Gesture Recognition on Low Resolution American Sign Language(ASL) Images

Altaful Amin
Department of Biomedical Engineering
Duke University
Durham, NC 27708
aa547@duke.edu

Abstract

American Sign Language is widely used among the Deaf Community in North America. Deep learning techniques can be used to help this community by creating a system that can in real time take an ASL sign and interpret it in real time, bridging the communication gap between ASL users and non-users. In this paper, some preliminary work is done to build a Convolutional Neural Network (CNN) model and test its performance on identifying hand gesture images of letters of the alphabet. The results show that the current model performs at over 85% accuracy on images with various resolutions.

1 Introduction

American Sign Language is the most commonly used mode of communication for the Deaf communities in the United States and Canada. A census count of exact number of users do not exist, but there are approximately 28 million people in the US who are hard of hearing and 2 million of them are deaf [1]. This makes a wide user base, but unlike other natural languages such as English or Spanish, resources are not as widely available yet to make a connection with non-users of the language. A machine learning model that can understand and interpret sign language would be extremely beneficial for this community. The goal is to build a model that can accurately predict gestures from a range of high and low resolution images. This means it can be implemented over a wide range of platforms with various image capture qualities. For the scope of this project, we will use an alphabets in ASL dataset and investigate our model on it.

1.1 Sign Language MNIST dataset

The Sign Language MNIST dataset is the ASL letter database of hand gestures and contain 24 of the 26 letters (J and Z are excluded since they are a dynamic gesture). This dataset was created to closely resemble the original MNIST dataset, which are 28x28 pixel images with 784 total pixels laid out in a CSV format [2]. The labels in the dataset range from (0-25), which maps to the letters (A-Z). There are 27,455 cases of training data and 7172 cases of test data in the set. The data was generated by taking a small set of colored images of the hand gestures and feeding them through a pipeline that cropped, grayscaled, and resized the images to create a bigger sample size and variation. A sample of the original colored images can be seen in Figure 2.

2 Related Work

Similar work was done using Indian Sign Language by grad students in India where they developed a machine learning model to interpret sign language from webcam images [3]. This was a complex

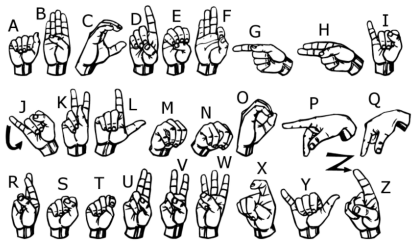


Figure 1: Letters in the American Sign Language

Figure 2: Images in the dataset

model as it involved capturing and extracting a frame from a video and undergo six stages of preprocessing before it gets classified. Images were scaled to 45x45 size, and the classifier was trained using 500 positive, 500 negative and 50 test images [3].

Researchers at the University of North Florida have worked on classifying ASL gestures as captured by a 3D Leap Motion Sensor [4]. A strict algorithm to extract pinch and grab strength and average distance, spread and tri-spread from the sensor data was used and classification was performed using knn and SVM methods [4].

3 Methods

In the first stage of data preprocessing, the test and train labels were separated from the csv datasets. The labels were binarized in one-hot encoding for better classification performance. The pixel values in the dataset range from 0-255, and the values were normalized by dividing by 255. The data needed to be converted into a numpy array for processing. The rows of the dataset were all reshaped into 28x28 pixels to be used by the CNN. In addition, the training data was split into 90% training and 10% validation.

For classification, a keras sequential model is used in this project. At first there are two convolution layers with filter size of 32, 5x5 kernel and ReLU activation. This is followed by maxpool to downsample and dropout to reduce overfitting. Next layer contains two more convolution layers with filter size of 64, 3x3 kernel and ReLU activation. The Flatten and Dense layers get the image ready to be classified to its label. Adam optimizer is being used to optimize the model with a learning rate of 0.001. Categorical crossentropy is used to measure loss and the metric of interest is accuracy.

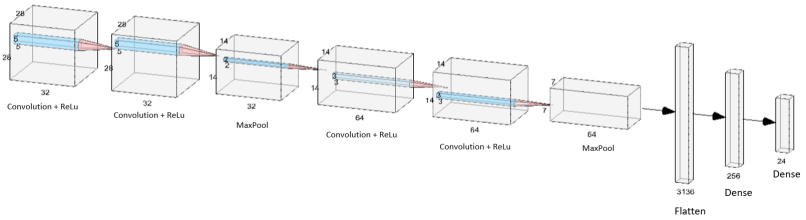
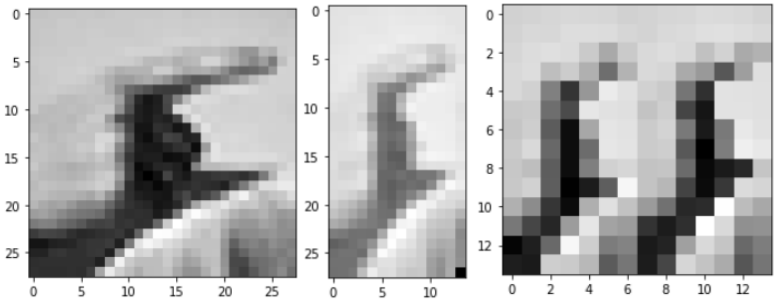


Figure 3: Model architecture

In order to simulate lower resolution images, the method used took the average and combined multiple pixels into one. To generate a 28x14 image, a for loop was created that took the average of two pixels and assigned that as pixel value in a new array. For instance, the average of pixel 1 and pixel 2 became the pixel value for pixel 1 in a new array. This provided an array with 392 pixels, which can be reshaped into 28x14 pixel image. To generate 14x14 images, the same method was followed but this time four pixels were being averaged and stored in a new array. The resulting array consisted of 196 pixels, which can be reshaped into 14x14 images.



(a) 28x28 (b) 28x14 (c) 14x14

Figure 4: Image of the sign for letter C from the dataset with different pixel dimension

4 Results

The model trained for 12 epochs on the training and validation set before making predictions on the test set for each group of resolution. As seen on Table 1, the model performed best on the original image set with accuracy score of 95.74%. On the 28x14 image set, the model performed decently with a accuracy score of 90.26%, however, the loss went up to 0.4679. This can be attributed to the lower number of pixels present in the image to provide the same data. Figure 4(c) shows us a sample image from the 14x14 dataset, and it is observed that averaging and lowering the pixel count caused the hand gesture to replicate twice upon reshaping. Running the model on this dataset yielded a loss of 0.3971 and accuracy of 89.32%. Figures 5 and 6 show us the confusion matrix for actual vs predicted label for 28x28 and 28x14 images. The values around the diagonal reveal to us some of the commonly misclassified letters by the model.

Table 1: Loss and accuracy for each image set

Dimension	Test Loss	Test Accuracy
28x28	0.2257	0.9574
28x14	0.4679	0.9026
14x14	0.3971	0.8932

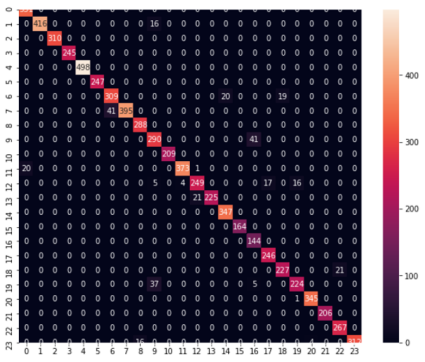


Figure 5: Confusion matrix of actual vs predicted label on 28x28 images

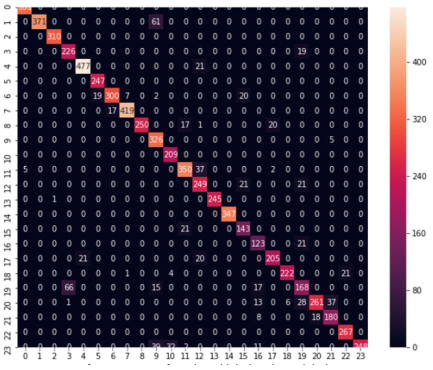


Figure 6: Confusion matrix of actual vs predicted label on 28x14 images

5 Discussion

The letters K and U, K and R, D and U, B and K, and M and N were most commonly misclassified by the model because of their similar hand gesture, as seen on Figure 1. This can be potentially

improved by modifying the brightness contrast of the images during preprocessing.

The double replication issue that was observed with the 14x14 image calls for developing a better method to change the pixel count or resize the image. My lack of python proficiency made it difficult for me to achieve the intended goal. Future work on this will include developing an algorithm that can take in a CSV file and performing rescaling/resizing operations while maintaining the same visual output as the original image.

The CNN model was run for only 12 epochs due to limited processing power being available. Being able to train for more epochs and customizing a optimizer and learning rate that update with each epoch would potentially improve accuracy and lower loss of the model predictions.

References

[1] M. Jay, "American Sign Language," StartASL.com. (2020)

[2] Sign Language MNIST Drop-In Replacement for MNIST for Hand Gesture Recognition Tasks. https://www.kaggle.com/datamunge/sign-language-mnist

[3] K. Dabre and S. Dholay, "Machine Learning Model for Sign Language Interpretation using Webcam Images," 2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA), 317–321 (2014).

[4] C. Chuan, E. Regina and C. Guardino, "American Sign Language Recognition Using Leap Motion Sensor," 2014 13th International Conference on Machine Learning and Applications, 541–544 (2014).