AMERICAN SIGN LANGUAGE RECOGNITON USING CNN

Final Project Report – CMSC 678 (Fall 2017)

Prajwal Gadad (prajwal1@umbc.edu)
Vinayak Appasaheb Bhatte(bhatte1@umbc.edu)

Table of Contents

Abstract

A real-time sign language translator is an important milestone in facilitating communication between the Hearing impaired and the general people. We hereby present the development and implementation of an American Sign Language (ASL) fingerspelling translator based on a Convolutional Neural Network (CCN). We train the model and try to learn the machine using CNN concepts. Once the model is trained we then test the model using an external android app. In this app, we open the camera and test in real time using our own hands. The camera captures the real-time images and is converted in text format. The accuracy for the trained model comes to about 97.3%.

Introduction

American Sign Language (ASL) substantially facilitates communication in hearing impaired. However, there are only ~250,000-500,000 speakers which significantly limits the number of people that they can easily communicate with. The alternative of written communication is cumbersome, impersonal and even impractical when an emergency occurs. To diminish this obstacle and to enable dynamic communication, we present an ASL recognition system that uses Convolutional Neural Networks (CNN) in real time to translate a video of a user's ASL signs into text.

Our problem consists of three tasks to be done:

1. Collecting the ASL datasets for 0-9 digits and A-Z alphabets.
2. Training the model and testing it using these datasets
3. Real-time testing using Android app.

Below image shows the different types of Hand signals from digits 1-9 as well alphabets from letters A-Z.

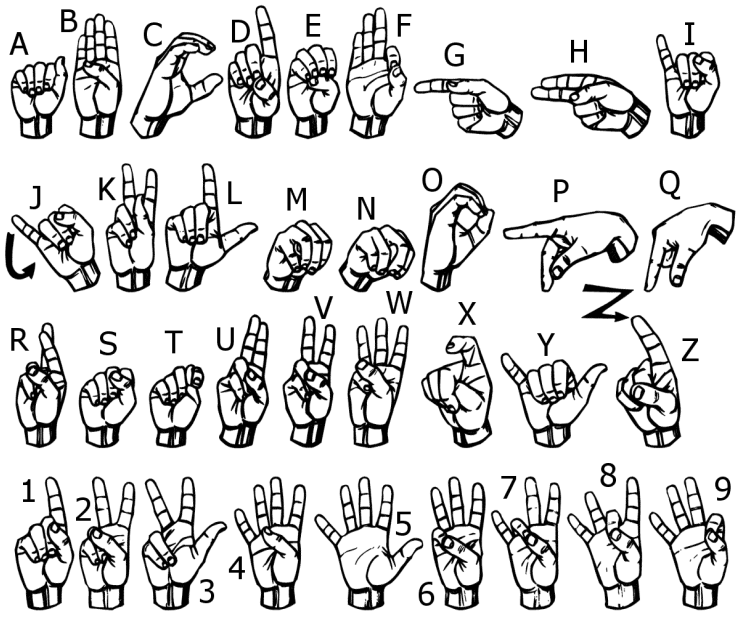


Figure 1: Image showing different types of Hand Signs

This problem represents a significant amount of challenges due to number of considerations, including:

- We had to extract the features from the images
- Dimensionality reduction (e.g. resizing the images to lesser dimensions)
- Choosing the band (e.g. selecting one of the bands from RGB)

While Neural Networks have been applied to ASL letter recognition in the past with accuracies that are consistently over 90%, many of them require a 3-D capture element. The constraints imposed by the extra requirements reduce the scalability and feasibility of these solutions.

Our system trains the model using the existing data sets of ASL signs using the Machine learning concepts. The data are trained using the Convolution Neural Networks. This trained data is then checked for the accuracy using the same datasets. We also try to implement the testing part using an android app, where the app opens the camera and try to capture the real-time images of our hands. These images are converted into the form of texts, which makes the general people readily understand the language.

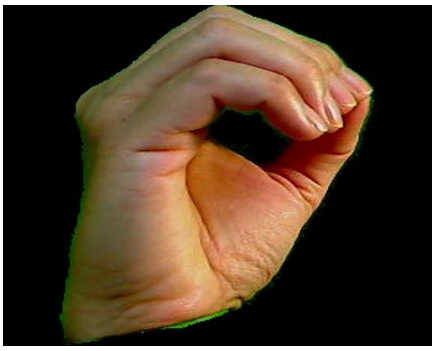
Related Work

ASL recognition is not a new computer vision problem. Over the past two decades, researchers have used classifiers from a variety of categories that we can group roughly into linear classifiers, neural networks and Bayesian networks.

Approach and Methods

Machine Learning

The images are obtained as the datasets from the internet. The images are grouped into digits (0-9) and alphabets (A-Z). Each classification contains about 70 images and of dimensions 400x400 pixels as shown below.



Hand Sign digit '0'

Hand Sign digit '5'

Hand Sign digit '9'


Hand sign letter ‘A’


Hand sign letter ‘G’


Hand sign letter ‘Z’

From the above images, we see that the images have a black background which means that there is no noise is minimal. Also, the images are in RGB pattern with no depth in these pictures. Here, we need to resize the image to have a faster process as the overall matrix dimensionality gets reduced. In this we resize the image using Python code to a dimensionality of 50x50. We also convert the RGB to a single band of blue as shown below.

The rezing factor is done by using Scipy.misc. SciPy (pronounced “Sigh Pie”) is a Python-based ecosystem of open-source software for mathematics, science, and engineering.


Hand Sign digit ‘0’

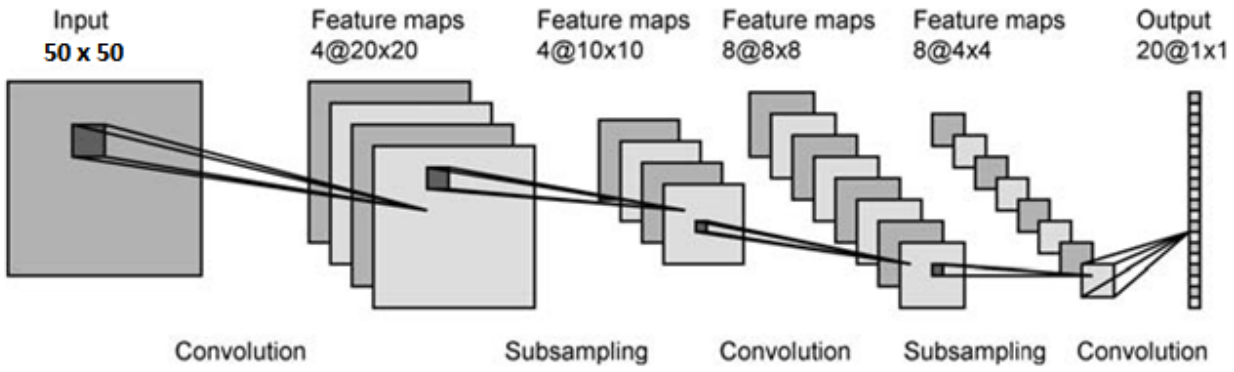
Once we get the reduce dimensional images for all the datasets, we then train the model using this latest dataset. To train the model in Machine Learning we have many algorithms like, K-nearest neighbors in patter recognition, Perceptron for supervised learning in binary classifications, State Vector Machine (SVM) for supervised learning with associated learning, PCA which uses orthogonal transformation using a statistical procedure.

In our approach, as the images are considered, we try to train the model using the concepts of Convolution Neural Networks (CNN). CNN is a is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery.

Convolution Neural Network

When a computer sees an image (takes an image as input), it will see an array of pixel values. Depending on the resolution and size of the image, it will see a 50 x 50 x 1 array of numbers (The 1 refers to RGB values). Each of these numbers is given a value from 0 to 255 which describes the pixel intensity at that point.

CNN are Good for vision problems where inputs have local structure. Shared structure of weights leads to significantly fewer parameters. A CNN consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers and normalization layers. The block diagram for CNN is shown below:



A CNN unit contains the following layers:

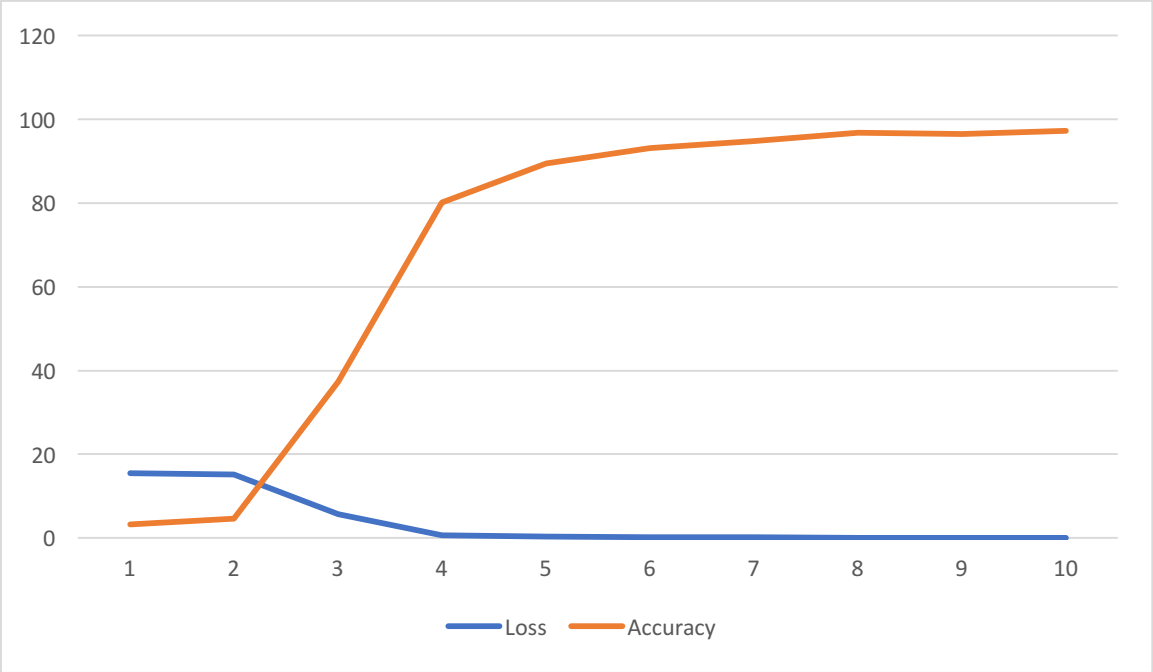
1. Convolution layer containing a set of filters.
2. Pooling Layer
3. Non-linearity

These layers basically take an input volume (whatever the output is of the conv or ReLU or pool layer preceding it) and outputs an N dimensional vector where N is the number of classes that the program must choose from. For example, here we have digit classification as well as alphabets classifications program, N would be 35 since they combine to 35. Each number in this N dimensional vector represents the probability of a certain class. The way this fully connected layer works is that it looks at the output of the previous layer and determines which features most correlate to a specific class.

In our case, programing is done in python. CNN works by importing keras as a library. Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. Keras allows for easy and fast prototyping (through user friendliness, modularity, and extensibility). Supports both convolutional networks and recurrent networks, as well as combinations of the two. Runs seamlessly on CPU and GPU.

Experimental Results

After training the model, we test the model using the same data sets as in training model and predict it. Below figure is the experimental output.



We observe that for initial one epoc, we have very less percent of accuracy whereas the loss percent is high compared to the accuracy. But as we the epoc continues till 4th iteration we see a sudden increase in the accuracy and the loss percent is decreased. This is because the initial epocs tries to understand the algorithm, once it reaches a specific result the percent in accuracy is increased. After 10 epocs we also notice that as we increase the number of epocs we reach the stage of overfitting the model. Hence, we choose 10 epocs. The tabular column below shows the loss and accuracy percentage in detail.

Epoch	**Loss %**	**Accuracy %**
1	15.45	3.23
2	15.18	4.66
3	5.65	37.27
4	0.70	80.11
5	0.35	89.48
6	0.21	93.18
7	0.15	94.74
8	0.09	96.77
9	0.09	96.53
10	0.07	97.24

Conclusion

We collected the datasets in the form of images of American Sign Language. The data of 400x400 was resized to 50x50. We implemented the datasets and trained it using Convolution Neural Network. We are able to produce a robust model for Digits and Alphabets and get a better Accuracy of about 97.24%. We also used the trained model in Android Application.

Future Work

Apart from reading the texts, Voice output in real time can be added to the Android Application. IT can also be combined with NLP to form meaningful sentences from the gestures.

References

1. Real-Time Static Hand Gesture Recognition for American Sign Language (ASL) in Complex Background Jayashree R. Pansare, Shravan H. Gawande, Maya Ingle Vol.3 No.3(2012), Article ID:22132,4 pages DOI:10.4236/jsip.2012.33047.
2. R. Sharma et al. Recognition of Single Handed Sign Language Gestures using Contour Tracing descriptor. Proceedings of the World Congress on Engineering 2013 Vol. II, WCE 2013, July 3 - 5, 2013, London, U.K.
3. Lifeprint.com. American Sign Language (ASL) Manual Alphabet (fingerspelling) 2007.
4. Mitchell, Ross; Young, Travas; Bachleda, Bellamie; Karchmer, Michael (2006). "How Many People Use ASL in the United States?: Why Estimates Need Updating" (PDF). Sign Language Studies (Gallaudet University Press.) 6 (3). ISSN 0302-1475. Retrieved November 27, 2012.
5. M. Jeballi et al. Extension of Hidden Markov Model for Recognizing Large V ocabulary of Sign Language. International Journal of Artificial Intelligence & Applications 4(2); 35-42, 2013