University of Oxford

Group Design Practical

Group 14

Machine Learning in the Browser with the BBC Micro:Bit

Louis-Emile Ploix[1], Ike Glassbrook[2], Joseph Simkin[3], Alikhan Murat[4], Andy van Horssen[5], and Ewan Hawkrigg[6]

Internal supervisor: Qian Xie

External supervisor: Robert Knight

May 2024

[1]louis-emile.ploix@stcatz.ox.ac.uk
[2]isaac.glassbrook@lmh.ox.ac.uk
[3]joseph.simkin@some.ox.ac.uk
[4]alikhan.murat@magd.ox.ac.uk
[5]andy.vanhorssen@sjc.ox.ac.uk
[6]ewan.hawkrigg@keble.ox.ac.uk

Contents

1 Introduction

The Group Design Practical is a course taken by all 2nd year undergraduate students at the University of Oxford studying for a degree in Computer Science, Mathematics and Computer Science, or Computer Science and Philosophy. This report details the work of Group 14 from February to May 2024 to design, implement, and deploy a product satisfying the specification as provided by Micro:Bit.

1.1 Technical context

Our project is based on a machine learning tool developed by researchers at the Centre for Computational Thinking and Design at Aarhus University. This tool is publicly deployed at `https://ml-machine.org`, with its source code available at `https://github.com/microbit-foundation/cctd-ml-machine`. Our project is a fork of this repository, extending the existing project.

ML-Machine is an application for users with access to a physical micro:bit device to train a machine learning model that uses the micro:bit's sensors as input data. The user is given a 3 step process:

1. Named 'Gestures' are added, for the trained model to eventually use as classification categories. The user then adds to each gesture several recordings of the accelerometer that correspond to the particular gesture. Alternatively, they can use the example dataset, containing the categories 'shake' (with recordings of the micro:bit being shook), 'still' (with recordings of the micro:bit not moving), and 'circle' (with recordings of the micro:bit being moved in a circle).

2. The user trains a model – either a dense layers model, or a k-nearest neighbours model – to classify the data. The recordings are pre-processed using a set of filters, which build the feature set. Before training, the user may select in particular which filters they would like to use.

3. The micro:bit then has its sensors periodically polled in order to predict, of the labelled gestures, the one most similar to its current movement.

The web application is written using the Svelte framework[7], acting essentially as an HTML template language. The micro:bit itself requires additional drivers to interface with the application, as first a bluetooth connection must be established, and then the appropriate sensor data must be streamed to the application.

[7] `https://svelte.dev`

These are written in C++, and then converted into `.hex` files, which users can install on their micro:bits in order to allow this procedure to work. Both models are trained in the browser using TensorFlow.js[8].

1.2 Project Specification

In consideration of the existing context, the specifications set out by Micro:Bit were designed with a view to utilise our work for experimental purposes, and to provide a benchmark for evaluating the feasibility of future projects. On this basis, the specification gave the following requirements of a final product:

- That a user should be able to train any model on not just the Micro:bit's accelerometer, but also an additional sensor.
  - The user should be able to choose which sensor's data is to be streamed into the training data (and thus, which sensor's data is polled once the model is trained).
  - Information given by the sensor should be visualised to the user in real-time in a manner which is understandable.
  - The sensor data should be amenable to machine-learning analysis via the models available.
- In addition to the Dense Neural Network, and the k-Nearest Neighbours models of the base application, a new neural network architecture should be investigated for implementation.
  - This network should be capable of predicting on simple patterns with reasonable accuracy.
  - The technical details of the network should be of pedagogical value, in addition to being amenable to high-level explanation.
  - The trained model should be of a size that could fit on the micro:bit itself, rather than needing to run on a connected device[9].
  - Model training should be responsive on standard browsers ran on computers with low-end modern hardware.

2 Logistics

This section describes the timeline of

2.1 Timeline

2.2 Role Delegation

3 Implementation

3.1 Sensors

3.1.1 Magnetometer

3.2 Models

3.2.1 Integration of new sensors with existing models

3.2.2 Mixture of Experts

4 Areas of Further Development

[8] `https://www.tensorflow.org/js`

[9] The Micro:Bit team have expressed a desire to run the prediction models directly on the micro:bit in future, rather than in the browser, allowing for the model to be stored independently of the application. This was not itself within the scope of our specification however.

5 Concluding Remarks