Lab 1: Time Series Data and Pre-Processing

Introduction

In your previous study of neural networks, you likely used image data. While image data is common for machine learning applications, it is not the most common data type for low-power embedded machine learning applications. More commonly, you will use time series data, for example from inertial sensors/inertial measurement units (IMU).

IMUs can be found in many devices including motorcycles, aircraft, and drones. They are used for, i.a., navigation and vehicle tracking purposes but also for gesture recognition or Human Activity Recognition (HAR). IMUs consist of accelerometers and gyroscopes, with some including also a magnetometer. The Arduino we use for this course contains an IMU consisting of an accelerometer, a gyroscope, and a magnetometer.

In this lab, you will use IMU movement data, preprocess this data and build a system that is able to recognize at three different gestures. You can either use self-recorded data, or the data we provide you. For this lab, you will use Edge Impulse.

This lab contains quite some questions. We don't expect you to write long answers for them. However, at least a minimal explanation is required. Screenshots without any text do not count.

Submission Instructions

For the submission of this lab, please document what you did (please include screenshots) and answer the questions below. Please upload a PDF with your documentation and your answers to the questions, as well as your presentation slides when submitting this lab.

Part 1 – Setup

Task 1. Create an Edge Impulse account[1] and create a new project in Edge Impulse.

(optional) Task 2. Install the *Edge Impulse CLI* and *Arduino CLI*[2], connect the Arduino, and verify that your device is connected to Edge Impulse.

Part 2 – Data Collection or Uploading

Task 3a. If you decide to collect your own data, collect data for at least 3 motions. Otherwise, proceed with *Task 3b*.

[1] `https://edgeimpulse.com/`

[2] Installation instructions for the CLI tools: `https://docs.edgeimpulse.com/docs/edge-ai-hardware/mcu/arduino-nano-33-ble-sense`

Your motions can be, i.a., *side movements*, *vertical movements*, *L-shaped movements*, *circular movements*, *diagonal movements*, *waving motion*, *snake motion on your desk*, etc. Collect at least twenty 10-second-samples per motion. Each sample of your motion should contain a periodically repeating pattern of the same motion. Make sure to perform variations on the motions. For example, perform both slow and fast movements and potentially vary the orientation of the board. You'll never know how your user will use the device. Only vary the orientation if you don't have two motions that otherwise look the same, e.g., don't do it if you chose both an *up-down* and a *left-right* motion. In addition, record idle data. Record the same amount of data without moving the device as for each of the classes (e.g., twenty 10-second-samples). Include both samples keeping the device laying on the table and those where you hold the device in your hand.

Task 3b. If you decide not to collect your own data, upload the provided data to Edge Impulse. The data contains three different motions (*side movements*, *vertical movements*, and *circular movements*) plus an idle class with no movement.

Task 4. Look at your data. Do all of your samples for the same class look similar? If you recorded the data yourself, and you have outliers, you might want to re-record that sample.

> **Question 1.** Which features/patterns are representative for each of your motions?
> **Question 2.** Your data doesn't only include movements but also idle data. Why might it be beneficial to include idle data?

Task 5. Split your data into *train* and *test* data. You can do that in the *danger zone* on your Dashboard. Check on your data acquisition page whether the train and test sets are balanced. If not, move samples between the sets to balance them.

Part 3 – Impulse Design

Task 6. Create an Impulse with the following three processing blocks (*Spectral Analysis*, *Flatten*, and *Raw Data*) and one classification block each. If you like, you can also test the *Spectrogram* processing block. **For this lab, we want to use only the accelerometer data.** Thus, deselect all the other data sources. Save the Impulse.

> **Question 3.** Which settings did you choose in the *Time series data* input block? Explain what these settings do and why you think that your choices are a good choice. It might be a good idea to look at the patterns present in your data.

Task 7. Explore the different preprocessing blocks and generate features.

Question 4. What do the different preprocessing blocks do? Briefly explain them. Why should we use preprocessing blocks instead of feeding the data directly into the neural network like using the raw preprocessing block?
Question 5. Which preprocessing block creates the best features separating the classes? Add screenshots from the feature explorer. What kind of features did the blocks extract?

Task 9. Continue with at least the *Spectral Analysis* and the *Raw Data* blocks. If one of the other blocks looks more promising to you, continue using this one as well. For each of your processing blocks (you decided to continue with), train multiple classifiers. You shall train a *CNN classifier*, a *Dense classifier* and a *CNN+Dense classifier*. Please note to add a reshape layer before your first CNN layer and a flatten layer after your last CNN layer.

You can start with the visual (simple) mode to create your neural network architecture. But make sure to switch to Keras (expert) mode[3] to understand the model in detail. If you like, do some modifications here using your previous knowledge. Select your board (`Arduino Nano 33 BLE Sense`) in the upper-right corner before training to get the correct profiling information.

Question 6. What are 1D Convolutions? How do they differ from 2D convolutions used for images?
Question 7. Briefly explain your models. Is there anything special? Include the code for your models in your submission.
Question 8. What is the training performance of the models? How much memory is your model expected to need? What execution time is estimated? You can add screenshots. Check also for the memory usage and execution time of your DSP blocks.

Part 4 – Model Testing

Task 10. Evaluate your models in the *Model Testing* section.

Question 9. How good do your models classify the test data? Do the models generalize well? Create plots (bar graphs) comparing your models. Also explain *F1 Score* and *Uncertainty* displayed in the confusion matrix.

Task 11. If your models didn't perform well, head back to the data acquisition and record additional data and/or modify your models with the knowledge you have to avoid overfitting.

Task 12. After finding well-functioning models, you will use next an automated method for finding a good model. Head to *Experiments* and use the EON Tuner[4] to automatically search for a good model. This might take a while.

[3] You can find the expert mode behind the three dots next to the title "Neural Network settings".
[4] https://docs.edgeimpulse.com/docs/edge-impulse-studio/eon-tuner

Question 10. Did the EON Tuner come up with a better model than you? If so, in which regard is it better? Is it still better when you limit it to using only accelerometer data? *(To answer the latter question, first answer Question 11.)*
Question 11. If the EON tuner resulted in a better model, add this model as a new impulse and use it as well in the *Model Testing* section, and add its results to the plots you created in Question 9. *Please note: Before doing so, save your current version with the versioning tool on the left.*

Task 13. From this task onward, we will only use your best performing model. Save your previous state with the *Versioning* tool in Edge Impulse, and afterward remove all blocks no longer needed from your impulse design. Train your impulse one more time.

Part 5 – Live Classification

Task 14. Head to *Live classification* and test your model with an Arduino Nano 33 BLE[5]. Try each of your motions.

Question 12. Explain the output of the classification results. Does your model work? Did it misclassify some timestamps? Is this a bad thing? Why (not)?

Task 15. Head to *Devices* and add your phone as an additional device. You can just scan the QR Code for this. Now perform live classification with your phone. You might want to lower the sample lengths. Additionally, you can try to switch to classification mode on your phone to try continuous classification.

Question 13. Does the classification also work for your phone? If it doesn't work, why not? Does the performance change when you change the orientation of your phone? In which orientation do you have to hold your phone for it to work best?

[5] You will find devices in HRS3, room 502, a room you can use throughout the semester.