**Data Science Task - NLP with Tensorflow**

**Requirements**

1. Pick a text (NLP) dataset or image/text (CNN/NLP) dataset of your choice from here: <https://github.com/tensorflow/datasets>

Movie Review data: "[http://ai.stanford.edu/~amaas/data/sentiment/aclImdb\_v1.tar.gz"](http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz%22).

50k reviews in total, including 25k positive reviews and 25k negative reviews

1. Pick one of the pretrained models to do a transfer learning on your dataset: <https://tfhub.dev/>

I used the pretrained Wikipedia Word2Vec model (Wiki-words-250) to conduct sentiment analysis on the movie reviews.

For comparison and for my personal interest, I also used pretrained GloVe (glove-6B-200d) to build another model

1. Describe the purpose of the model, show the model architecture and performance (two metrics of your choice)

The purpose of the model is to use audience reviews to predict their sentiments/preference on the movies.

From the pretrained Wikipedia Word2Vec model, we pulled in the word vectors for the vocabulary used in audience reviews, and used them to train the model.

Considering the data have very balanced target values, I used F1 score and AUC-ROC to evaluate the model performance. Please see below.

* F1 score is 0.863 for training, and 0.863 for validation, when epoch=12
* AUC-ROC is 0.941 for training, and 0.939 for validation, when epoch=16

The architecture of the model using the pre-trained Wikipedia Word2Vec is shown below:

Table

Description automatically generated

1. Please choose a problem directly related to your work experience

Sentiment Analysis

1. Now, build your own model in tensorflow from scratch to solve the same problem based on your work experience, show the model architecture and performance (two same metrics used from step 3) – note that a model from scratch should have significantly poorer model performance due to time/computing resource constraints, and limited training dataset

The model with custom embedding was built. Please see its performance below.

* F1 score is 0.886 for training, and 0.878 for validation, when epoch=2
* AUC-ROC is 0.949 for training, and 0.947 for validation, when epoch=2

The model architecture:

Table

Description automatically generated

1. I am more interested in how you would design the model architect and why, not the model performance

Embedding layer is to custom-produce embedding vectors and pull in the pretrained word vectors.

Bidirectional LSTM cell is to train the data by leveraging the context of each word (i.e. the sequence of words).

Dropout is used to regularize the model complexity and curb overfitting

The additional two hidden layers of Dense is to further train the data and improve the model fit overall.

1. Offer your opinions, in a few bullet points or short sentences, steps you can take (you do not have to build the model) to improve your own model to get to a closer power as the pretrained model

The custom-embedding model gave the performance as good as two pre-trained-embedding models (wiki word2vec, GloVe).

The custom-embedding model run much slower than the pre-trained-embedding models. Not a surprise…it has to run a vast shallow neural net to produce word vectors.

One thing I wanted to point out is that I didn’t get a chance to do hyper-parameter tuning due to my time constraint for this take-home project. The only hyper-parameter I checked is the number of epochs. Ideally, if I have enough time, I would like to set up grid search with cross validation to tune some other hyper-parameters below (arranged in the priority order).

* Learning rate
* Type of Optimizer
* Momentum
* Number of neutrons in each layer
* Dropout rate
* Activation function
* Kernel initializer

I will use GridSearchCV and KerasClassifier for hyper-parameter tuning.

The hyper-parameter tuning will improve the performance of all the three models, but I am not sure how much the incremental we can get.