cs224n

(arthurg)

June 3, 2020

Question 1.

When using convolutions on character level language models, the convolutions (of size k) are able to operate on words of arbitrary length. However, the output of this convolution will be a vector of size $len - k$.

Question 2.

The minimum w_{word} is 1. After padding the *sow* and *eow* tokens, the minimum length for x_{padded} would thus be $R^{1+2} = R^3$

To ensure we apply at least one full convolution, we need to padd x_{padded} to size 5. Then, we need padding of 1. $x_{reshaped} \in R^{e_{char}x1+2+(2*1)} = R^{e_{char}x5}$

Question 3.

It's useful for the extremes of x_{gate} to set $x_{highway}$ be either fully x_{proj} or fully x_{conv_out} because it allows certain character embeddings to optionally pass through another layer.

It's probably a better idea to set the bias to positive. This will ensure $x_{gate} -> 1$. If $x_{gate} = 0$, we will have no gradient on x_{proj} which makes the layer useless.

Question 4.

- Parallizes better on GPUs

- Multi-headed attention might improve translation accuracy when trying to do things like verb - noun agreements

Question 5.

I tested my Highway network by:

1. Running a batch of x_{conv_out} to check the dimensions are correct

2. Set the weights of w, and send in an x_{conv_out} vector (of size 5). Manually do the matrix math to check the output matches. In these test cases, I made sure to find cases where the sigmoid is 0, sigmoid is 1, and where the relu is 0.

I'm confident that these two tests will cover the edge cases. In general, since I used pre-defined components in PyTorch, most of the edge cases are handled for me.

Question 6.

I tested my CNN network by:

1. Running the CNN on a batch of x_{emb} to check the dimensions are correct

2. Set the weights of w, and send in an x_{emb}. Through hand computing the convlustion, and pooling operation, I manually network was acting properly.

3. Set the weights of w, and send in an x_{emb}. Through hand computing the convlustion, and RELU operation, I manually network was acting properly.

I'm confident that these three tests will cover the edge cases. In general, since I used pre-defined components in PyTorch, most of the edge cases are handled for me.