Homework 2 report

Dnyanesh Balasaheb Marne

Git : https://github.com/DnyaneshMarne/Deep_Learning

In this homework we are supposed to create captions for the video. The data provided has the videos features which we can directly feed to the encoder instead of converting the video into tensor first. The RNN model uses 'GRU' encoder decoder along with Attention layer.

First we take the training data JSON label data and create word count, and index the words to create the encode sentences in terms of numbers. Then we combine these encoded sentences to the respective video features provides as ground truth. I first removed all symbols and made the words lowercase for consistency. Also doing the same for the sentences. We are going to filter out words with frequency less than 3. I even tried filtering out words like 'a', 'an', 'the' to see how it behaves I thought it would give us captions describing with verbs more than just printing 'a' 'a' 'a' but it did not work that way it actually gave BLEU score of 0 and when I saw the predicted captions it was filled with just 'something' as all the words.

I tried using LSTM but continuously faced memory issues on CUDA gpu V100 on palmetto. Then I used GRU to see the behaviour and it worked so kept it. While searching for the reason came across a document mentioning that GRU uses less memory than LSTM and provides better result for small dataset which made sense when I had memory allocation issues. The data to encoder is fed in format of (tensor(video feature),tensor(encoded sentence)) the encoder output and hidden states are then fed to decoder which also uses the attention layer to focus on certain part of the data while predicting to get the notion of persistence.

Without the Attention layer the output lacked variety in caption generation and BLEU score was around 50 percent. This changed after the additional attention layer. Another factor which improved the score was schedule sampling by feeding either the predicted word to the next time step's input or ground truth based on 'randomness' which solves the issue of exposure bias. By adding the prediction data to the feed we are not only relying on the training data which helps to avoid the exposure bias.

The hyper parameters for the training are as follows :

Learning rate = 0.0001

Optimizer = Adam

Epoch = 50

Loss function = Cross Entropy

When I ran the model with 0.001 learning rate and 70 epochs I got around 60 percent of BLEU score.

```
(pytorch_env) [dmarne@node0227 deep_learning]$ sh hw2_seq2seq.sh /MLDS_hw2_1_data/testing_data/feat out.txt
/home/dmarne/deep_learning/MLDS_hw2_1_data/testing_data/feat
Average bleu score is 0.6058093525877642
```

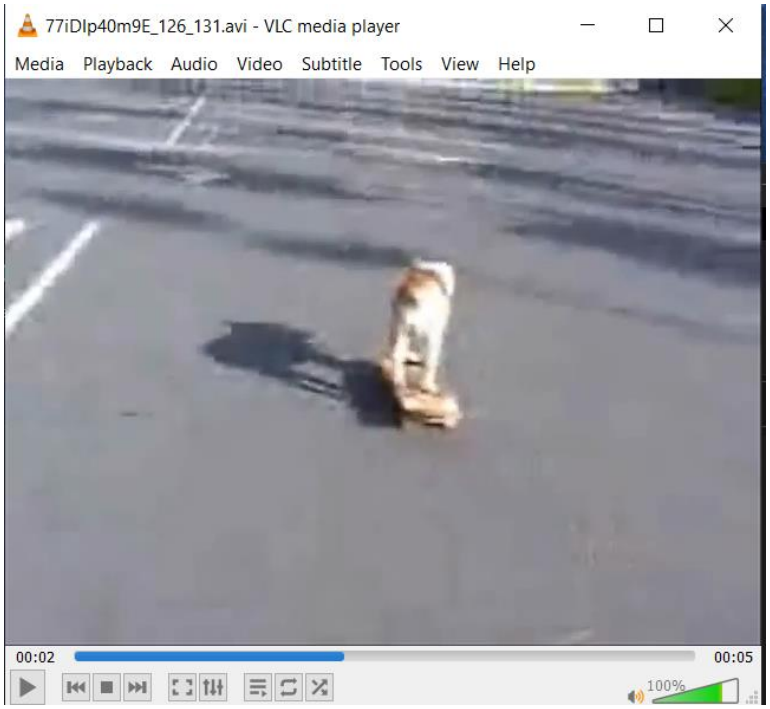
Then after trying different parameters got the BLEU of around 69%

```
(pytorch_env) [dmarne@node0227 deep_learning]$ sh hw2_seq2seq.sh /MLDS_hw2_1_data/testing_data/feat ou1t.txt
/home/dmarne/deep_learning/MLDS_hw2_1_data/testing_data/feat
Average bleu score is 0.6697128010263839
```

```
(pytorch_env) [dmarne@node0227 deep_learning]$ sh hw2_seq2seq.sh /MLDS_hw2_1_data/testing_data/feat ou1t.txt
/home/dmarne/deep_learning/MLDS_hw2_1_data/testing_data/feat
Average bleu score is 0.6975722926960934
```

Even though the BLEU score increased the output captions are not that refined model does relate the things in the video in some way like –

For this testing video it predicts ‘a man is playing guitar’ we can see it’s not a guitar but a drum but it kind of relates to a musical instrument partially because maybe the training data has got of guitar videos which is a musical instrument and video is recorded in a same environment like above.



For this video model predicts – ‘a dog is running in the’

Well it’s not running but clearly it’s a dog so even when the model does not get everything right there is some correlation to what it predicts and what is in the video so BLEU is increased.