ASSIGNMENT #5 – NEURAL NETWORKS

Context

This assignment is an opportunity to demonstrate your knowledge and practice solving problems about attention. This assignment contains an implementation component (i.e. you will be asked to write code to solve one or more problems) and an experiment component (i.e. you will be asked to experiment with your code and report results).

Logistics

Assignment due date: April 10, 2023

Assignments are to be submitted electronically through Brightspace. It is your responsibility to ensure that your assignment is submitted properly and that all the files for the assignment are included. Copying of assignments is NOT allowed. High-level discussion of assignment work with others is acceptable, but each individual or small group is expected to do the work themselves.

Programming language: Python 3

For all parts of the implementation, you may use the Python Standard Library (https://docs.python.org/3/library/) and the following packages (and any packages they depend on): Keras, NumPy, Pandas, scikit-learn, SciPy, TensorFlow. Unless explicitly indicated below or explicitly approved by the instructor, you may not use any additional packages.

You must implement your code yourself; do not copy-and-paste code from other sources. Please ensure your implementation follows the specifications provided; it may be tested on several different test cases for correctness. Please make sure your code is readable; it may also be manually assessed for correctness. You do not need to prove correctness of your implementation.

You must submit your implementation as a single file named “assignment5.py” with classes functions as described below (you may have other variables/functions/classes in your file). Attached is skeleton code indicating the format your implementation should take.

You must submit your report on experimental results as a single file named “assignment5.pdf”.

Implementation Component

For the implementation, we will use the dataset called "Activity recognition with healthy older people using a batteryless wearable sensor". The dataset consists of chest-mounted accelerometer readings from participants performing seven different activities: (1) sitting on a bed, (2) sitting on a chair, (3) lying on a bed, and (4) ambulating (i.e. standing up or walking).

This dataset is publicly available from the UCI Machine Learning Repository. You may download it and find more details on the dataset here: https://archive.ics.uci.edu/ml/datasets/Activity+recognition+with+healthy+older+people+using+a+batteryless+wearable+sensor.

The dataset contains files, each corresponding to observations on one participant. They are organized into two directories, depending on the environment the trials were completed in. Each column in each file represents: (1) time in seconds, (2) acceleration in G for frontal axis, (3) acceleration in G for vertical axis, (4) acceleration in G for lateral axis, (5) ID of antenna reading sensor, (6) received signal strength indicator (RSSI), (7) phase, (8) frequency, and (9) activity label. For our purposes, we wish to build a model that can predict the ongoing activity label given the other features.

Question 1 [10 marks]

Create a Keras data generator class for this dataset. The data generator should be usable for training or testing a Keras model. The data generator should perform appropriate pre-processing on the sequences (e.g. scaling, normalization, etc.). You may also consider adding data augmentation in the data generator.

The data generator class must be called "ActivityRecognitionDataGenerator" and it should have the following three member functions: "__init__", "__len__", "__getitem__" (you may also use other helper functions).

The function "__init__" is a constructor that should take two input arguments (in addition to "self"). The first input argument is a list of full file paths to a subset of files containing the data. The second input argument is the batch size.

The function "__len__" should take zero input arguments (in addition to "self"). It should return the total number of batches in one epoch.

The function "__getitem__" should take one input argument (in addition to "self"). The first input argument is a batch index. The function should return two values: (1) a batch of data and (2) the activity labels for the batch of data.

Question 2 [10 marks]

Write a function that creates, trains, and evaluates a recurrent neural network to translate a sequence of sensor data to a sequence of activity labels.

The function must be called "activity_rnn_model".

The function should take one input argument: (1) a list of full file paths to files containing the data.

The function should return three values: (1) a Keras model object that is a recurrent neural network for this dataset, (2) performance of the model on the training set, and (3) the performance of the model on the validation set.

(Hint: use the data generator you wrote; randomly split the dataset into a training set and a validation set)

Question 3 [10 marks]

Write a function that creates, trains, and evaluates an attention-based neural network to translate a sequence of sensor data to a sequence of activity labels.

The function must be called “activity_attention_model”.

The function should take one input argument: (1) a list of full file paths to files containing the data.

The function should return three values: (1) a Keras model object that is an attention-based neural network for this dataset, (2) performance of the model on the training set, and (3) the performance of the model on the validation set.

(Hint: use the data generator you wrote; randomly split the dataset into a training set and a validation set)

Experiment Component

Question 4 [10 marks]

Conduct a series of experiments on the attention-based neural network you implemented for this. For each of these experiments, plot the performance on the training set and validation set as a function of the parameters.

a. Experiment on the size of the attention layers.
b. Experiment on the number of attention layers.
c. Experiment on the number of epochs of training.
d. Experiment on the value of the learning rate.

Report the parameters and performance of the model with the best performance.

Using the best performing model you found, compute performance on the held-out test set (i.e. a subset of the provided files).

(Hint: use the data generator you wrote)