NNonHexiwear:

A development environment to help you implement neural networks on Hexiwear

Dennis Shim, Seraphine Goh
UCLA
EE 202A

Introduction: Computing on the Edge

- Embedded devices
  - Increasing presence
- Edge computing
  - Analyze data from IoT devices locally (on the edge)
  - Send relevant data to cloud for further processing
- Importance:
  - Reduced latency
  - Increased reliability
  - Potentially increased security

Introduction: Mbed - Hexiwear

- Wearable IoT development platform
  - Mbed-Enabled → built-in embedded software
- 8 Sensors:
  - 6-axis Accelerometer & Magnetometer combo
  - 3-axis Gyroscope
  - Pressure sensor accurate up to Altitude sensing
  - Temperature and humidity combo
  - Ambient light sensor
  - Optical Heart rate sensor
- Other features:
  - BLE connectivity
  - Compact (2” x 2”)

Introduction: Neural Networks

- Neural Network (NN)
  - 1) Obtain data
    - Obtain training data (eg. from Hexiwear sensor)
  - 2) Training
    - Train NN to classify outputs based on training data set
  - 3) Actual implementation
    - Obtain new data to classify
    - Run new data through NN
    - Compare predicted & expected outputs

Problem Statement

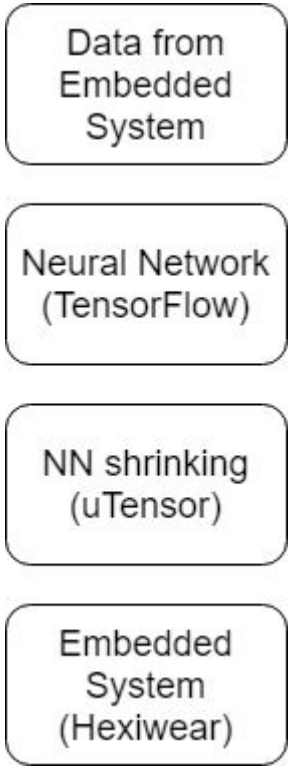
- Implementing NN is a fragmented, 3-step process
  - 1) Obtain data
  - 2) Training
  - 3) Actual implementation
- Goal: Incorporate NN implementations in edge computing
  - Put NN on embedded device (eg. Mbed compatible)
- Solution: Create library/API to implement NN on embedded system (Mbed)
  - Takes data from sensors in embedded system as input to NN
  - Streamline process of NN implementation in Hexiwear

Prior Work

- STMCube.AI
  - Map and run pre-trained NN on STM32 microcontrollers
  - Converts pre-trained NNs to code to run on MCU
- CMSIS-NN
  - Software library consisting of NN kernels
  - Maximize performance and minimize memory footprint of NN on Cortex-M processor cores
- uTensor
  - Machine learning framework for Mbed and Tensorflow
  - Takes model created/trained in Tensorflow
  - Converts to C++ code

Technical Approach

- Create a workflow to streamline the following process:
  - Obtaining training data for a neural network from sensors on an embedded system
  - Train a NN using the data in an existing framework, e.g. TensorFlow or Caffe
  - Shrink a full NN into smaller packaged C++ code that can be run on the embedded system
  - Add the shrunken NN to the embedded system and have it take in sensor data as input



Data Acquisition from Hexiwear

- Focused on IMU sensor data
  - Accelerometer, Magnetometer, Gyroscope

- Mbed project that runs on Hexiwear
  - Receives serial input specifying period and number of data points
  - Reads data from sensors and outputs it over serial

- Python script
  - Requests data from Hexiwear over serial
  - Parses and formats incoming serial data

NN for Human Activity Recognition

- Used UCI's HAR dataset
  - Collected from Android phone, 3000 examples
  - Each example consists of 128 time-series data points at 50 Hz (2.56 seconds)
  - Accelerometer and gyroscope
    - Gravity component filtered out of accelerometer data

- Imported data into Tensorflow + trained DNN → .pb file
  - Trained multi-layer deep neural networks (DNNs) to classify IMU sensor data
  - Approximately 78% accuracy using a 2-layer DNN
  - Expect this to be lower in Hexiwear because sensors are different

uTensor and Hexiwear

- uTensor shrinks NNs and makes them ready for Mbed deployment
  - Note: only certain mathematical operations are supported, we attempted to implement an LSTM/RNN but it didn't work

- Added uTensor to an Mbed project
  - .pb Tensorflow NN file converted to .cpp

- Main loop
  - Include the .cpp file generated by uTensor
  - Sample data from IMU sensors (filter out gravity)
  - Give data as input to NN and get output

Future Directions

- Retrieving all data
  - Options to have more sensors - IMU, barometer, humidity, light, heart rate
  - Making data retrieval easier by using Bluetooth/BLE

- Different/better NN (LSTM/RNN, convolution, etc.)
  - When uTensor adds the functionality or use a different one

- Platform generic - support more general Mbed devices
  - Streamline development even more