{softmax}\left(\boldsymbol{o}^{(t)}\right)
\end{aligned}$$

where parameters $\boldsymbol{W} \in \mathbb{R}^{D\times D}$, $\boldsymbol{U} \in \mathbb{R}^{D\times N}$ and $\boldsymbol{V} \in \mathbb{R}^{M\times D}$ are shared at every step.

To train an RNN we need gradients of loss w.r.t. the parameters: $\partial L/\partial \boldsymbol{W}$, $\partial L/\partial \boldsymbol{U}$ and $\partial L/\partial \boldsymbol{V}$. Your task is to arrive at the equations given on slide 17 in the lecture.

Use the fact that $\partial L/\partial \boldsymbol{o}^{(t)} = \hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)}$, where $\boldsymbol{y}^{(t)}$ is the true output.

Hint: Since parameters are shared, the total gradient is the sum of the contributions over all the steps. Because of that, it might be easier to introduce copies of parameters, e.g. $\boldsymbol{W}^{(t)}$ – a copy of $\boldsymbol{W}$ at step t, calculate $\partial L/\partial \boldsymbol{W}^{(t)}$ and sum over all t.

We want to show the following formula:

$$\frac{\partial L}{\partial \boldsymbol{V}} = \sum_t \left(\hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)}\right)\left(\boldsymbol{h}^{(t)}\right)^T \tag{1}$$

$$\frac{\partial L}{\partial \boldsymbol{W}} = \sum_t \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right)\frac{\partial L}{\partial \boldsymbol{h}^{(t)}}\left(\boldsymbol{h}^{(t-1)}\right)^T \tag{2}$$

$$\frac{\partial L}{\partial \boldsymbol{U}} = \sum_t \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right)\frac{\partial L}{\partial \boldsymbol{h}^{(t)}}\left(\boldsymbol{x}^{(t)}\right)^T \tag{3}$$

We start with (1). We can unroll the gradient into all the contributions across time, and then apply the chain rule:

$$\frac{\partial L}{\partial \boldsymbol{V}} = \sum_t \frac{\partial L^{(t)}}{\partial \boldsymbol{V}} = \sum_t \frac{\partial L^{(t)}}{\partial \boldsymbol{o}^{(t)}} \frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{V}}$$

Each element of this product can be written as:

$$\frac{\partial L}{\partial \boldsymbol{o}^{(t)}} = \hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)},$$
$$\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{V}} = \left(\boldsymbol{h}^{(t)}\right)^T,$$

where we slightly abuse the notation in the second term. The value $\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{V}}$ is actually a tensor, but most of the entries are zero. See also: `https://web.stanford.edu/class/cs224n/readings/gradient-notes.pdf`.

We now consider (2). Again, expand the different time contributions and apply the chain rule:

$$\frac{\partial L}{\partial \boldsymbol{W}} = \sum_t \frac{\partial L}{\partial \boldsymbol{W}^{(t)}} = \sum_t \frac{\partial L}{\partial \boldsymbol{h}^{(t)}} \frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{z}^{(t)}} \frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{W}^{(t)}}$$

The Jacobian $\frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{z}^{(t)}}$ is a diagonal matrix since $h_i^{(t)}$ does not depend on $z_j^{(t)}$ if $i \neq j$. The tanh operation is applied element-wise. Thus, $\frac{\partial \boldsymbol{h}_i^{(t)}}{\partial \boldsymbol{z}_i^{(t)}} = 1 - \tanh(\boldsymbol{z}_i^{(t)})^2 = 1 - (\boldsymbol{h}_i^{(t)})^2$. Hence, we have:

$$\frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{z}^{(t)}} = \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right),$$

and also:

$$\frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{W}^{(t)}} = \left(\boldsymbol{h}^{(t-1)}\right)^T.$$

For (3), computations are similar except that we use $\frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{U}^{(t)}} \left(\boldsymbol{x}^{(t-1)}\right)^T$ instead of $\frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{W}^{(t)}} = \left(\boldsymbol{h}^{(t-1)}\right)^T$. These shows the formula at slide 17.

Additionally, note that we could compute the derivative $\frac{\partial L}{\partial \boldsymbol{h}^{(t)}}$ recursively from $\frac{\partial L}{\partial \boldsymbol{h}^{(t+1)}}$:

$$\begin{aligned}\frac{\partial L}{\partial \boldsymbol{h}^{(t)}} &= \sum_{s=t}^{T} \frac{\partial L^{(s)}}{\partial \boldsymbol{h}^{(t)}} \\ &= \frac{\partial L^{(t)}}{\partial \boldsymbol{h}^{(t)}} + \sum_{s=t+1}^{T} \frac{\partial L^{(s)}}{\partial \boldsymbol{h}^{(t)}}\end{aligned}$$

We can compute the derivatives:

$$\frac{\partial L^{(t)}}{\partial \boldsymbol{h}^{(t)}} = \frac{\partial L^{(t)}}{\partial \boldsymbol{o}^{(t)}} \frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{h}^{(t)}} = \boldsymbol{V}^T \left(\hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)} \right)$$

$$\sum_{s=t+1}^{T} \frac{\partial L^{(s)}}{\partial \boldsymbol{h}^{(t)}} = \sum_{s=t+1}^{T} \frac{\partial L^{(s)}}{\partial \boldsymbol{h}^{(t+1)}} \frac{\partial \boldsymbol{h}^{(t+1)}}{\partial \boldsymbol{z}^{(t+1)}} \frac{\partial \boldsymbol{z}^{(t+1)}}{\partial \boldsymbol{h}^{(t)}} = \sum_{s=t+1}^{T} \boldsymbol{W}^T \mathrm{diag} \left(1 - \left(\boldsymbol{h}^{(t+1)} \right)^2 \right) \frac{\partial L^{(s)}}{\partial \boldsymbol{h}^{(t+1)}}$$

As explained in slide 18, we see that the gradient depends on future times.

Problem 3: What do you need to change in the equations that you got in the previous exercise if the output $\boldsymbol{o}^{(t)}$ is used as an input to another neural network?

Instead of using $\left(\hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)} \right)$ we would need to calculate $\partial L / \partial \boldsymbol{o}^{(t)}$ depending on the subsequent layers and loss. Everything else remains the same.