Pracschool: AI Intern Assessment Assignment

Dec-2017

In case of any doubt/clarification contact : Dr. Chitta Ranjan (cran2367@gmail.com)

Last date of Submission: 8th December 2017 till 11:59pm

Where to submit the Assignment: cran2367@gmail.com

Problem

Suppose you are tasked to develop models on a video data. A video data can be broken into a sequence of images.

Deep Learning methods have proven to work best for images and sequences. In this assignment we will work on Deep Learning methods for them.

Pre-requisite

Installed

a. Jupyter notebook
b. Keras with tensorflow backend

Can refer to https://www.pyimagesearch.com/2016/11/14/installing-keras-with-tensorflow-backend/

Installation is known to be easier in Mac or Linux. A Windows system maybe difficult to work on.

Tasks

1. MNIST digit (image) classifier

MNIST image classification is a popular problem. The MNIST dataset contains handwritten digits. It has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

Deep Learning techniques have proven to work the best on this data. The following link implements a Deep Learning model on the MNIST data using Keras.

https://github.com/wxs/keras-mnist-tutorial/blob/master/MNIST%20in%20Keras.ipynb

Your task:

a. Follow the steps in the following link
b. Understand each step in detail.
c. Does any change in layer, activations or any other training settings improve the test accuracy?

We will assess your understanding of the model and its components, like layer and activations.

2. Sequence-to-sequence autoencoder for classifier

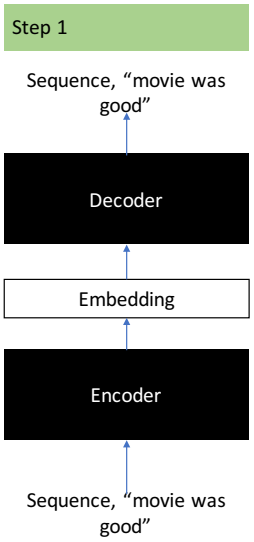
[2.1] https://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/
[2.2] https://chunml.github.io/ChunML.github.io/project/Sequence-To-Sequence/
[2.3] https://blog.keras.io/building-autoencoders-in-keras.html
[2.4] https://keras.io/getting-started/faq/#how-can-i-obtain-the-output-of-an-intermediate-layer

IMDB movie review sentiment classification is another popular problem. The data contains 25k highly-polar (good or bad) movie reviews for training and testing. The objective is to predict whether a given movie review has a positive or negative sentiment. [2.1] presents one solution to this problem using Deep Learning.

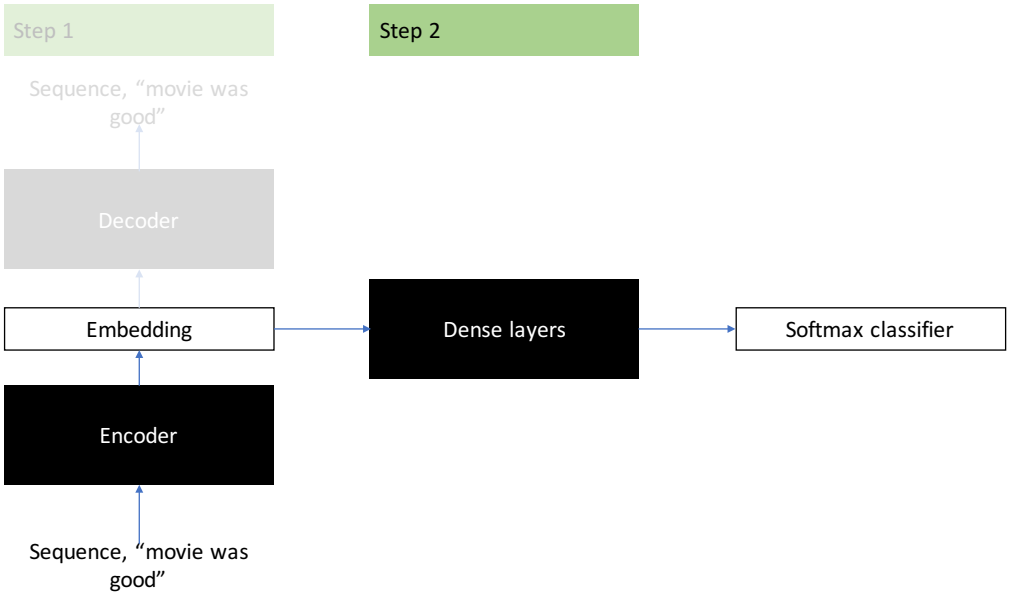
In [2.1]'s solution, word embeddings are used on the reviews. These embeddings are like word2vec, that encodes the words as real-valued vectors in a high dimensional space, where the similarity between words in terms of meaning translates to closeness in the vector space.

We will solve this problem differently. We will treat the review strings as sequence of words, and build an autoencoder on it. [2.2] and links therein provide some examples of modeling sentences as sequence of words.

Each review will be a sequence. And we will build an autoencoder in Step 1. You may use the sequence-to-sequence autoencoder codes given in [2.3]. Note that building a sequence-to-sequence encoder as shown in [2.2] can be more difficult than [2.3]. [2.3] is advisable but we will leave it on the candidate.



In Step 2, we will fetch the output of the encoder, which is an embedding, and build a classifier using it. To fetch the embedding, you may use the instruction in [2.4].



After fetching the intermediate layer from the encoder, adding Dense layers will be similar to the one in [2.1] that builds Dense layers on top of the word embeddings.
Reaching very high classification accuracy is not the priority in this task. The objective is to be able to extract features from the text sequence in an unsupervised manner and then use it for building a classifier.

Your task:

a. Build a sequence-to-sequence autoencoder.

b. Be able to fetch the encoder output (an intermediate layer).
c. Build a classifier using Dense layers on top of the embedding fetched in (b).

This task will assess your ability to implement models and its components on your own.