SIGN LANGUAGE PREDICTOR

Software Design Description

By

Alex Athas, Liam Le, Jeffrey Liv, Jeevitha Gudivada, Anushree Ankola,
Vineeth Vasudevan, Elizabeth Tsan

Term: Spring 2018

Under guidance of Prof. Lidia Morrison

CPSC 462- Software Design

Department of Computer Science

California State University, Fullerton

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 Purpose

- From User's perspective: Our project focuses on translating the sign languages, specifically the American Sign Language, acted by the deaf and hard of hearing community to the non-deaf community via a textual representation on the screen of the web application.
- From Developer's perspective: Developing this application helped us learn the language python, used to develop the front-end and was used extensively to build an Artificial Neural Network. This project also helped us understand various concepts of software design such as class diagram, use cases, sequence diagram and so on.

1.2 Scope

The scope of this Software Design Document is to help the developer familiarize themselves with different modules in building a Sign Language Predictor and integrating them after completion. The scope of our project is to implement a web application that translates American Sign language and displays the prediction results as text to the user. The project also fulfills the task of real time data collection from the web camera, connected to the PC. The sign captured from it is sent through the trained model to predict the nature of the sign.

1.3 Overview

The application is designed for people who do not know American Sign Language. Our project fulfills three objectives:

- There are many people who do not know American Sign Language. It is our goal to help others understand what people are saying when they sign. Our application can possibly help a non-signer understand an American Sign Language signer in various situations from asking for direction to answering questions coming from law enforcement.
- To predict the sign with good accuracy on successful upload of a still image/ video capture of the sign the user depicts.
- To train the Neural Network model efficiently to be able to take any kind of image as an input in various settings such as background color, skin tones, pixel size, etc. without much image processing required.

1.4 Reference Material

[1] https://www.tensorflow.org/install/install_linux
[2] https://www.djangoproject.com/
[3] https://codelabs.developers.google.com/codelabs/tensorflow-for poets/?utm_campaign=chrome_series_machinelearning_063016&utm_source=gdev&utm_medium=yt-desc
[4] https://github.com/googlecodelabs/tensorflow-for-poets-2
[5] https://www.youtube.com/watch?v=cSKfRcEDGUs&list=PLOU2XLYxmsIIuiBfYad6rFYQU_jL2ryal&index=6
[6] https://www.reddit.com/r/django/comments/5uczde/tensorflow_with_django/

1.5 Definition and Acronyms

- SDD – Software Development Design
- ANN – Artificial Neural Network
- Training Data – Data to train the Neural Network to predict the future data correctly
- Testing Data – Data to test the Neural Network to get the level of prediction'
- PC – Personal Computer

2. SYSTEM OVERVIEW

2.1 General Overview

American Sign Language translator that helps bridge the gap between signers and non-signers. Our web application takes still images of signs, either by manually uploading them (by users) or captured through the web camera of the American Sign Language alphabet and can return a prediction of the signs in the images. Along with returning the predicted letter, the web application will return the accuracy of the prediction as well as other possible letters it could be and their accuracy. This application has camera interaction where signs can also be predicted, and the corresponding letter will be displayed along with the accuracy of the prediction. The algorithm being used is based on the Inception V3/Mobilenet models and is used to train our data. As training progresses, accuracy increases.

2.1.1 Design Logic and Development Limitation

We provide a highly interactive user interface, developed using Django framework coupled with HTML5 and CSS3. User can access the web application by entering the URL in the browser. From the homepage, the user can press of train model button from the homepage, the model trains using the training set and assigns labels using classification technique of the artificial neural network. The algorithm to train the images is based off a pretrained model developed by Tensorflow for Poets from Codelabs. The algorithm creates a final layer on top of a pretrained layer assigning new features found from our dataset of sign languages. The new features are saved as bottleneck files and separate label file is created for all the images. Finally, when the model is up for training, training accuracy, validation accuracy and cross-entropy for each training step is calculated. It is found that the validation accuracy is lower than training accuracy and the cross-entropy decreases (by backpropagation). This implies the model is training well on the dataset provided. When a new sign is uploaded by user or captured by web camera, it goes through the pretrained model at the back-end and predicts the alphabet on the results page. The percentage of prediction is also displayed on the results page.

2.1.2 Assumptions

- The system must have a web camera connected at all times to capture the still images.
- The system predicts the alphabets well with any dataset.
- Users must have a basic understanding on how to operate the web application.
- The system must have enough memory to incorporate the web application.

3. SYSTEM ARCHITECTURE

3.1 Architectural Design

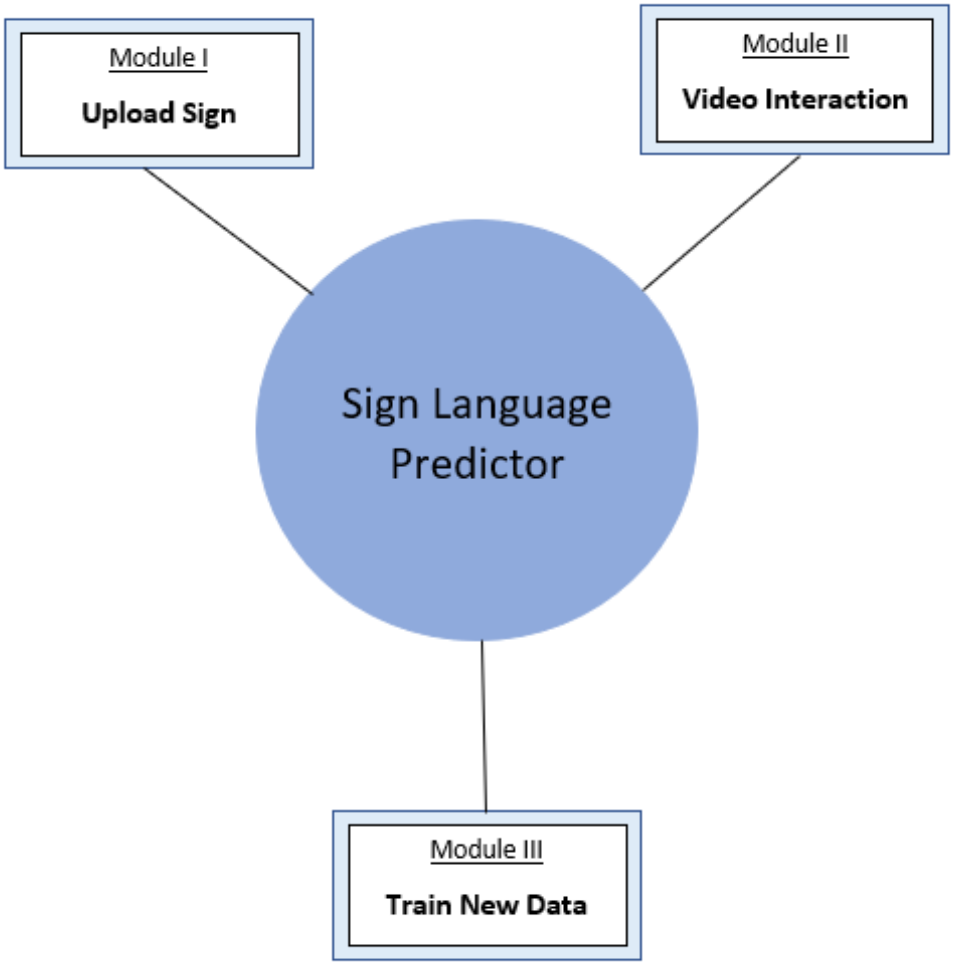

Figure 3.1 Modular Structure Design

The Sign Language Predictor is developed to bridge the gap between signers and non-signers. The web application implemented takes in still images/captures video of the American sign Language alphabet/numbers and returns the prediction results along with accuracy report. It comprises of three main modules. Each module provides prediction of the ASL and displays it to the user in text format. We have the upload static sign module, video interaction module and train new data module.

3.2 Decomposition Description

Below is a brief description of the modules along with their functionality.

Module 1 – Upload Still Sign

This module utilizes upload event feature. In this module, using the upload sign option on the web page the user can select the sign image he wants to predict from the media folder where the sign images are organized according to alphabets and numbers. After successful upload of the image, this module will display the predicted result on result page as text along with accuracy of prediction.

Module 2 – Video Interaction

This module utilizes camera interaction. Using the video option on the webpage the signer can enact the sign that he wants to predict. The camera takes in live feed of the ASL enacted by the signer and the prediction is displayed on the same page with the accuracy of the results.

Module 3 – Train New Data

This module utilizes upload new data event. The module provides option to train new data where the user can upload new signs to our dataset and train the prediction model to obtain better results and accuracy.

4. DATA DESIGN

4.1 Class Diagram

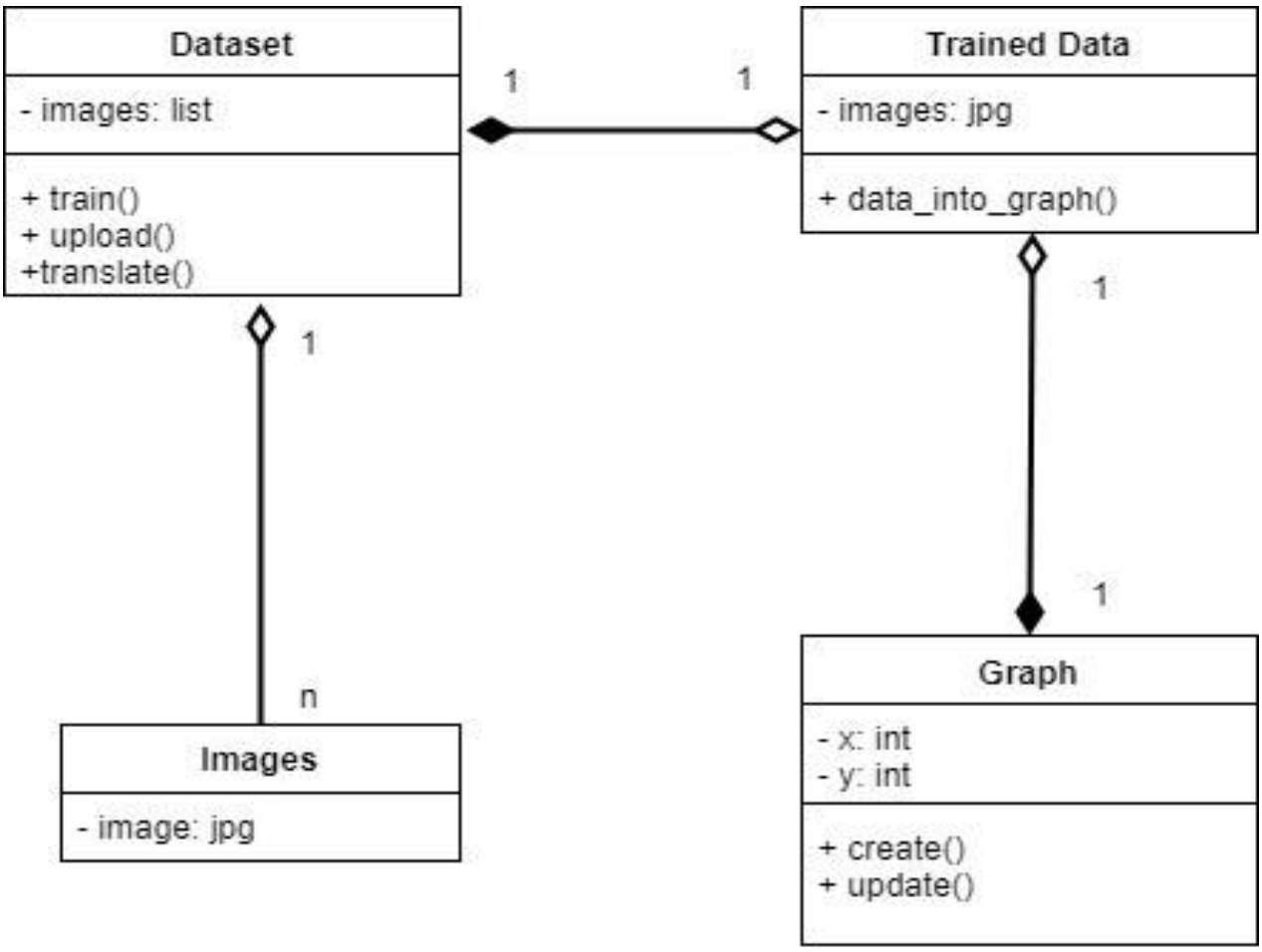

Figure 4.1: Class Diagram for Sign Language Predictor

The Class Diagram helps understand what makes up the main program such as what variables and methods we are using in a more English manner to make it easier to understand. The boxes consist of three sections usually which starts with the name of the class on top, followed by variables, their data types, and a - or a + if they are private or public. The last box is for methods or functions which are used for specific arithmetic, image processing, and training. There are also arrows or diamonds depicting on what class can survive without another class or if they cannot live without one or the other. Numbers near the line show on how many needs either a certain amount of a class or if you can have numerous amounts of a class.

5. HUMAN INTERFACE

5.1 Overview of User Interface

Our web application is designed to provide a friendly graphical user interface so that the users can interact with the system easily. The interface elements used in the webpages include text field and button input controls, navigation bar, tooltips, and file input which are easy to access for facilitating all the necessary functions. We keep the design simple in the way that the user can instantly use the system with the least training support. All the input controls in this application is validated to ensure that the users enter the valid data. For privacy purpose, the system also asks the user for camera access permission to capture the signs, translates those signs, and displays those signs' meaning along with the accuracy reports. For the future data training purpose, the system provides an interface allowing the user to submit the additional images for datasets. From the navigation bar, the users can be promptly directed to other pages such as the home, the tutorial, the about, and the contact pages.

5.2 UI Screenshots

Figure 5.1: The Home Page

Figure 5.2: The About Page

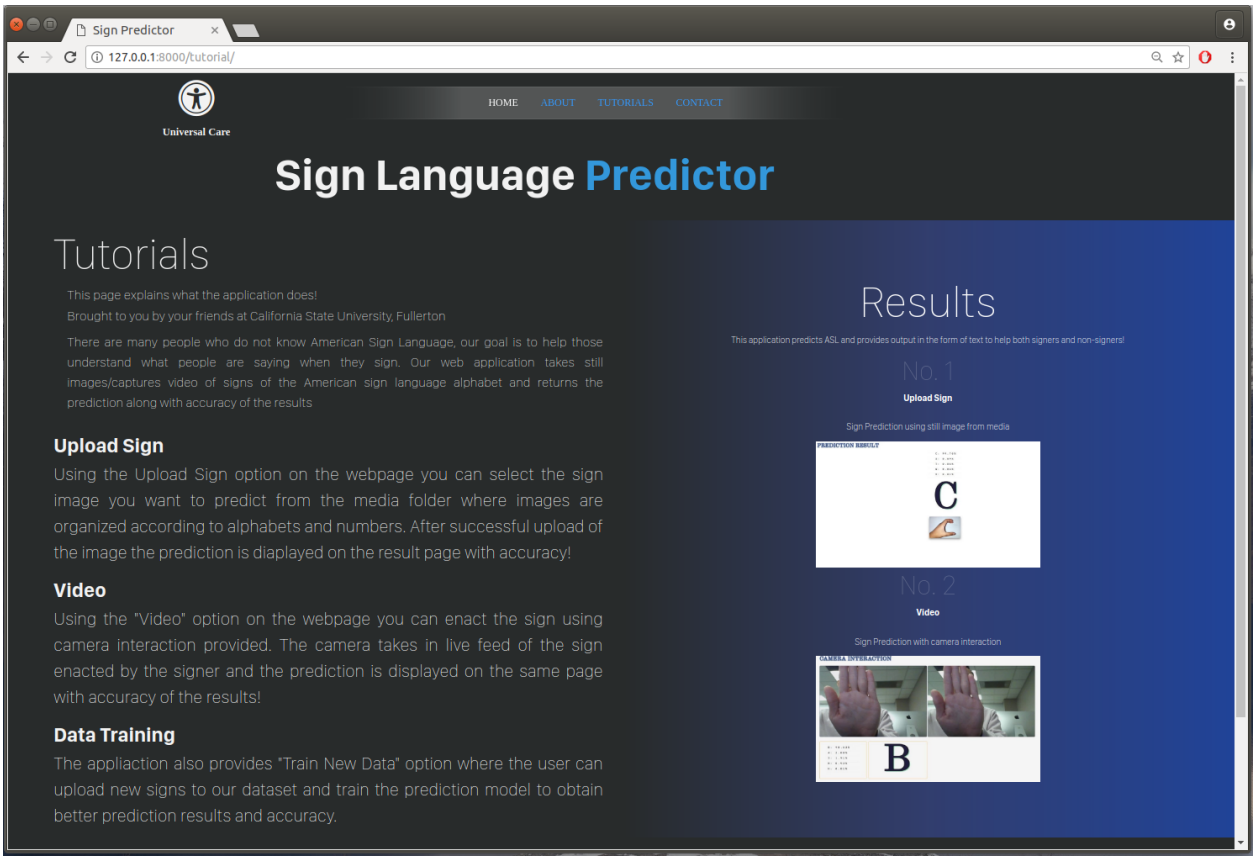

Figure 5.3: The Tutorial Page

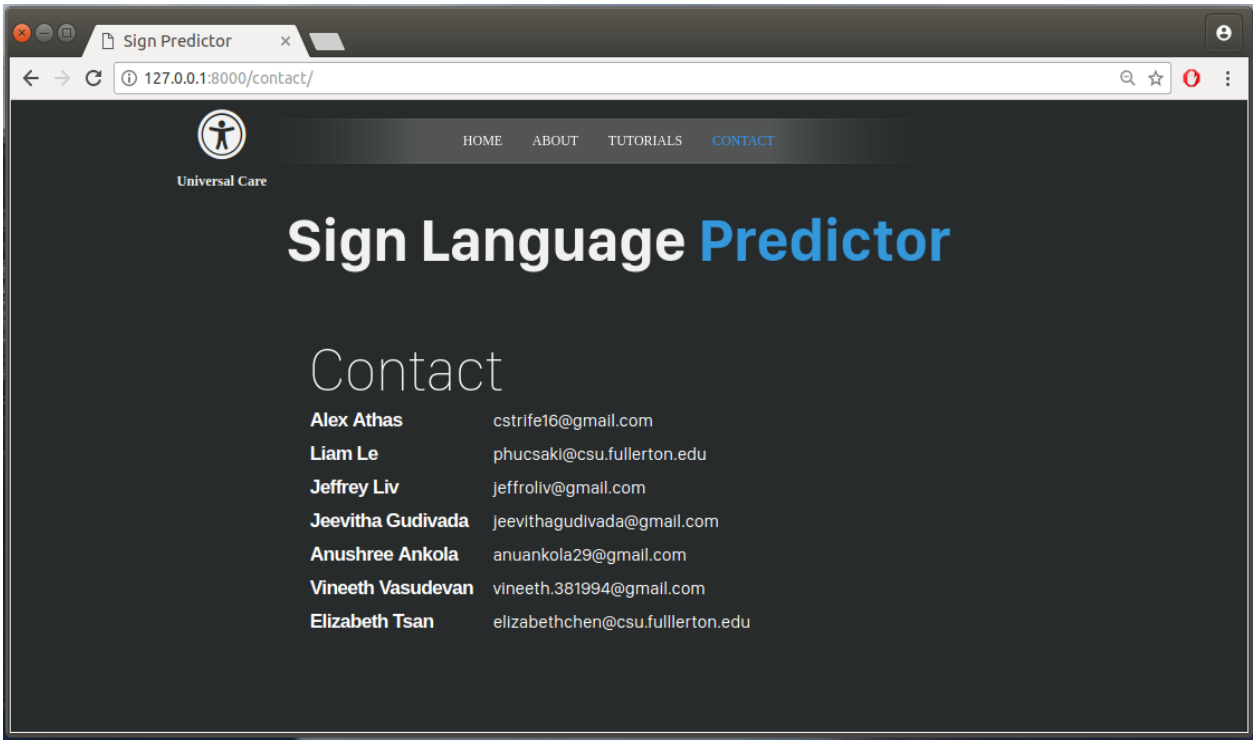

Figure 5.4: The Contact Page

Figure 5.5: The Upload Static Sign Page

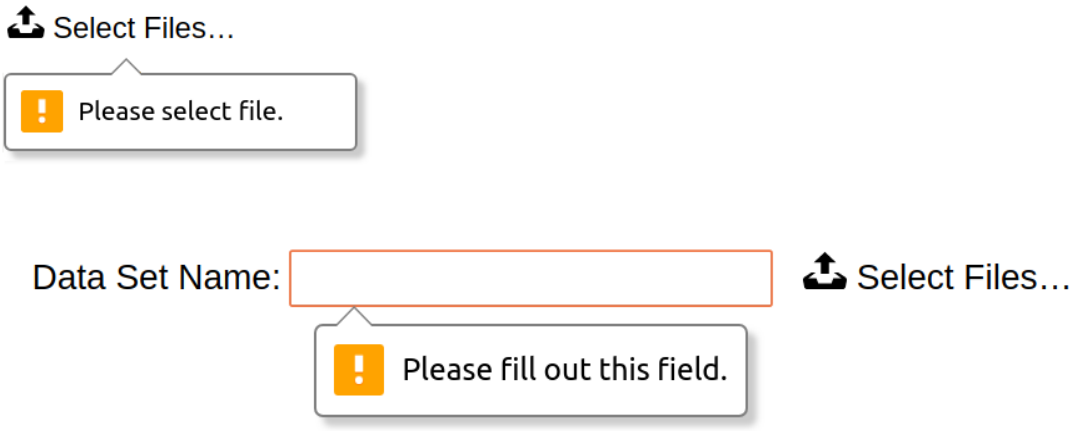

Figure 5.6: Validation Messages

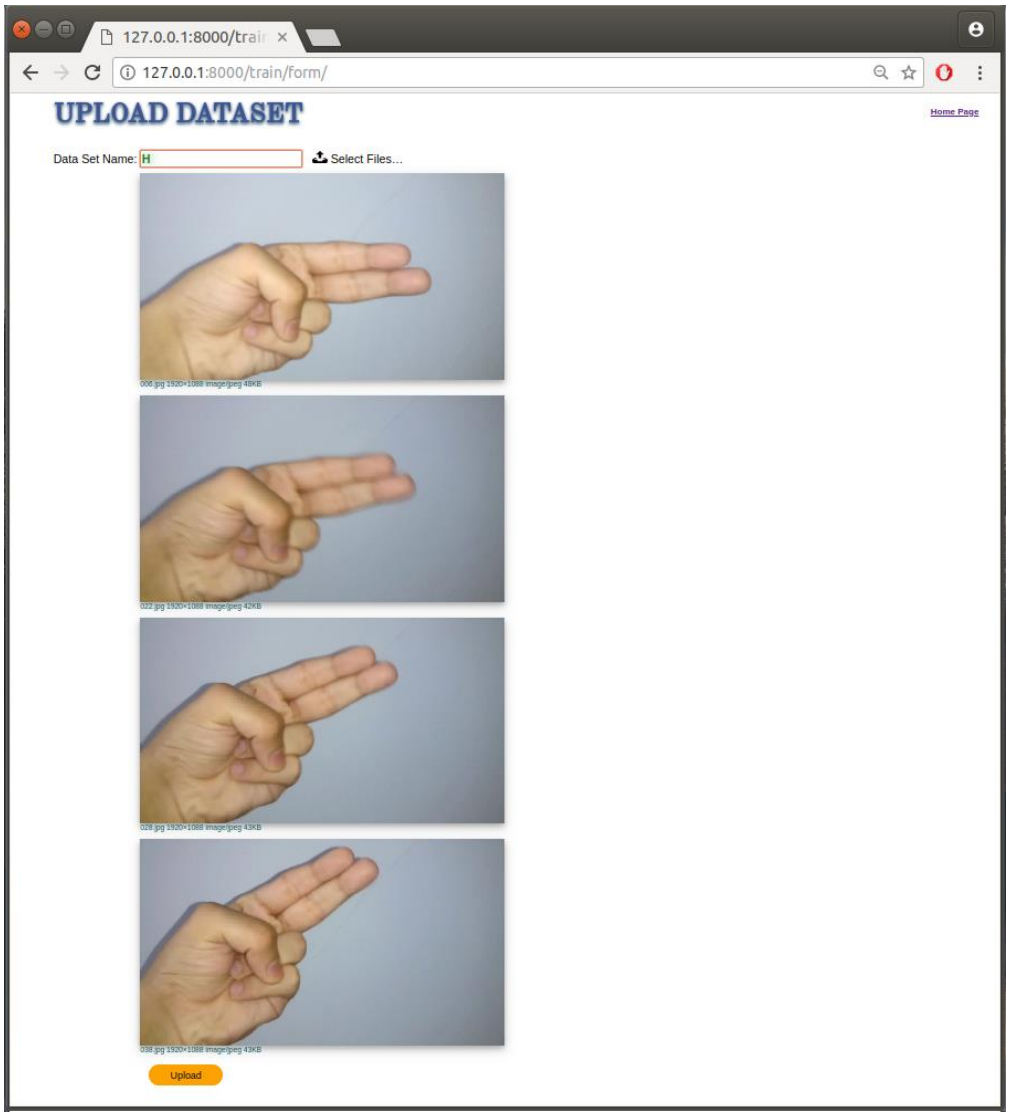

Figure 5.7: Upload Dataset Page

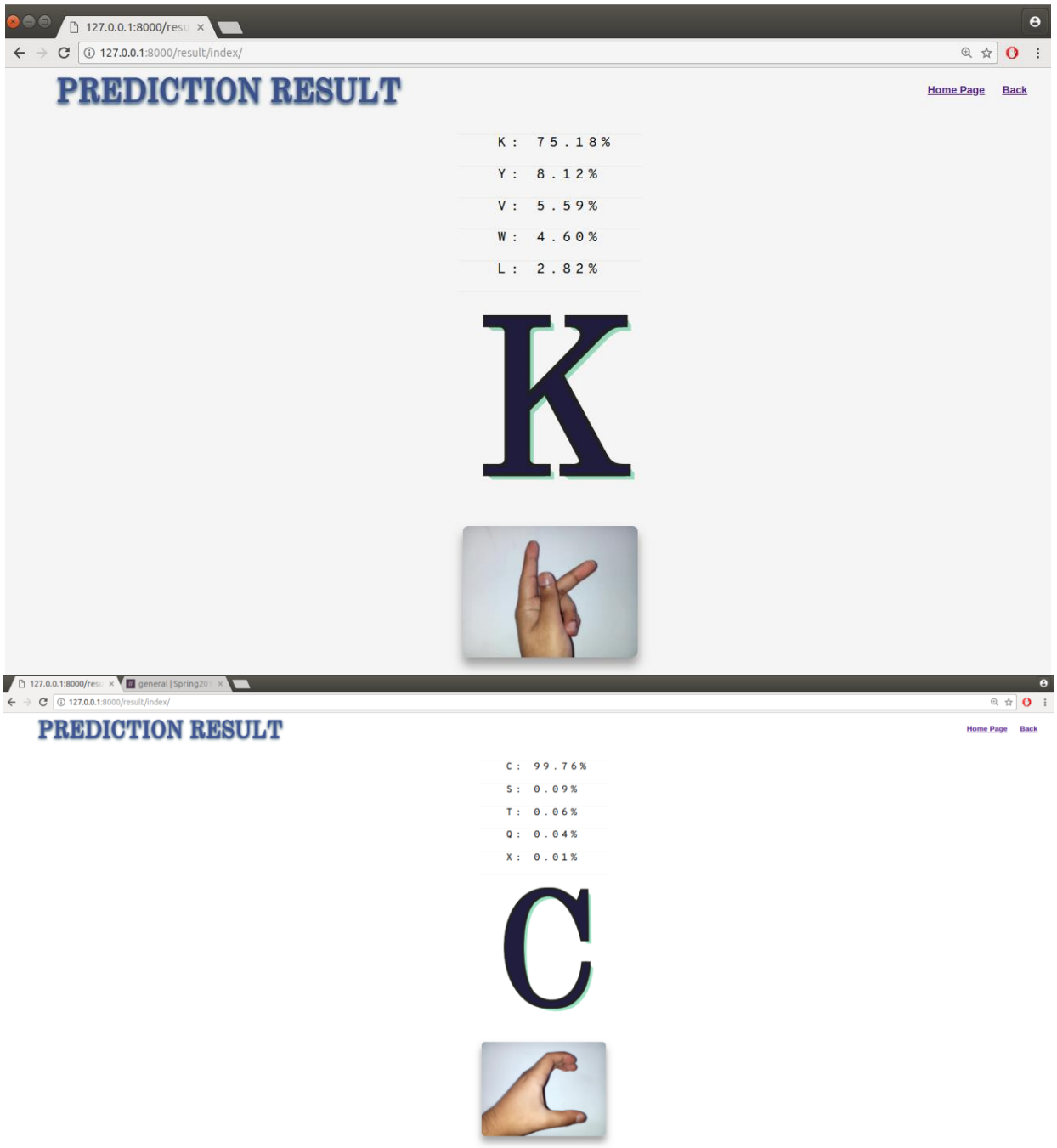

Figure 5.8: Prediction Result and Accuracy Report

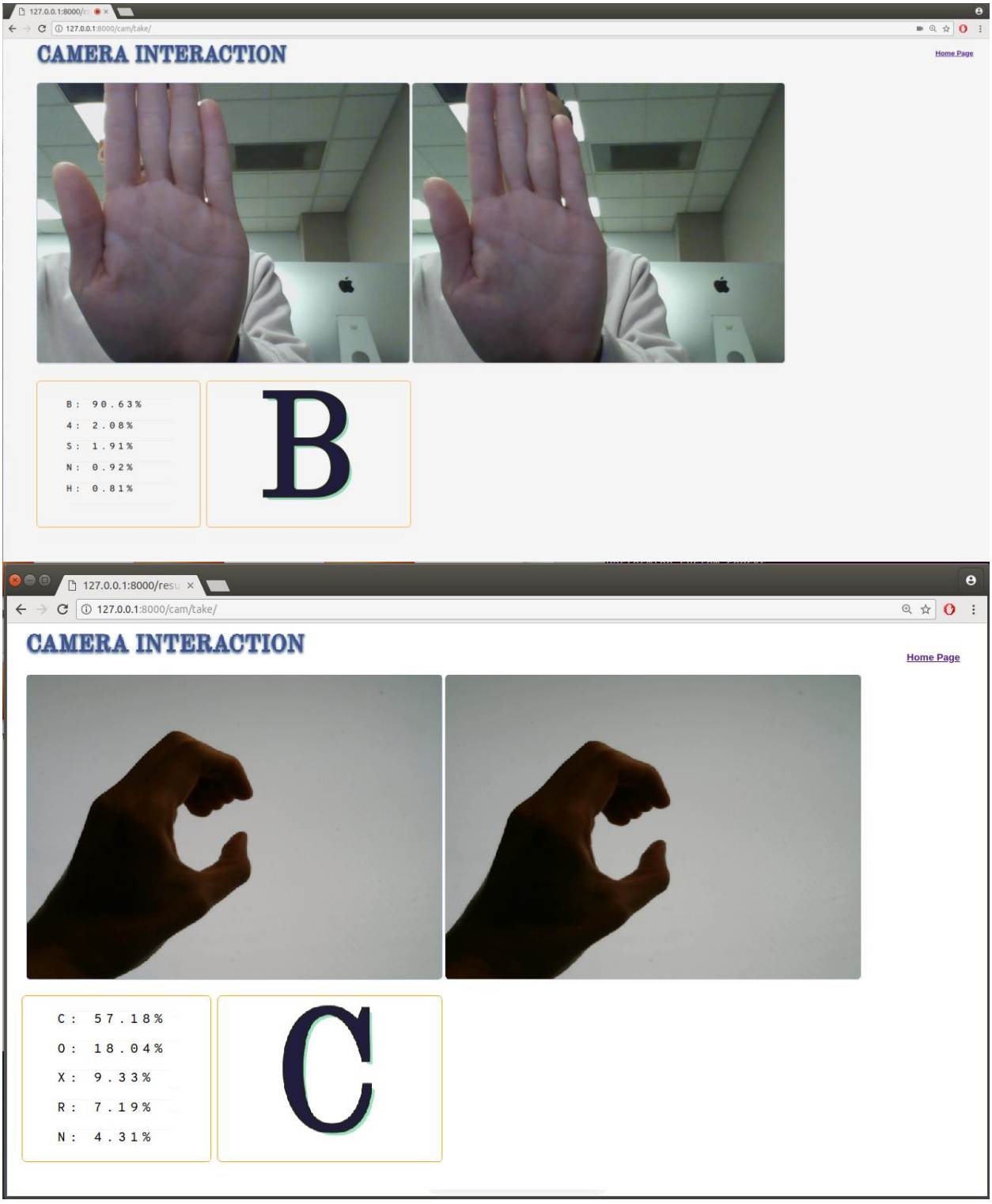

Figure 5.9: The Camera Interaction Page with Prediction Result