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

1) Why we have applied dropout for this project and not for tv_script_generation?
A number of students applied dropout in tv_script_generation. Whenever you wish you prevent you model from overfitting, Dropout can be applied.

2) I still can't understand tf.nn.dropout vs tf.contrib.layers.dropout.
Contrib layers are meant to be sugar coated versions of `nn` class. Both serving the same purpose.

3) Why the NN was unable to learn when I applied tf.contrib.rnn.DropoutWrapper at the encoding layer?
`tf.contrib.rnn.DropoutWrapper` is perfectly correct way to use in embedding layer. I doubt having `keep_prob` around 0.9 would help in better learning.

Kudos ! I think you've done a perfect job of implementing a seq2seq model for Machine Translation. It's very clear that you have a good understanding of the basics.
There were some minor errors which I am sure you can improve for your next submission !

Looking forward to your next submission. :)

Required Files and Tests

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

The `iPythonNB` and helper files are included.

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

Great work. Unit testing is one of the most reliable methods to ensure that your code is free from all bugs without getting confused with the interactions with all the other code. If you are interested, you can read up more and I hope that you will continue to use unit testing in every module that you write to keep it clean and speed up your development.

But always keep in mind, that unit tests cannot catch every issue in the code. So your code could have bugs even though unit tests pass.

Preprocessing

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

Clean and concise.
Both source and target are correct mapped to their respective mappings.
Good job using `EOS_ID` only at end of target which will help decoder **stop** translating.

Neural Network

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

Correct.
Placeholders is the building block in computation graph of any neural net (especially in **tensorflow**).

Often I find students confused between `tf.Variable` and `tf.placeholder`. This answer gives correct usecase for both.

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

Good job concatenating the `GO_ID` to the beginning of each batch. ✌
Effectively manipulating elements computational graph is one of the most important skills to acquire building deep models. And you are handling `tf` tensors amazingly.

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

Your Encoder RNN layer is implemented perfectly.
`tf.nn.dynamic_rnn` is perfect for variable length time sequences as in our case of translations models. This is more or less the default choice when building seq-to-seq RNN architectures.

Further, there are a lot of other options when choosing `RNNCells` which you must be aware of.

Library of RNNCells

- `BasicRNNCell`
- `BasicLSTMCell`
- `GRUCell`
- `LSTMCell`
- `LayerNormBasicLSTMCell`
- `CoupledInputForgetGateLSTMCell`
- `TimeFreqLSTMCell`
- `GridLSTMCell`
- `New: NASCell`
- `MultiRNNCell`
- `DropoutWrapper`
- `DeviceWrapper`
- `ResidualWrapper`
- `AttentionCellWrapper`
- `CompiledWrapper`

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

Nice work! But a small mistake here.

Your `train_logits` is the output layer of your model. And you apply a dropout layer to it. Dropouts aren't applied to the output layers. If you do wish to apply a dropout, you can apply it to `train_pred` instead.

This is also probably why your training loss isn't going down beyond a point. In fact, it would have gone lower had it not been for a high keep_probability value you set which basically reduces the effects of dropout too.

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

Good work!

An important point -

This is the inference/prediction function where, as you may already know, we don't apply dropout since dropout is only meant for training and adding it to prediction makes us lose some connections. If you check out the code where the graph is built and the training happens, the output of this particular function is run in the session with keep_prob as 1.0. Which is good. But it also means that since we have explicitly defined this function for inference/prediction, adding a dropout here is not really needed.

I just want to be sure you are aware of the above :)

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

Defining scope using `tf.variable_scope` to share variables between training and inference was very important for this part.

Pro Tip: When building such systems at scale, the hidden state connecting encoder and decoder is often the bottleneck for computation. **Attention** is an important concept which comes at rescue. This talk by a Google developer and this awesome article on seq2seq and attention is perfect to understand how it works.

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

You placed all the lego blocks correctly except for one small piece.

Pro Tip: Google recently released a general-purpose encoder-decoder framework for Tensorflow that can be used for Machine Translation, Text Summarization, Conversational Modeling, Image Captioning, and more. Check out the blog post and Git repo.

Neural Network Training

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

You might have to fine-tune some parameters based on the above changes, but you still did a good job with your tuning!

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

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.**

Once you correct the mistake and tune your hyperparams, much better translation will be produced.

RESUBMIT PROJECT

DOWNLOAD PROJECT

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH