Hochschule Ravensburg-Weingarten
University of Applied Sciences

# PREDICT HAND MOVEMENT

## Using LSTM Neural Network on a BBC Micro:bit

JUNE 12, 2019

RAJ KUMAR RANA | 27916 | ELECTRICAL ENGINEERING AND IT
MANISH KARKI | 27915 | ELECTRICAL ENGINEERING AND IT

**FH SUPERVISOR:** DR. RER. NAT., PROFESSOR MARKUS PFEIL

# Introduction

[1]

*Figure 1. The photo shows BBC Micro Bit with its original packaging behind it.*

In this project, we train a LSTM (Long short-term memory) to classify a movement of a BBC Micro:bit using its accelerometer into one of the given movements; ideal, up, down, left and right.

The requirements of the project are as follows:

| Hardware | Software |
|---|---|
| • Two BBC Micro:bit<br>• USB Type A to Micro B 5-Pin cables | • Python 3.6.7<br>• keras 2.2.4 library (Tensorflow 1.13.1 as backend)<br>• numpy 1.16.2 library<br>• pandas 0.24.1 library<br>• matplotlib 2.2.2 library<br>• scikit-learn 0.20.3 library<br>• Jupyter notebook<br>• microfs 1.3.1 library<br>• pyserial 3.4 library<br>• mu-editor 1.0.2 library |

[1] [Source] By Aruld - Own work, CC BY-SA 4

## Description

The top view of the directory looks as shown below.

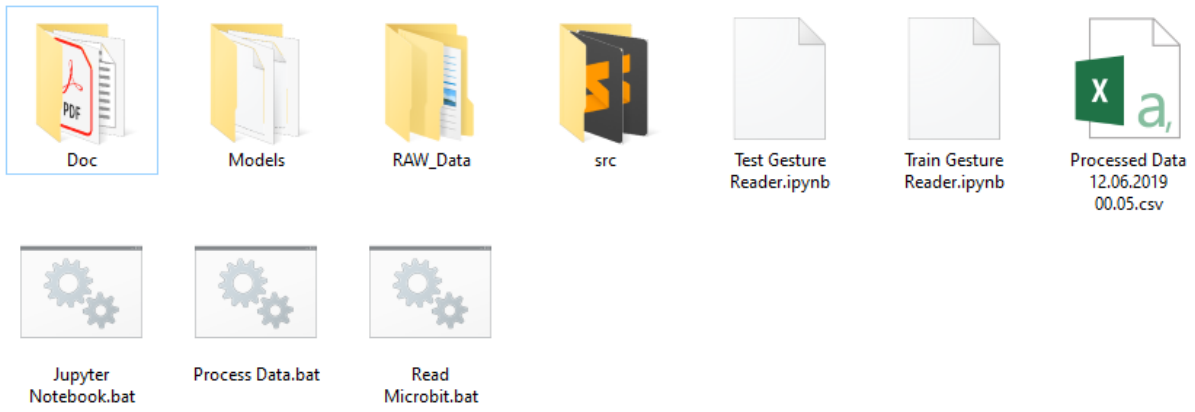



*Figure 2. Top view of main directory*

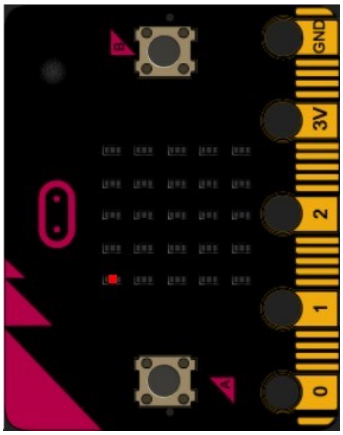
The description of each component is as follows:

- **Doc:**
  This directory contains documentation of this project.
- **Models:**
  After training the models, they are saved here in **HDF5** format. The format of model name is as follows:
  **Model <Test Accuracy> <Date> <Time>.HDF5**
- **RAW_Data:**
  When we read the collected data from the BBC Micro:bit using the "Read Micro:bit" batch file, they are saved here in a subdirectory. The naming format of the subdirectory is as follows:
  **<Label Type> <Date> <Time>**
- **src:**
  This directory contains all the source codes of this project.
- ***.ipynb files:**
  These are the Jupyter Notebook files that contain codes with documentation about the training, testing and a simple usage of the trained model.
- **Processed Data (.csv file):**
  This is the processed csv file that contains all readings from the csv files in RAW_Data directory. The data are read from RAW_Data, processed (with moving average windows and normalization) and stored in a single csv file. The programs in Jupyter Notebook use this csv file to train and test the model.
- **Batch files (.bat):**
  These batch files are shortcuts to perform following actions:

  **Jupyter Notebook.bat:** Opens Jupyter Notebook at current directory.
  **Read Micro:bit.bat:** Copies data from BBC Micro:bit to **RAW_Data** directory.
  **Process Data.bat:** Processes the data in **RAW_Data** directory and creates **"Processed Data . . . .csv"** file.

## • Data collection in BBC Micro:bit

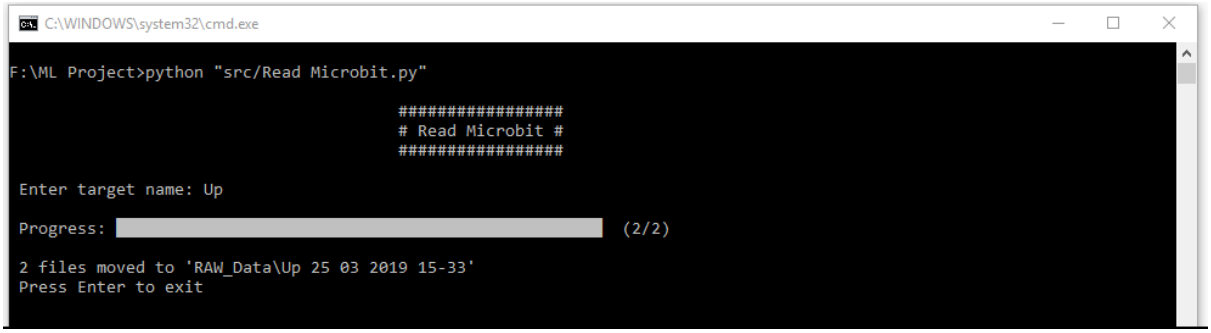
1. Plug in a BBC Micro:bit via USB to the computer
2. Open up the Mu-Editor and load the “Collect Data.py” file
3. Click on the “Mode” option located at the top left of the editor and select “BBC Micro:bit”
4. Flash the file to the BBC Micro:bit by clicking on the “Flash” option in the Mu-Editor
5. Restart the BBC Micro:bit and power it with a power source i.e USB or battery (USB preferred)
6. Hold the BBC Micro:bit at position as shown below:




7. Then choose the number of data to collect by pressing button A or B. (Max 14 data)
8. Press both A and B to start collecting data.
9. When you press the button A, a countdown of 1 sec will be displayed.
10. After the countdown, BBC Micro:bit samples and saves the acceleration data for 1.5 seconds at 0.1 second interval. So perform the movement after the countdown.
11. You can see the number of data left to be collected by pressing the button B.
12. To collect further data, repeat step 9.
13. In the end, a smiley face will be displayed to indicate that data collection is finished.

## • Data retrieval from BBC Micro:bit to computer

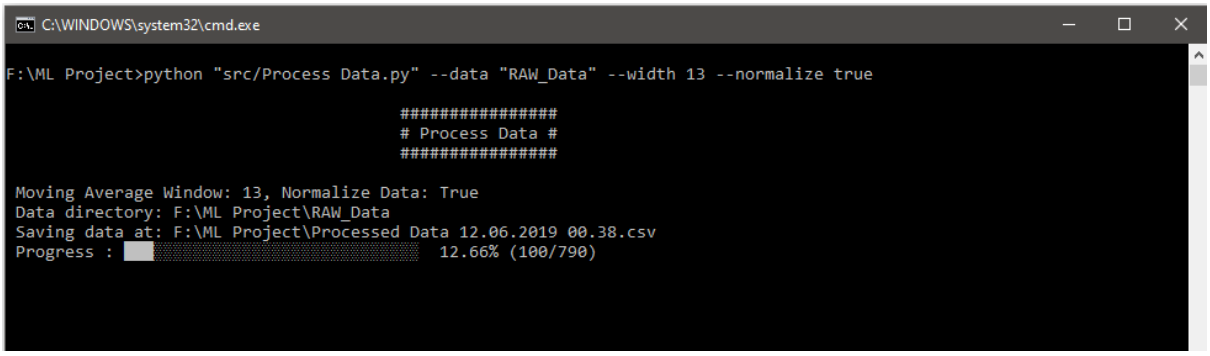
1. Plug in the BBC Micro:bit with data via USB to the computer
2. Open the “Read Micro:bit” batch file which runs the “src/Read Micro:bit.py” file.




3. Then type the name of the movement data type collected in the BBC Micro:bit.
4. Now all the csv data files are moved from the BBC Micro:bit to the **RAW_Data** directory.

## • Data processing of the collected raw data

1. Once you have sufficient data collected in the RAW_Data directory, open the “Process Data” batch file which runs the “src/Process Data.py” file with given data directory, moving average window width and normalization arguments.




2. In the end, there will be **“Processed Data <date> <time>.csv”** file ready to be loaded for training and testing the model.

## • Model training and testing

1. Open the “Jupyter Notebook” batch file.
2. Load the “Train Gesture Reader.ipynb” file.
3. Select the processed csv file for training and testing of the model.
4. Run all the cells.
5. In the end, decide accuracy threshold to save the trained model or not.

## • Predicting hand gestures using trained model

1. Flash “src/Data Sender.py” in one BBC Micro:bit and “src/Data Receiver.py” in another using Mu-Editor.
2. Connect the receiver BBC Micro:bit via USB to the computer.
3. Load the “Test Gesture Reader.ipynb” file in the Jupyter Notebook.
4. Load a model from the “Models” directory.
5. Configure the serial port configuration of the receiver BBC Micro:bit in the code.
6. Run all the cells.
7. In one cell, the program waits for data from receiver BBC Micro:bit via USB.
8. Then turn on the sender BBC Micro:bit and hold it like during the data collection process.
9. Press button A to start countdown.
10. After the countdown, perform the movement.
11. You can see the data transferred from sender to receiver BBC Micro:bit by looking at the LEDs.
12. Once the receiver transfers all data to the program via USB, it will display a tick sign as it receives an end token at the end of the data.
13. If the end token packet was lost, press the button B on the transmitter BBC Micro:bit to send the end token manually.
14. Then the program in Jupyter Notebook processes the data, plots its graph and displays the predicted movement.