Unnamed Sign Language Interpreter

Crimson Code Hackathon 2020

Kyle Newton, Jason Bowen, Dustin Peters, Colin Greely

Contents

[Plan 3](#_Toc33116661)

[Technologies 4](#_Toc33116662)

[Front End 4](#_Toc33116663)

[Back End 4](#_Toc33116664)

[Miscellaneous 4](#_Toc33116665)

[Roles 5](#_Toc33116666)

[Milestones 6](#_Toc33116667)

[Front End 6](#_Toc33116668)

[Back End 6](#_Toc33116669)

[Miscellaneous 6](#_Toc33116670)

[UML Diagrams 7](#_Toc33116671)

# Plan

We are planning to build a Machine learned Sign Language interpreter for a small subset of the basic American Sign Language Dictionary. The current plan is to train the model on each of the letters of the alphabet, excluding ‘J’ and ‘Z’ due to movement constraints at the time of writing this document. We plan on using a tensorflow model to train based on a dataset obtained from Kaggle. This will be paired with a frontend of react, which will handle the generation and sending of the images to a server that will then process them through the trained model.

We plan to train the model as soon as possible, starting at the beginning of the hackathon, while using the time that will be necessary to train the model to build the front end of the application.

The front end of the application will run as a web service, with mockups presented further into this document.

This document is to keep track of internal design decisions as well as milestones that we are planning on hitting throughout the 2020 Crimson Code Hackathon.

# Technologies

## Front End

* [React JS](https://reactjs.org/)
* HTML/CSS

## Back End

* Python
* Tensor Flow
* Numpy

## Miscellaneous

* No official deployment yet
* Dataset
  + <https://www.kaggle.com/grassknoted/asl-alphabet>

# Roles

* Kyle: Project Lead/Floater
* Colin: Build/Train the model/Floater
* Jason: Front end/Floater
* Dustin: Front end/Floater

# Milestones

## Front End

* Get Camera input to appear in page
* Get snapshots from camera
* Send snapshots to Backend route
* Display data returned from Backend Route

## Back End

* Route to get data from front end
* Route to send data from back end
* Process image sent from front end
* Train Model
* Make sure hardware can keep up with inputs

## Miscellaneous

* Don’t give up at 3am
* Drink way too much caffeine

# UML Diagrams

