UDACITY Logout

‹ Return to Classroom DISCUSS ON STUDENT HUB

Machine Translation

REVIEW | CODE REVIEW | HISTORY

Meets Specifications

Great job! 👍

You seem to have understood theoretically and in practice (coding) how RNNs work for this kind of task.

A Machine Translation tutorial from NVIDIA - Introduction to Neural Machine Translation with GPUs.
Another cool tutorial - Language Translation with Deep Learning and the Magic of Sequences.

Hope you enjoy those.
Keep up with the good work! 😃

PS: For almost all of your models - the training validation loss shows up as - nan; you should refer to ***this knowledge question*** - so as to how to rectify it.

Submitted Files

The following files have been submitted: `helper.py`, `machine_translation.ipynb`, `machine_translation.html`

This submission contains all of the three files:

- `helper.py`
- `machine_translation.ipynb`
- `machine_translation.html`

Preprocess

The function `tokenize` returns tokenized input and the tokenized class.

The function `tokenize` is correctly implemented and returns the tokenized input and the tokenized class.

The function `pad` returns padded input to the correct length.

The function `pad` is correctly implemented and returns padded input to the correct length.

Models

The function `simple_model` builds a basic RNN model.

Good job using the `GRU` instead of `SimpleRNN` in implementing the `simple_model`.

Suggestion: Here are a few suggestions for you to ponder upon:

- Try using different `RNN` units like `LSTM` and compare it with `GRU`.
- You should try evaluating different activation functions like `tanh` and `relu`.

The function `embed_model` builds a RNN model using word embedding.

The function `embed_model` is correctly implemented and builds a RNN model using word embedding.

The Embedding RNN is trained on the dataset. A prediction using the model on the training dataset is printed in the notebook.

The function `bd_model` builds a bidirectional RNN model.

The function `bd_model` is correctly implemented and builds a bidirectional RNN model.

The Bidirectional RNN is trained on the dataset. A prediction using the model on the training dataset is printed in the notebook.

The function `model_final` builds and trains a model that incorporates embedding, and bidirectional RNN using the dataset.

You did a great job building the `model_final` using all of the building blocks from previous models.

Prediction

The final model correctly predicts both sentences.

You correctly trained the model in the dataset and obtained correct predictions for both the sentences.

DOWNLOAD PROJECT

RETURN TO PATH