American Sign language

By Ilana Shukhman, Omer Rugi and Lilach Mor.

Introduction

Sign language is a visual language that consists of specific moves, hand signals, and various facial expressions.

With the development of technology, software tools become usable for identification of sign language and various software products are produced.

Object detection is a process of detecting real-world objects in images and videos.

. Like many other object detection and recognition problems, deep learning delivered solution for sign language recognition.

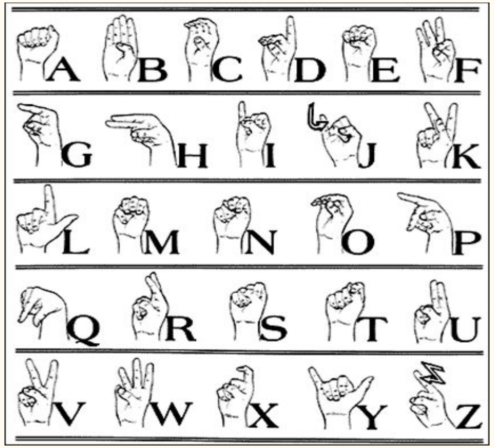
The purpose of this work is to use deep learning to identify sign language characters from pictures. To do so, we will use different approaches, such as logistic regression, MLP and CNN.

The dataset

We took the data from Kaggle, Sign Language MNIST.

The American Sign Language letter database of hand gestures represents a multi-class problem with classes of letters (excluding J and Z which require motion).
The labels in the data represent each of the letters (and no cases for J or because of gesture motions).
The data is divided into a training set and the test set.

There are 27,455 cases of training data and 7,172 cases of test data, header row of which represent a single
pixel image with grayscale values between 0-255.

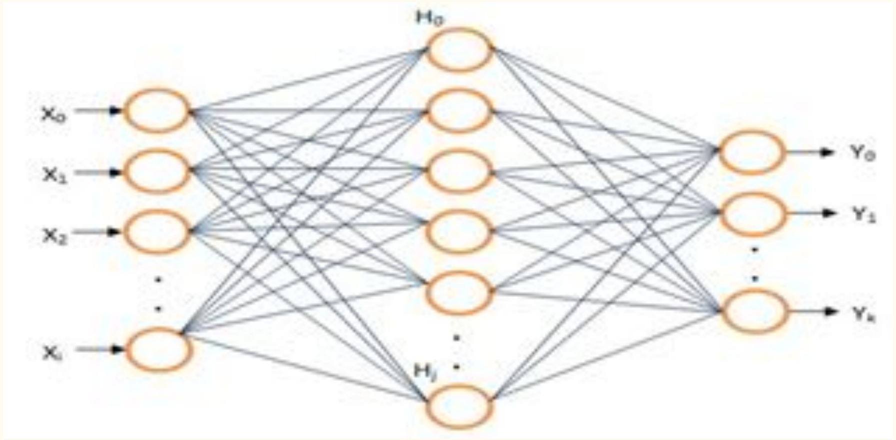


First attempt - Logistic regression

As we learned at class, we computed the final result using Soft Max. For the loss function we use the cross entropy function, and updated our model with the AdamOptimizer function that minimize our loss function.

Second attempt - MLP

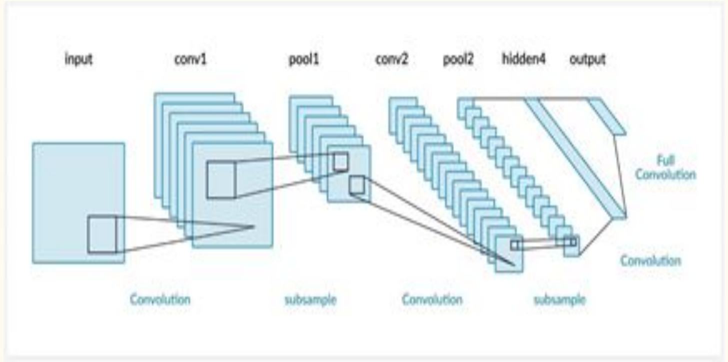
As for the Neural Network, we used the same learning rate, optimizer function and loss function, with additional two hidden layers - each of them with 512 neurons, and between the layers we used the relu function as the activation function.



CNN

In our CNN we have 3 hidden convolution layers: in each of the layers we did batch normalization, and between every two layers we did maxpool. The first layer contains 75 kernels in size 28X28, the second layer contains 50 kernels in size 14X14, with dropout of 20 percent, and the third one contains 25 kernels in size of 7X7.

After the last maxpool, we flattened the matrix we got from the CNN part to get a hidden layer with 400 neurons.
From this layer we used the relu function to get to the other hidden layer with 512 neurons and dropout of 30 percent. In the end, we used SoftMax to output the final results.



Results

After running all of the three models, we got the following results:

The first model was the logistic regression. We got that the test accuracy is 66%, and the loss of the train is 0.747. When we used the Gradient descent function to minimize our loss function, we got to the same results after 2,000 iterations - therefore, we decided to use the Adam Optimizer function.

The second model was the neural network. In this case, the test accuracy was 81%, and the loss of the train was 0.0023, which is much better than the first model.

The third model was the CNN and was the best out of the three - the test accuracy reached 97%, and the loss of the train was 0.0105.

Results

Model	Test accuracy	Train loss
Logistic Regression	66%	0.747
MLP	81%	0.0023
CNN	**97%**	**0.0105**

Conclusion

The fundamental problem with the data we used is that the letters J and Z cannot be described using a picture, because they require motion. However, with the remaining letters we got a result of 97% accuracy, which is very good. If we were to find a way to distinguish between the letters that require motion and those that don’t, we could achieve a similar result that includes all the letters.

The high result that we got shows us that we are not far from a solution for the communication problem between the hearing people and the non-hearing people.