STEP-BY-STEP: TRAINING IMU-BASED GESTURES WITH LIVE FEEDBACK

Michael Schnebly [1]

ABSTRACT

Recognizing user-defined gestures in inertial measurement unit (IMU) data unlocks new forms of creativity and accessibility in human-computer interaction. However, training gesture recognition models is a difficult task that requires a deep understanding of machine learning. We present Step-by-Step, a software tool that allows users to train gesture recognition models with live audiovisual feedback. Step-by-Step uses a simple neural network to learn to recognize and distinguish multiple gestures in IMU timeseries data. Users can train the model by performing gestures and receiving live feedback on the model's performance. Step-by-Step is designed to be accessible to users with no machine learning experience, while providing a powerful codebase for advanced users.

1 INTRODUCTION

2 BACKGROUND

3 METHOD

3.1 Hardware

For measuring body movement, we used the Bosch BNO-055, a 9-axis IMU that includes an accelerometer, gyroscope, and magnetometer. The IMU uses I2C communication protocol to stream data to a microcontroller (Espressif ESP8266) which passes that data on to a laptop (Macbook Pro 2017) via USB for further processing.

Note that the particular hardware components used in this project are not essential to the system. They could be replaced with other options so long as the sensor data contains information necessary to recognize and distinguish the gestures of interest and the computer (or microcontroller if using purely embedded approach) is capable of running the software at a rate matching that of data collection.

3.2 Software

The software is written in Python and designed to be accessible to users with no machine learning experience while providing a powerful codebase for advanced users. The software is structured into three main components: data processing, neural network, and user interface. The data processing component handles data collection, labelling, and storage. The neural network component handles model structure, training, and inference. The user interface component handles user interaction and feedback.

3.3 Data Processing

3.3.1 Collection

The IMU data is collected as a timeseries of frames. Each frame is a 3D vector representing linear acceleration in each of the three axes: x, y, and z. The data is collected at a rate of 100 Hz and can be streamed directly from an IMU, recorded to a file, or loaded from a file.

3.3.2 Labelling

The data is labelled by associating each frame with a gesture. A gesture is a sequence of frames that represents a single movement of the body. Here, we focus on percussive gestures and assume that a gesture is complete at the local peak in linear acceleration's magnitude. This constraint allow us to label gestures using a simple peak-detection algorithm. When a local peak meet's user-defined criteria (e.g. acceleration is above a certain threshold), the peak frame is labelled as containing a gesture. All other frames are labelled as containing no gesture.

3.4 Neural Network

3.4.1 Structure

The input to the network is a sequence of frames, each of which is a 3D vector representing linear acceleration in each of the three axes: x, y, and z. The output of the network is a vector of probabilities, one for each gesture. The network is trained to maximize the probability of the

[1] School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA. Correspondence to: Michael Schnebly <michael_schnebly@g.harvard.edu>.

Final project of TinyML at Harvard University (CS249R - Fall 2023), Cambridge, MA, USA. Copyright 2023 by the author(s). Code available at www.github.com/michael-schnebly/step-by-step.

correct gesture and minimize the probabilities of all other gestures. The neural network is a shallow, two-branched model. One branch represents a recent history of sensor values while the other represents a recent history of model outputs. Combining the information from the these two sources, the model is trained to maximize the probability of the correct gesture and minimize the probabilities of all other gestures.

The sensor branch consists of an input layer (short timeseries of recent sensor values) and a 1D convolutional layer (recognizes patterns in the timeseries). The memory branch consists of an input layer (short timeseries of recent model outputs) and a max pooling layer (identifies the maximum probability that a gesture has been predicted recently). These branches are then concatenated and followed by a fully connected output layer.

3.4.2 Training

4 Findings

References

Author, N. N. Suppressed for anonymity, 2018.

Duda, R. O., Hart, P. E., and Stork, D. G. *Pattern Classification*. John Wiley and Sons, 2nd edition, 2000.

Kearns, M. J. *Computational Complexity of Machine Learning*. PhD thesis, Department of Computer Science, Harvard University, 1989.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (eds.). *Machine Learning: An Artificial Intelligence Approach, Vol. I*. Tioga, Palo Alto, CA, 1983.

Mitchell, T. M. The need for biases in learning generalizations. Technical report, Computer Science Department, Rutgers University, New Brunswick, MA, 1980.

Newell, A. and Rosenbloom, P. S. Mechanisms of skill acquisition and the law of practice. In Anderson, J. R. (ed.), *Cognitive Skills and Their Acquisition*, chapter 1, pp. 1–51. Lawrence Erlbaum Associates, Inc., Hillsdale, NJ, 1981.

Samuel, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):211–229, 1959.