

Return to "Artificial Intelligence Nanodegree and Specializations" in the classroom

DNN Speech Recognizer

| REVIEW |
|-------------|
| CODE REVIEW |
| HISTORY |

Meets Specifications

Dear Excellent Student,

Congratulations on meeting all the specifications for this project.

The hard work demonstrated in this submission is very commendable and shows a good understanding of the concepts from the lessons. Keep up the hard work and no obstacle will be above you. Have a great day celebrating your success and good luck in your future endeavours! (U)

Pro Tips

Here are some resources that can help you learn further.

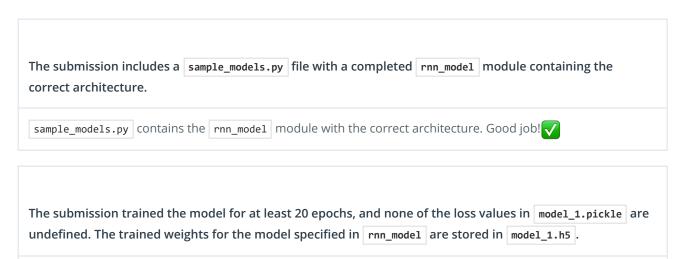
- Deep learning: from speech recognition to language and multimodal processing
- Dense LSTMs for Speech Recognition
- Machine Learning is Fun Part 6: How to do Speech Recognition with Deep Learning
- Deep neural network training for whispered speech recognition using small databases and generative model sampling

STEP 2: Model 0: RNN

The submission trained the model for at least 20 epochs, and none of the loss values in model_0.pickle are

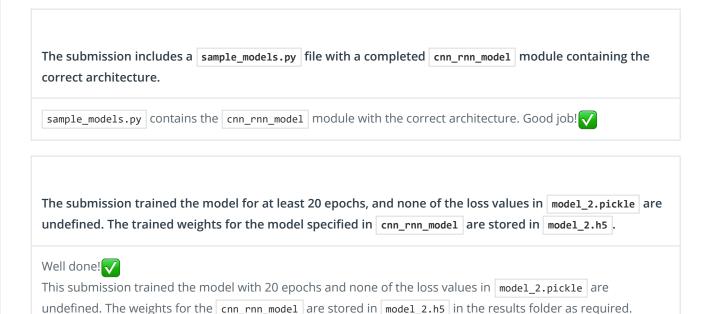
Well done! Well weights for the model with 20 epochs and none of the loss values in model_0.pickle are undefined. The weights for the simple_rnn_model are stored in model_0.h5 in the results folder as required.

STEP 2: Model 1: RNN + TimeDistributed Dense



Well done! This submission trained the model with 20 epochs and none of the loss values in model_1.pickle are undefined. The weights for the rnn_model are stored in model_1.h5 in the results folder as required.

STEP 2: Model 2: CNN + RNN + TimeDistributed Dense



STEP 2: Model 3: Deeper RNN + TimeDistributed Dense

9/12/2019 Udacity Reviews

The submission includes a sample_models.py file with a completed deep_rnn_model module containing the correct architecture.

sample_models.py contains the deep_rnn_model module with the correct architecture. Good job!

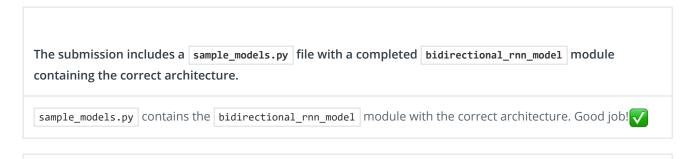
The submission trained the model for at least 20 epochs, and none of the loss values in model_3.pickle are undefined. The trained weights for the model specified in deep_rnn_model are stored in model_3.h5.

Well done!

Well done!

This submission trained the model with 20 epochs and none of the loss values in model_3.pickle are undefined. The weights for the deep_rnn_model are stored in model_3.h5 in the results folder as required.

STEP 2: Model 4: Bidirectional RNN + TimeDistributed Dense



The submission trained the model for at least 20 epochs, and none of the loss values in model_4.pickle are undefined. The trained weights for the model specified in bidirectional_rnn_model are stored in model_4.h5.

Well done!

This submission trained the model with 20 epochs and none of the loss values in model_4.pickle are undefined. The weights for the bidirectional_rnn_model are stored in model_4.h5 in the results folder as required.

STEP 2: Compare the Models

The submission includes a detailed analysis of why different models might perform better than others.

Good job comparing the trained models, take particular observation of the trend of change in validation loss that indicates whether the model overfits or not. It is also great to see that you have a good understanding of the models and recognize the fact that the deeper one (3) is able to detect more complex patterns in the data, giving much better performance than the other models. Keep up the good work!

9/12/2019 Udacity Reviews

STEP 2: Final Model

The submission trained the model for at least 20 epochs, and none of the loss values in model_end.pickle are undefined. The trained weights for the model specified in final_model are stored in model_end.h5.

Well done!

This submission trained the model with 20 epochs and none of the loss values in <code>model_end.pickle</code> are undefined. The weights for the <code>final_model</code> are stored in <code>model_end.h5</code> in the results folder as required.

The submission includes a sample_models.py file with a completed final_model module containing a final architecture that is not identical to any of the previous architectures.

sample_models.py | contains the | final_model | module with a CNN + Bidirectional RNN + TimeDistributed Dense architecture. Good job building this new architecture from components of the trained models. Nicely done!

The submission includes a detailed description of how the final model architecture was designed.

Excellent work here, presenting a detailed account of the final model architecture in your answer to question 2.

■ DOWNLOAD PROJECT

RETURN TO PATH

Rate this review