UDACITY

Logout

PROJECT

Translation From One Language to Another Language

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW | CODE REVIEW | NOTES

Requires Changes

SHARE YOUR ACCOMPLISHMENT

3 SPECIFICATIONS REQUIRE CHANGES

Kudos ! I think you've done a perfect job of implementing a Sequence to Sequence model perfectly. It's very clear that you have a good understanding of the basics. Keep improving and keep learning. As it appears, you have grasped the concept of Sequence to Sequence Models (LSTMs & RNNs) very well, here's a very popular blog that might help you in visually understanding further details.

There were minor errors which I am sure you can rectify in your next submission.

Advanced tips for improving net results

- Try and use deeper architectures, which have general tendency to blow up or vanish the gradients - so there's a net architecture known as Residual Nets, used to circumnavigate the issues with deeper architectures
- Try using more fully connected layers or Bi-Directional LSTMs to make the predictions even better, as the ordering of sentence is always not dependent on previous words, can also be slightly dependent on the words coming later (very slightly). BLSTMs can help then
- Try and use more sophisticated methods like lemmatisation and stemming to create a more pruned vocabulary. Have a look at the NLTK library to understand more operations

If you are keen on learning a bit more into what Natural Language Scientists use regularly in their nets. Try reading up a bit more on

- Word2Vec Algorithm
- Glove Algorithm
- Sequence2Sequence tutorial

Keep up the good work you've done to get you this far. Very impressive !

Required Files and Tests

✓ **The project submission contains the project notebook, called "dlnd_language_translation.ipynb".**

✓ **All the unit tests in project have passed.**

Preprocessing

✓ **The function `text_to_ids` is implemented correctly.**

The pre-processing step in `text_to_ids` seems to be working perfectly fine. Good implementation in especially carefully appending `<EOS>` after end of sentence for target text. Nice job !

Neural Network

✓ **The function `model_inputs` is implemented correctly.**

Good job in implementing the placeholders correctly. Also, the fact that you used naming of tensors, where it was not mandated, that's a good practice !

✓ **The function `process_decoding_input` is implemented correctly.**

The encoding functions works perfectly as intended. Good job there !

✓ **The function `encoding_layer` is implemented correctly.**

Rate this review

Good job implementing the LSTM layer perfectly.

I would strongly recommend usage of dropout layers as they help in prevent overfitting, which seems to be the case here
Example usage would be:
```
drop = tf.contrib.rnn.DropoutWrapper(lstm, output_keep_prob=keep_prob)
```

✓ **The function `decoding_layer_train` is implemented correctly.**

✓ **The function `decoding_layer_infer` is implemented correctly.**

✓ **The function `decoding_layer` is implemented correctly.**

This was again one of the relatively challenging part of the project. Good to see a perfect implementation of shared variable instance. Normally a shared variable helps in a lot of ways, namely

- Efficient storage in memory, without needing to store separate variables for each instance
- Passing them as a scope, saves efforts whenever you change the code later (say for initialising the variable instance)

Good job !

✓ **The function `seq2seq_model` is implemented correctly.**

Your implementation seems to be fine - though you could apply embedding directly for the decoding layer directly as you did for encoding layers like
```
tf.contrib.layers.embed_sequence(dec_input, target_vocab_size,dec_embedding_size)
```

Neural Network Training

🔁 **The parameters are set to reasonable numbers.**

I think there's a clear chance of overfitting as you're using too many parameters. I think the optimal mix of parameters should be like :

- Lower number of layers - as more the number of layers, more parameters, extremely high chances of overfitting
- Lower `keep_probability` as those are the probabilities of layers not being dropped, in this case it's irrelevant as you've not used dropout at all
- Learning Rate and epoch combination is fine normally, in this case (if you don't apply the suggested changes) more no of epochs could be needed

🔁 **The project should end with a validation and test accuracy that is at least 90.00%**

Both validation and test accuracy are lower than 90 %

Language Translation

✓ **The function `sentence_to_seq` is implemented correctly.**

🔁 **The project gets majority of the translation correctly. The translation doesn't have to be perfect.**

Doesn't look like a fairly decent translation when I translated using Google Translate ! Although I really applaud your efforts on putting so much efforts and getting a squence2sequence model running !

RESUBMIT PROJECT

DOWNLOAD PROJECT

Rate this review