ASL Interpreter

Team B6: Aaron Selesi, Malcolm Fitts, Young Woo Kim

Application Area:

Use Case: Two way translation from English to ASL

Motivation: Assist communication between deaf and hearing individuals

Sign to Speech/Text

Speech to Text/Sign

Solution Approach: Neural Network

Base of the project is convolutional neural network that can learn to recognize signs.

Softmax Activation for Multinomial Classification ($y \in \{1, 2, \ldots, C\}$)

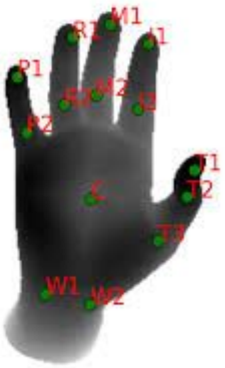
Implementation: Tensorflow/Keras libraries

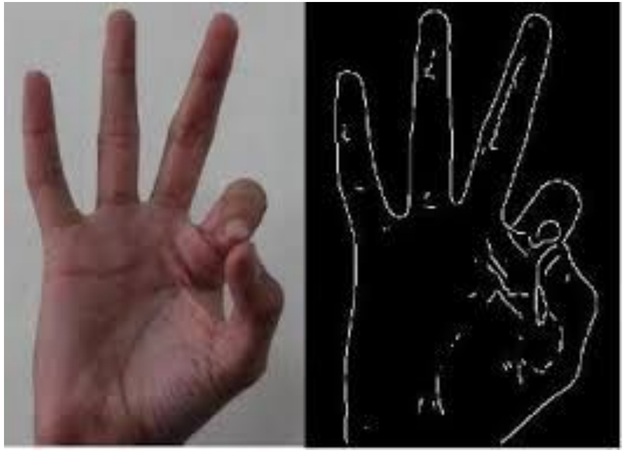
Categorical cross-entropy loss function, Adam optimizer,

Possible Architectures (order of complexity): LeNet-5, AlexNet, VGG, EfficientNet

Solution Approach: Feature Extraction

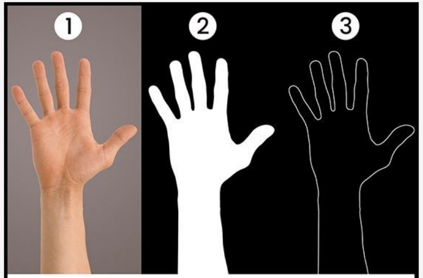
- Trying to pick important bits of data from the overall image to condense and input
- Detect Region of Interest / Bounding Box
- Detect hand outline
- Detect 5 fingertips, 5 joints, 1 center of palm



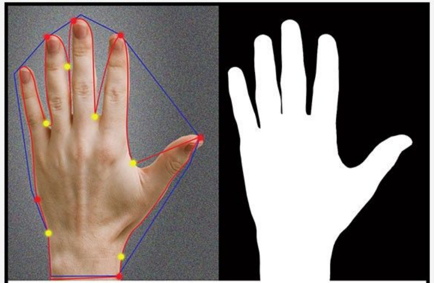


Solution Approach: Image Processing

- Kindle Fire Tablet camera used to capture image data
- OpenCV for image processing
- Smoothing
- Background Subtraction by Moving Averages for Dynamic gestures
- Edge detection

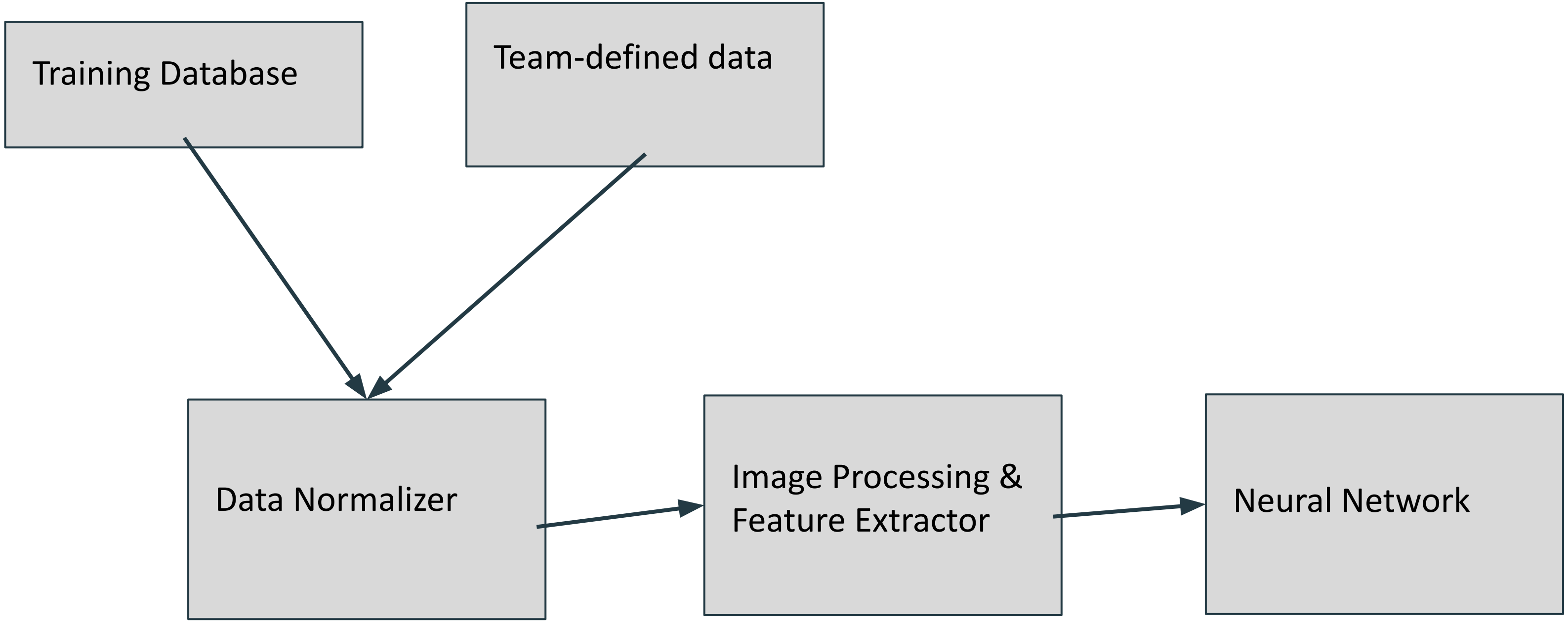


Edge Defection



Background Substraction

System Specification: Training



Training Database

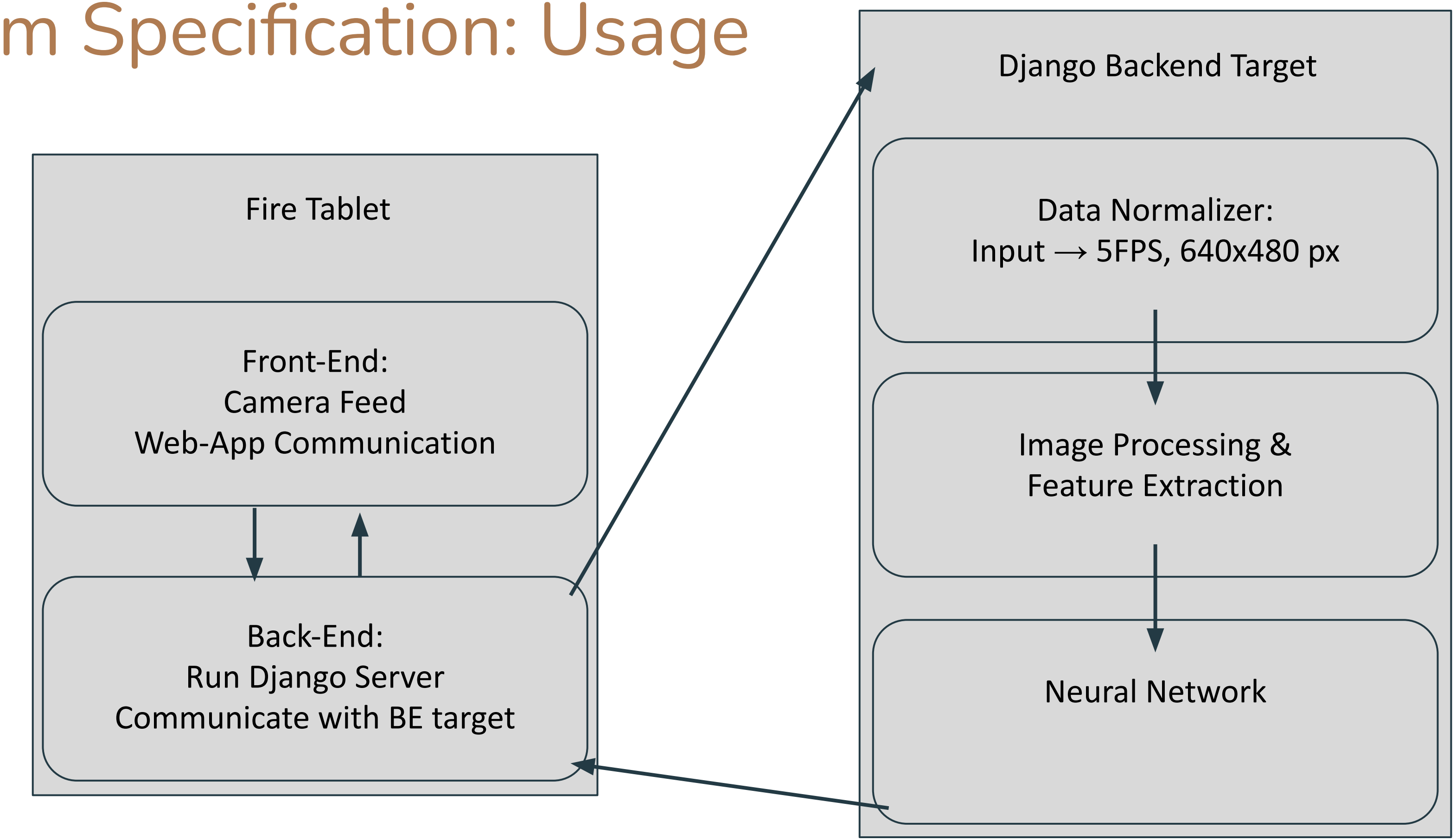
Team-defined data

Data Normalizer

Image Processing & Feature Extractor

Neural Network

System Specification: Usage



Fire Tablet

Front-End:
Camera Feed
Web-App Communication

Back-End:
Run Django Server
Communicate with BE target

Django Backend Target

Data Normalizer:
Input → 5FPS, 640x480 px

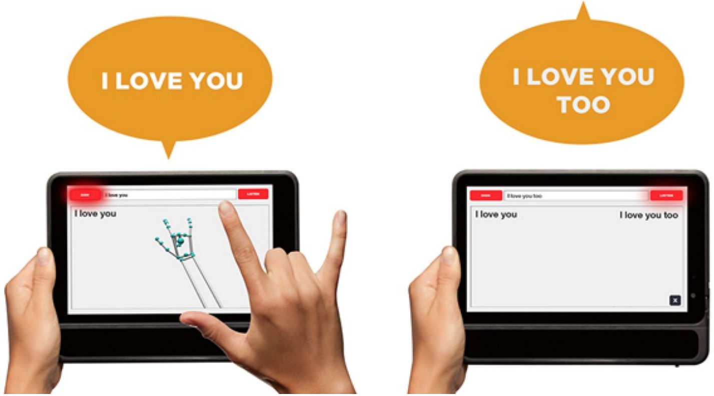
Image Processing &
Feature Extraction

Neural Network

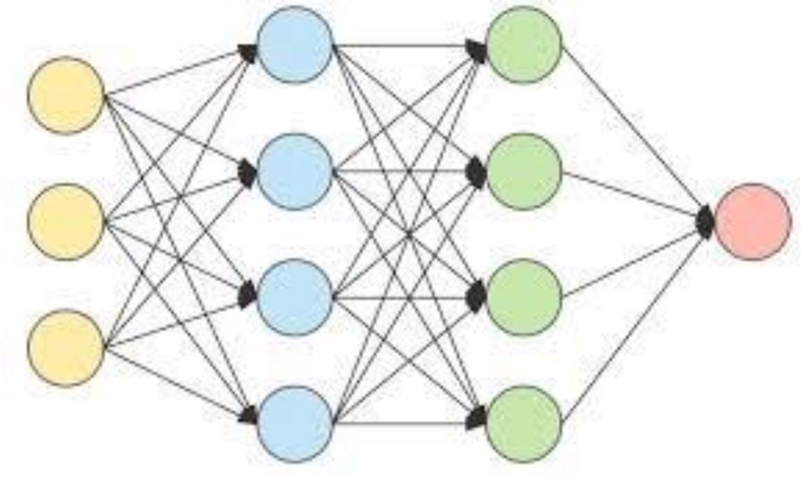
Implementation Plan: Django Web App

We are building a django web app that will serve as the interface for our ASL interpreter. It will run off our Kindle Fire 8 tablets that we purchased for this project and the final layout will look similar to an existing solution shown on the right.

By using a webapp we do not need to worry about platform making it accessible on many devices with a camera and internet connection.



Implementation Plan: Neural Network



We are planning to architect the neural network based on available data and our ability to parse it. We process the training data so the layer size is reasonable and so the network can be trained efficiently.

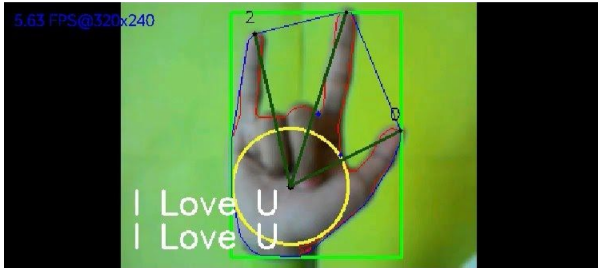
Reducing Scope: Recognizing 200+ words requires an output layer of 200+ nodes. So we are designing with the goal of 200 word lexicon

Packages: Keras, Tensorflow, numpy

Implementation Plan: OpenCV

To process the image data we are receiving from the Kindle Fire camera we will be using OpenCV. It will allow us to do edge detection of hand gestures.

This is made possible by the availability of the OpenCV api which allows for easy integration into python and the django web framework.



Metrics and Validation

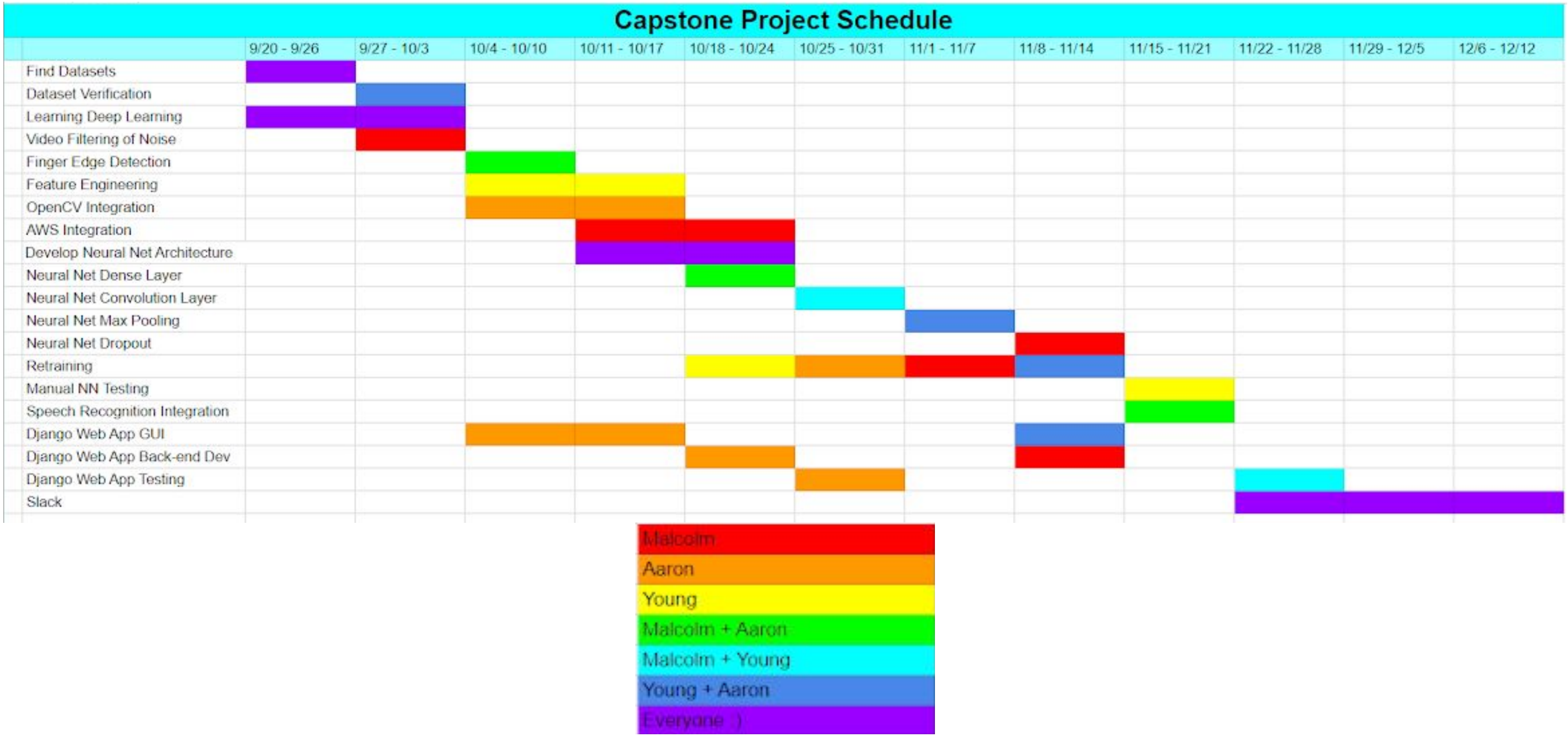
Web-App Metrics: Can handle requests with 2 seconds worth of video, Can handle simultaneous requests

Neural Network Metrics: Estimation accuracy, Best-N Guesses

Image Processing Metrics: Manual Review

Validation: Neural Network testing and re-tuning since NN has most quantitative tests available

Project Management / Gantt Chart



Capstone Project Schedule

	9/20 - 9/26	9/27 - 10/3	10/4 - 10/10	10/11 - 10/17	10/18 - 10/24	10/25 - 10/31	11/1 - 11/7	11/8 - 11/14	11/15 - 11/21	11/22 - 11/28	11/29 - 12/5	12/6 - 12/12
Find Datasets	🟪											
Dataset Verification		🟦										
Learning Deep Learning	🟪	🟪										
Video Filtering of Noise		🟥										
Finger Edge Detection			🟩									
Feature Engineering			🟨	🟨								
OpenCV Integration			🟧	🟧								
AWS Integration				🟥	🟥							
Develop Neural Net Architecture				🟪	🟪							
Neural Net Dense Layer					🟩							
Neural Net Convolution Layer						🩵						
Neural Net Max Pooling							🟦					
Neural Net Dropout								🟥				
Retraining					🟨	🟧	🟥	🟦				
Manual NN Testing									🟨			
Speech Recognition Integration									🟩			
Django Web App GUI			🟧	🟧				🟦				
Django Web App Back-end Dev					🟧			🟥				
Django Web App Testing						🟧				🩵		
Slack										🟪	🟪	🟪

🟥	Malcolm
🟧	Aaron
🟨	Young
🟩	Malcolm + Aaron
🩵	Malcolm + Young
🟦	Young + Aaron
🟪	Everyone :)

Project Design Summary

ASL Interpreter

- Trained on data set of american sign language gestures using deep learning
- Image data captured by camera on Kindle Fire tablet
- Image data processed by OpenCV and edge detection/feature extraction algorithms
- Run through a Django web app utilising the OpenCV api accessible through the Kindle Fire Tablets