Sign Language Recognition Project

Derek Syme

December 2019

The Sign Language Recognition Project was initiated as an attempt to learn and understand the basics and operation of Convolutional Neural Networks. The project was then adapted to make a display attraction for the Glaske hallway of LeTourneau University. The display project is set-up in the Glaske public area for students to view and for exhibition during tours. The project uses a Raspberry Pi embedded computer with an attached webcam to view the hand signals. The camera view as well as the predicted hand signal is output to a seven-inch LED display. Users can see first-hand how a neural network performs in application.

Figure 1: Student Using the Sign Language Recognition Project

The goal of the sign language recognition program is to take a small subset of hand signals and train a neural network to recognize these signs. The hand signals chosen for the project were the numbers zero through nine. The software used a Convolutional Neural network (CNN) with an architecture developed by Russell Thurman with some minor adjustments from the author.

Initial Recognition Program

Based on several general example CNN networks, an initial CNN model was assembled using the libraries Keras and Tensorflow for Python. The network was trained using supervised learning. This means that the training process needs truth data for the network to measure its performance against. For the task of image classification, data set consists of input images and output labels. The initial network was first trained on 500 images resulting in 70% accuracy. It was retrained using 1000 images and achieved 99% accuracy. This accuracy can be deceiving because it was trained using only a single subject with a constant background. If any other person's hand is used or a different background, the network would fail because it was not trained with this variation. Therefore, the network was accurate with the right conditions, but not robust to changing conditions.

Data Collection

Since the network was accurate but not robust, more data was collected to increase the variety and ability of the network. The data set collected was a combination of 1000 labeled images taken of a single user's hand which was used to initially train the first version and over 1000 labeled images that were captured from 50 different hands. A random selection of passing students were asked to record their hand for training data collection. A data collection program was created that acted as a convenient tool for gathering and organizing the data.

Collection Program

The program used for collection consists of a loop that continually captures and displays frames from the connected webcam. The program is also checking for keyboard input. When a key is pressed, it is checked against the list of viable characters. If it is one of the classes that is used for training, the program will save the current frame as the image data, and the key input as the label data. The program will also display text on the screen for a limited amount of time after capturing a data point. For example, after recording an image of a signed two, and recording the number 2 key-stroke, the text overlay on the display will read, “Recorded: 2”.

Data Augmentation

To additionally increase the robustness of the network, random data augmentation can be performed during the training process [32]. This means that on each iteration of training the network is trained with new variation of the data. Random geometric image transformations are applied to the data set before each iteration of training. An augmentation function was created to apply random geometric transformations on the images of the data set. This function is called during each iteration of the network training. The transformations chosen to augment the data are a randomization of horizontal flip, rotation of ±20°, zoom of ±20%, horizontal shift of ±20%, and vertical shift of ±20%. The geometric transformations allow the network to be trained to expect more variation in hand position. The horizontal flip also allows the network to predict left-hand and right-hand signals equally with the same training.

Training/Network Revision

The architecture for the CNN was developed by EE grad student Russell Thurman. When this architecture was loaded onto the Raspberry Pi, it could only run at 1 frame every 20 seconds. This is obviously unacceptable performance. Some performance improvements were made to the design including downsizing the input image size, reducing the number of layers in the network, and reducing

the number of parameters in some of the layers. These improvements, along with increasing the clock speed on the pi, allowed the program to run smoothly with real-time prediction. The final CNN network consists of 13 layers and 1,227,387 trainable parameters and is trained to 94% accuracy over approximately 50 different people. The structure of the network is shown in Figure 2.

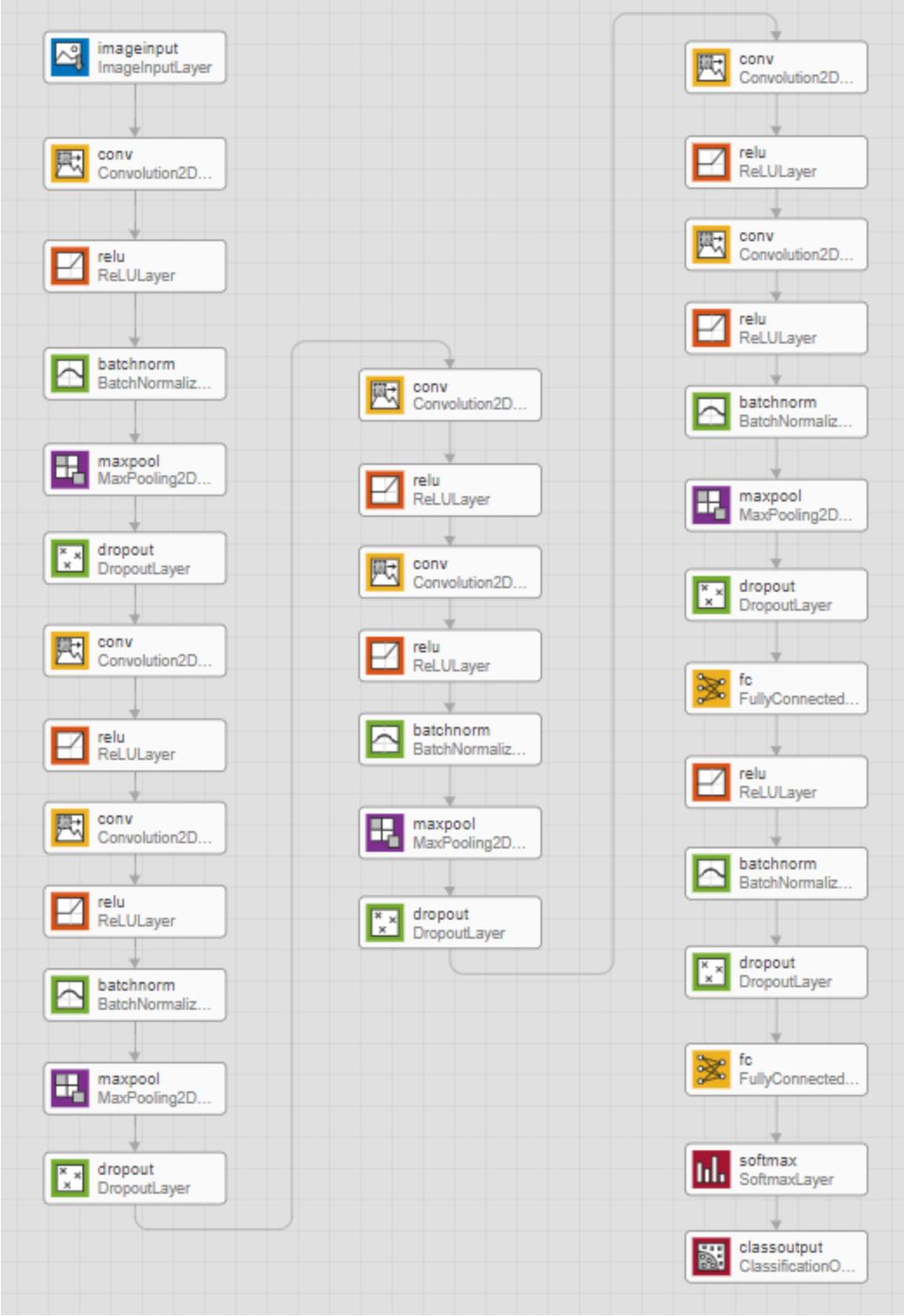

Figure 2: Sign Language Neural Network Structure

Sign Language Recognition Program

The recognition program is designed to use the CNN, trained in the program described in section 0, to recognize a hand signal held in front of a webcam. The program is designed to run on a raspberry pi 3 and display the camera feed as well as the predicted hand signal on a 7-inch LED display. A sleep routine is incorporated into the main code to preserve the CPU of the raspberry pi. The sleep routine is entered when a hand has not been detected for a given amount of time. A simple motion detection algorithm is implemented inside the sleep routine to end the sleep routine when motion is present in the camera view. The program also integrates the GPIO interrupts for power-down of the device.

Figure 3: Screenshot of Sign language Recognition Program