Machine Learning for Communications

Project #3

Deep Neural Nets for DOA Estimation

Due November 24, 2019

Note: You may use outside references including books and notes and may discuss the project with your classmates in general terms. However, you may not obtain code or direct assistance from another person.

PLEASE MAKE SURE YOU ANSWER ALL PARTS OF EACH PROBLEM AND CLEARLY MARK YOUR FINAL ANSWERS. YOUR ANSWERS SHOULD BE AS COMPLETE AND CLEAR AS POSSIBLE NOT JUST A LISTING OF THE ANSWER. MATLAB CODE SHOULD BE CLEARLY DOCUMENTED AND SUBMITTED SEPARATELY FROM YOUR REPORT IN A ZIP FILE PER THE INSTRUCTIONS IN LECTURE #1.

I pledge that I have neither given nor received any unauthorized assistance on this project.

____

(signed)

____________________________ ____________________

Name (print) Student Number

1 Mini-Project Overview

In this project you will build, train and evaluate a deep neural network for estimating the direction of arrival of an incoming signal (or signals) using a four-element linear array with spacing of half a wavelength.

2 Detailed Description

You are going to use Python and TensorFlow to complete this project. Here are some instructions regarding the installation of Python and TensorFlow on windows:

- Install Anaconda software from: https://www.anaconda.com/distribution/#download-section.
- Open the Anaconda Navigator.
- Type in "Anaconda prompt" in the search bar of your windows, not in the anaconda Navigator.
- Type in "conda create -n project3 tensorflow=1.14" to create a virtual environment on the Anaconda Prompt
- Type in "conda activate project3" to activate the virtual environment.
- Select "project3" from the dropdown list in the Anaconda Navigator.
- Click the Jupyter install icon in the navigator.
- Click the Jupyter Launch icon in the navigator
- Create a new folder "Project3".
- In your created folder, create a new python file.
- Run "pip install keras" in a cell.
- Your code goes in here.

As you have already seen in the class, a deep neural network can be built and trained in Python for estimating the direction of arrival of an incoming signal using the following code:

- # Create model
  - model = Sequential()
- # Input layer (8 real values - 4 complex values)
  - model.add(Dense(20, activation="tanh", input_dim=8, kernel_initializer="uniform"))
  - model.add(Dense(1, activation="linear", kernel_initializer="uniform"))
- # Compile model

  - model.compile(loss='mse', optimizer='adam')

- # Fit the model
  - model.fit(X1, Y1, epochs=500, batch_size=10, verbose=2)

- # Calculate predictions
  - PredTestSet= model.predict(X1)
  - PredValSet= model.predict(X2)

Notice that in the above code, X1 and X2 contain the training and testing features, respectively. Note that you may want to import the following libraries:

- from keras.models import Sequential
- from keras.layers import Dense
- from sklearn.metrics import r2_score
- import matplotlib.pyplot as plt
- import numpy
- import math

3 Required Validation

- Develop a model that will provide less than 0.01 degree of RMSE in the absence of noise. You should attempt to do this over the full range (i.e., 100% field of view). Repeat for 90% field of view. How much training did you need? Which training data did you use? Feel free to use as much training as you like. A matlab function is provided that will generate either (a) rx samples or (b) covariance data. Which input data worked better? Why?
- Determine the performance of your estimator at 0dB-10dB in increments of 2dB. Compare to the CRLB. Did you have to train the estimator for each SNR separately? Or could you train over a range of SNRs? Which is more realistic?
- Choose any conventional DOA estimator and compare its peformance to your DNN-based estimator (and the CRLB).
- Repeat for 2 signals. Was more training needed? How much more? Why?
- BONUS: Repeat for 4 signals.