MyoPose Final Report

Andrew Fantino

Motivation & Objective

Objective

Detect finger position with electrical signals from forearm muscles with hobby grade hardware and novel deep learning techniques

Motivation

Meta Orion AR glasses use sEMG wristband as controller

Project Impact

Framework for research in prosthetics and XR interaction

Goals and Deliverables

Open source framework for finger pose detection with

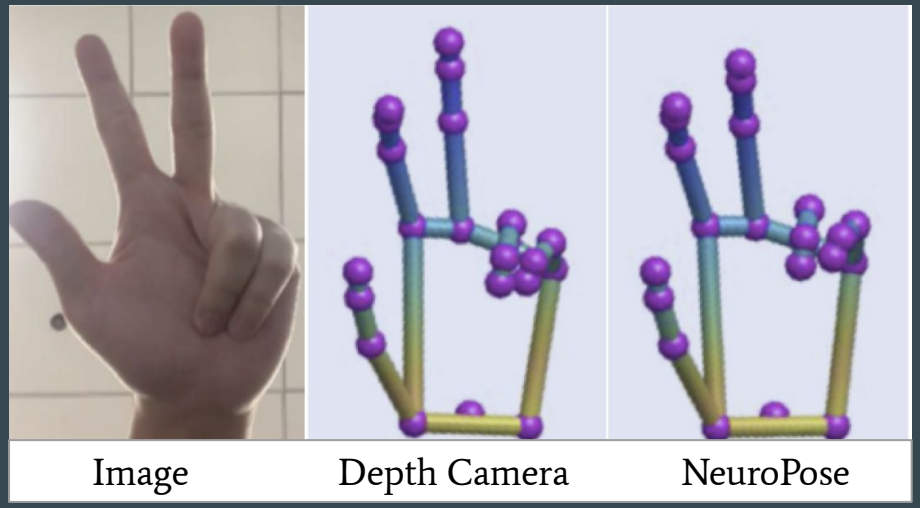
Technical Approach and Novelty

Current SotA: NeuroPose

- Uses Myoband (deprecated) + biological model of finger positions
- Uses 5 second window input to encoder-decoder architecture
- Attempted to use RNN, but slower and more power draw

MyoPose:

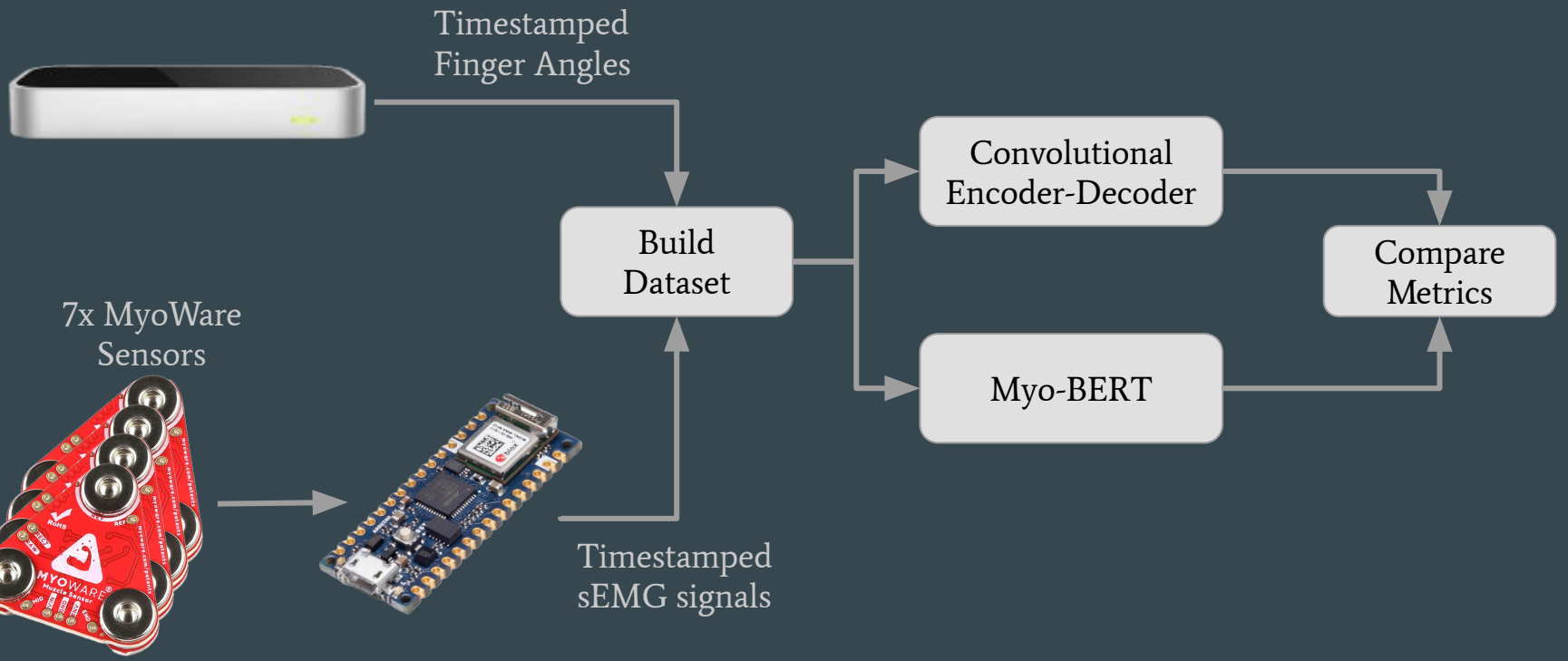
- Uses open source MyoWare EMG
- Use novel architecture based on LIMU-BERT and ViT



Methods

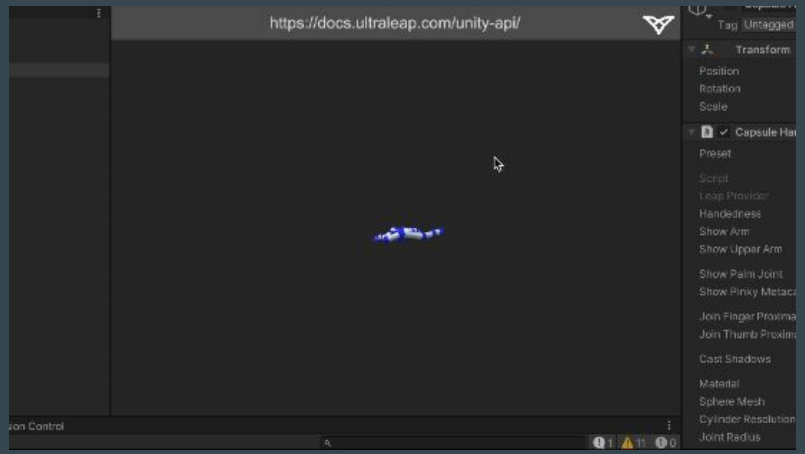
- No dataset, so I have to make my own
- Electrode placement is very finicky
- Implement NeuroPose convolution encoder-decoder and Transformer Architecture

Methods


Timestamped
Finger Angles
7x MyoWare
Sensors
Timestamped
sEMG signals
Build
Dataset
Convolutional
Encoder-Decoder
Myo-BERT
Compare
Metrics

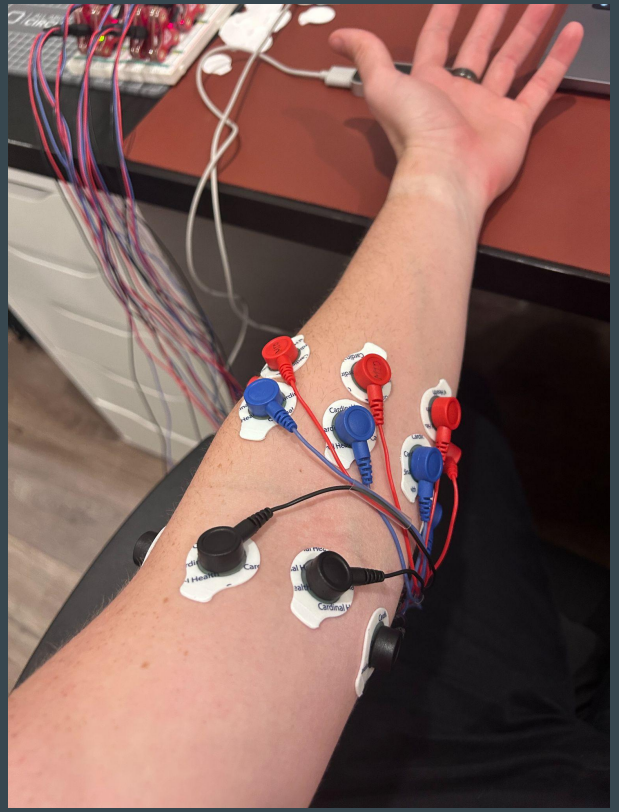
Leap Finger Angles

- Each joint has a vector3 representing their direction from the base to the tip of the joint
- Use projections to isolate the degrees of freedom and calculate euler angles
- Motion sensor cuts out sometimes and loss data



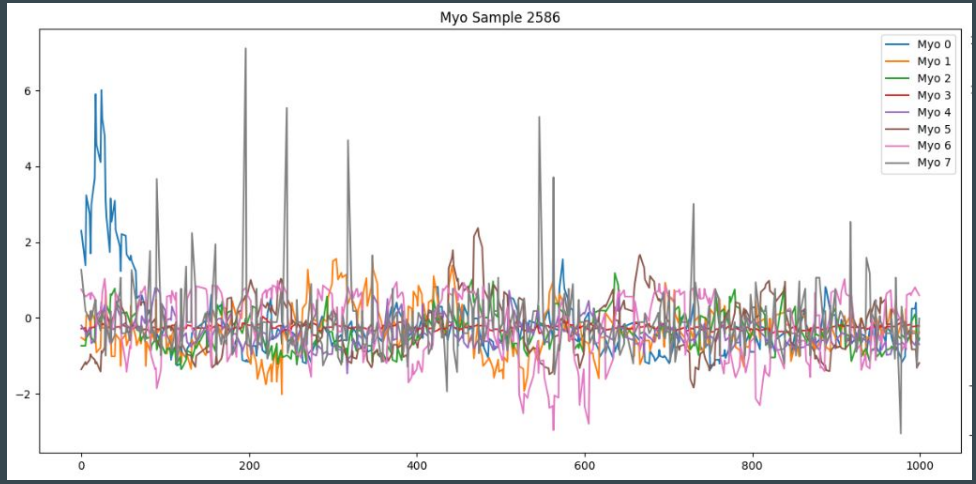
Stream sEMG Signals Through BLE

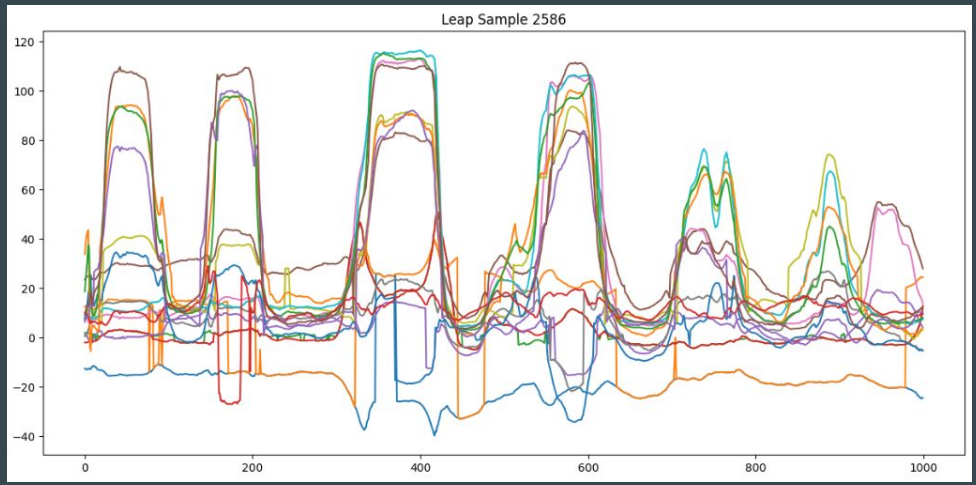
- Arduino Nano 33 IoT
- Uses json formatted in SenseML BLE and use online tool for live plotting of each signal to adjust the gain of the env filter on the MyoWare.
- `Bleak` script connects to MyoPose and saves samples in csv file



Data Cleaning

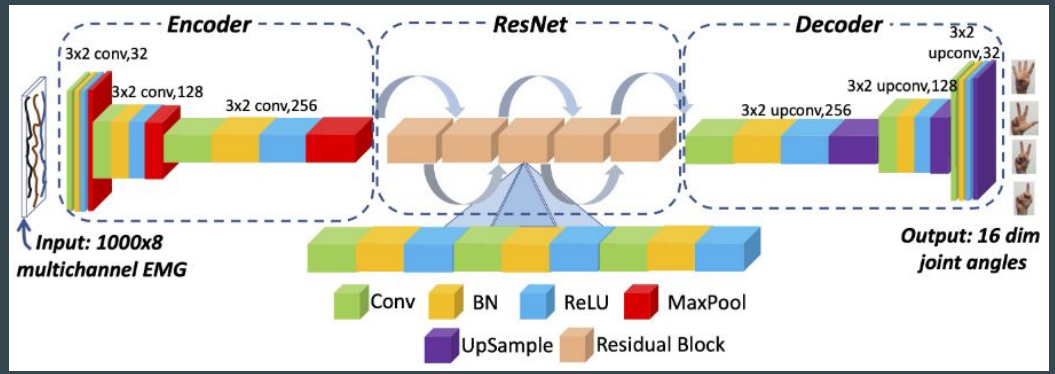
- Leap Motion Sensor loses tracking of hand
  - No labels for that portion of data.
  - Use simple interpolation between missing samples
- sEMG data timestamped at receive of Python script
  - Not constant, but allows for faster streaming data and no synchronization necessary
- Avg/Interp at each 5ms interval for both
- Normalize Myo signals with z-score for each channel





Autoencoder Implementation

- Input: [8 sEMG Channels, 1000 samples] → Output: [16 joint angles, 1000 samples]
- Hand skeletal constraints applied at output
- Custom loss function with smoothness term
- Added min-max normalization for thumb joint angles because no hand-skelatal constraints

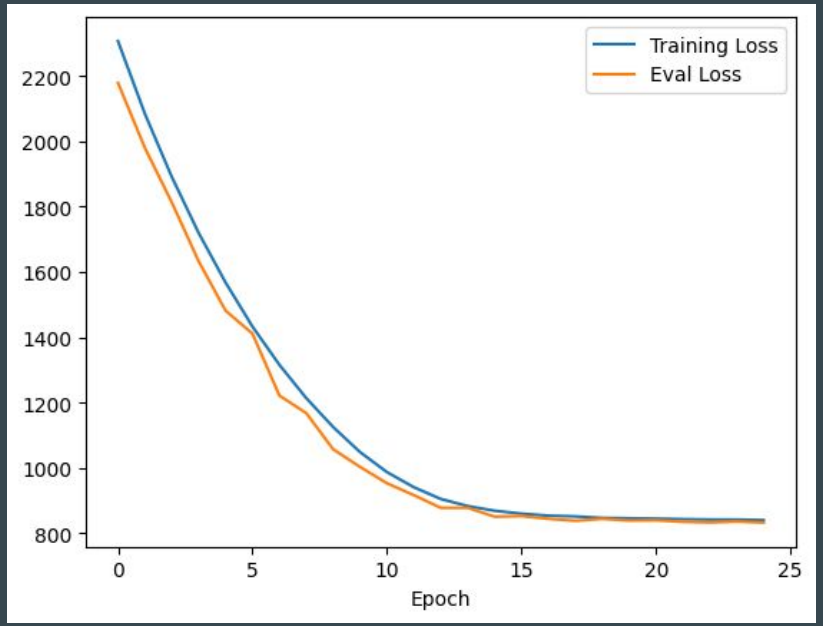


Evaluation Methods

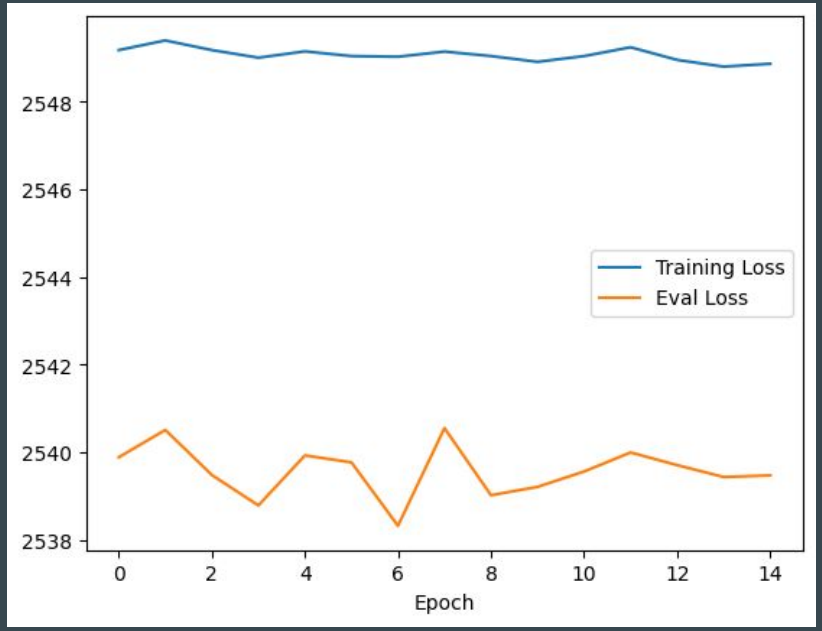
- 10 samples of 90 seconds each moving fingers in random motions
- Create windows and combine datasets
- Experiment with constraint normalized vs non-constrained
- Compare against NeuroPose stated performance and my implementations
- Experiment with bad electrode placement and less electrodes

Autoencoder Training

With Normalization

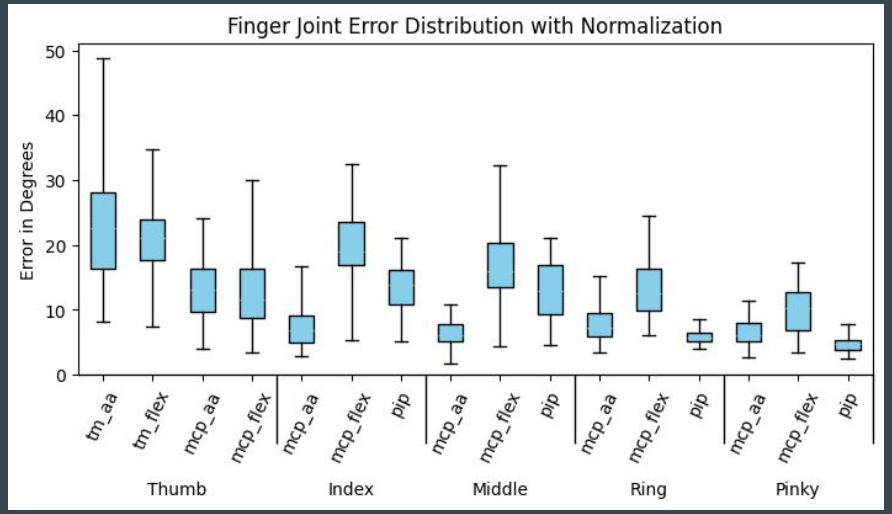


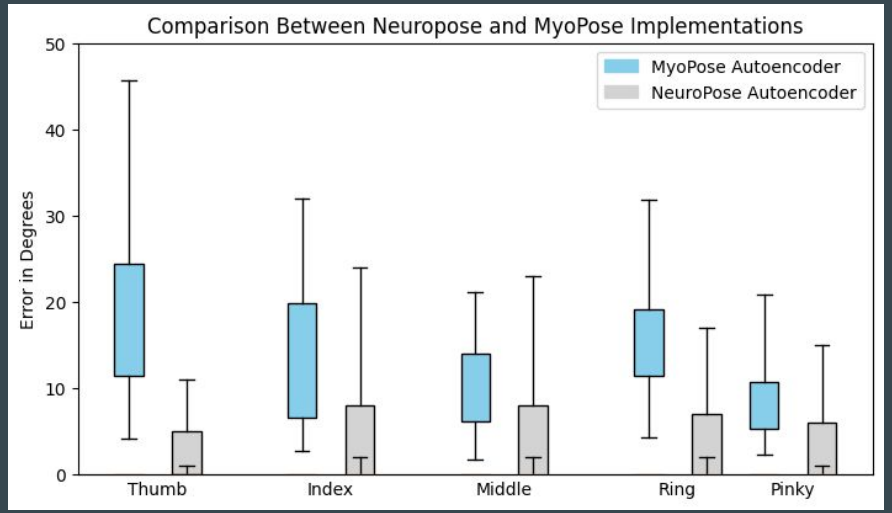
Without Normalization



Autoencoder Results

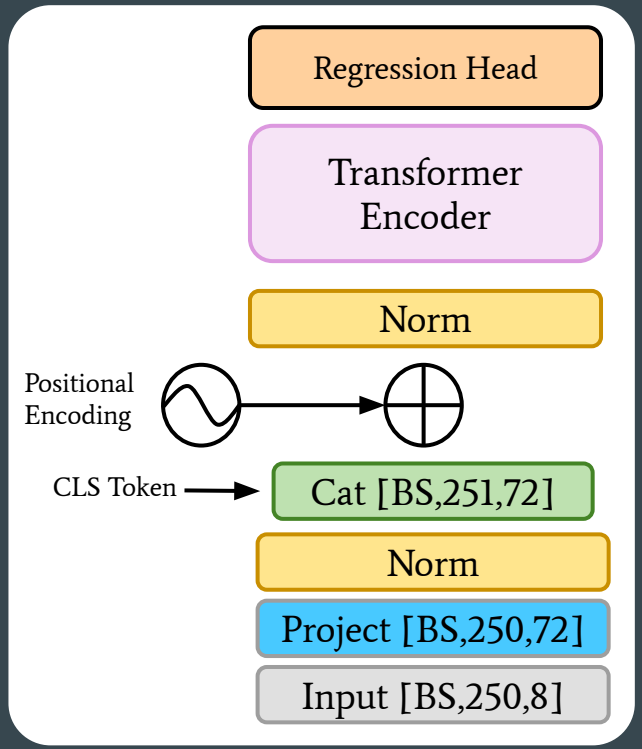
- 10-15% finger angle error
- Thumb is much less accurate than NeuroPose paper
- MCP abduction and adduction is much more accurate than other joints
  - This is likely because there is less variance in MCP_aa angles (±15°)
- Lower accuracy likely because of hardware limitations

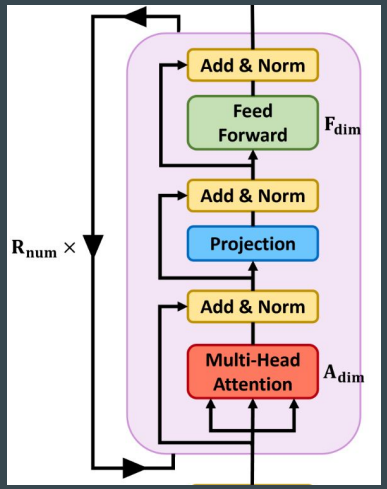




Transformer Implementation (Myo-BERT)

- Inspired by LIMU-BERT and ViT
  - 250 sample window (1.25s)
  - Prepend CLS token
  - Project to higher hidden dimension (72)
  - Positional embedding
  - Transformer Encoder R_{num} times with parameter sharing
  - Transformer out to Regression Head
- Memory encoded in model allows for smaller windows
- Pretrained weights from IMU data





Conclusions & Future Steps

- Potential to match NeuroPose performance with Myoware sensors
  - Very sensitive to electrode placement
  - Needs filtering, smaller packaging, and less wires to be viable
- Data interpolation is unreliable especially with long drop-outs of the Leap finger tracking
- Transformer Model is a WiP

- Miniaturize hardware
- Better filtering techniques
- Consider SeFT Architecture or more intelligent data interpolation

Questions