ASL Letter Detection Using Machine Learning

GitHub Link: https://github.com/Mnikito/4106/tree/main/ASL_Recognition

Gandhar Deshpande
M.S Student
UNC Charlotte
Charlotte, North Carolina
Email: gdeshpa1@uncc.edu

Mikhail Nikitin
Junior Student
UNC Charlotte
Charlotte, North Carolina
Email: mnikitin@uncc.edu

Shruthi Pasumarthi
M.S Student
UNC Charlotte
Charlotte, North Carolina
Email: spasuma5@uncc.edu

Trevor Goad
Senior Student
UNC Charlotte
Charlotte, North Carolina
Email: wgoad@uncc.edu

***Abstract*—There is a significant population of the world who have hearing disabilities. There is a lot of effort from the tech companies in the world who are trying to bridge the gap in between the ASL Language and Spoken English. This project uses Deep Learning to translate the ASL letter into the corresponding English Letter.**

***Index Terms*—ASL, Deep Learning, Image Classification**

I. Introduction and Motivation

This Project uses the American Sign Language as the input and it translates it to the corresponding English Letter. There are so many people in the world who find it very difficult to communicate with offices for their needs because they have hearing difficulties, and not a lot of us are literate in the Sign Languages. These communications are often misinterpreted as agitated people trying to disturb offices. In some cases, luckily there are a few staff who understand the Language and can help, otherwise a lot of people who have hearing disabilities return back frustrated and upset that their work cannot be done and that they have been falsely accused.

II. Approach

The project uses Deep Learning to translate the ASL Letters into the corresponding. English Letters. We have used three models MobileNet V2, Resnet 50, ResNet 9. Each of the group members picked up one network and trained the dataset. Each network gave a different accuracy on the same dataset. We used the ASL Kaggle dataset which has 87000 training input images which are of the size 200x200. These images are RGB images. In the code, we resized those images. We used different platforms, Google Collab, CPU and an NVIDIA GPU.

III. DataSet and Training Setup

The dataset we used is the Kaggle Dataset. This contains 87000 training images. It has already been separated into training dataset and a test dataset. The training dataset has 3000 images in each class and a total of 29 classes, 26 letters and the test dataset has 29 images, one image from each class. There are 26 alphabet classes, A to Z and three extra classes, space, nothing and delete.

For the training, our group picked up different parts of the project. One of us ran Resnet - 9, another person ran MobileNet-V2, another person ran ResNet-50 on the dataset. The fourth person took all the saved models and tested them on the test dataset and worked on the practical deployment of the model.

For diversity, we wanted to try out the models on different platforms, for that reason, one of our groupmates ran the whole code on Google Collab, one of us on an NVIDIA GPU installed in the laptop, and one of us used the CPU. The training time had a very significant difference when compared to each other.

IV. Training Details

A. MobileNet-V2

The MobileNet-V2 is a model developed by Google. We chose this model because of its small size and high speed. This network is used in many devices because of its usage of the lightweight depthwise convolutional layers. This network is used in several small devices such as mobile phones.

For the size of MobileNet-V2, the number of parameters and MACs is very low, this is what makes it suitable for smaller devices. The network is based on a bottleneck structure wherein the residual connections are in between the bottleneck layers. [5]

For the training of this network, the 200x200 input images from Kaggle were resized to 224x224. The training details using the MobileNet-V2 are as below:

Training Time:	1hr 4 minutes
Epochs:	2
GPU:	NVIDIA Tesla (Google Colab)
Training Accuracy:	99%
Validation Accuracy:	97%
Number of Parameters:	1.24M
Number of MACs:	318.99M

B. ResNet

In deeper networks, the performance was worse than a shallow network. This was because of the vanishing gradient problem, wherein the gradient is so small, that the update to the value is not possible [6]. This prevents the improvement of training. In the ResNet, the concept of the skip connections also known as the residual connections were introduced. The ResNet solved the problem of vanishing gradient.

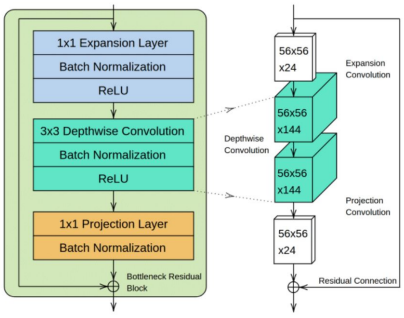

Fig. 1: Architecture of the MobileNet-V2 [1]

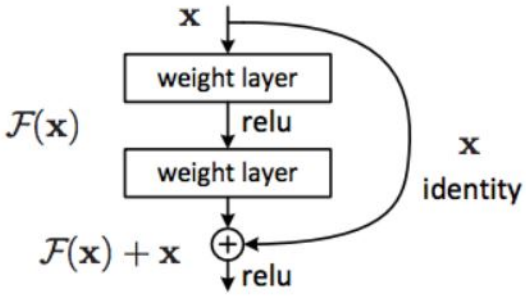

Fig. 2: The ResBlock [4]

1) ResNet-50: The ResNet-50 has 50 such ResBlocks, however, the residual connections go over 3 layers rather than two layers. This feature makes it a little different from the other ResNet networks. There are a lot of different layers that make up the ResNet-50 model, convolutional layers, depth-wise convolutional layers, and 1x1 convolutional layers which reduce the feature map size across the channels. [3]

The ResNet-50 gives a very high accuracy and performance for a network of that size due to the skip connections. However, the number of parameters and the number of MACs make it a very heavy model to deploy.

Training Time:	2hr 37 minutes
Epochs:	50
GPU:	NVIDIA GTX 1650
Training Accuracy:	87%
Validation Accuracy:	88%
Number of Parameters:	25.56M
Number of MACs:	0.34G

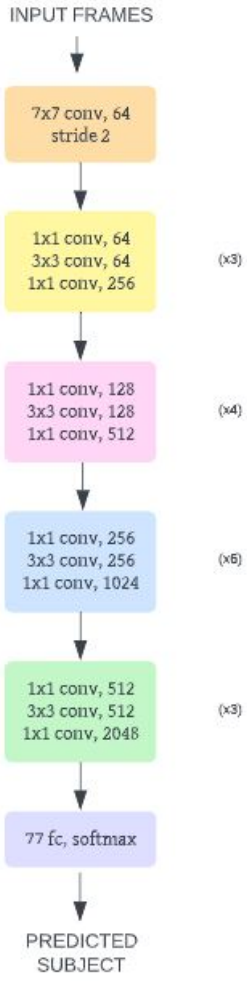

Fig. 3: ResNet-50 Architecture [2]

2) ResNet-9: The ResNet-9 is a smaller version of the ResNet-50 but has a very similar architecture as the ResNet-50. The ResNet-9 due to the smaller size, has fewer parameters and lesser number of MACs, however not enough to be deployed. The optimizer used in the training of the ASL dataset is the Adam optimizer.

Training Time:	8hr 35 minutes
Epochs:	7
GPU:	Nil- CPU
Training Accuracy:	100%
Validation Accuracy:	100%
Number of Parameters:	6.63M
Number of MACs:	1.52G

V. Comparison

Comparing all the three models and their performances, we felt that the MobileNet-V2 was better. This is because despite the less number of parameters, it achieves a noticeable accuracy. This model can be deployed onto many small devices

due to its small size, hence we chose this model to further our research.

ResNet-50 due to its large size and high number of parameters was not used, but it being one of the most popular networks, we wanted to train our dataset on the network and see the performance.

Although ResNet-9 gave a 100% validation accuracy, it seemed a little suspicious.

VI. Practical Deployment

To practically test out this project, we chose to do a real-time feed of the sign. Any person can sign to the camera, which will click a picture and the model will recognize the sign and translate it into the corresponding English Letters. We used the MobileNet-V2.

At the beginning, using the network did not give very accurate results. However, once there were a few adjustments done to the input, the code gave much better results compared to the previous try. In the code, the data was augmented such that the pixels in the image were inverted- when the background of the input image was lighter an the hand was darker, the image was augmented such that the background became darker and the hand signing consisted of lighter pixels.

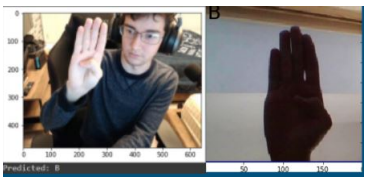

Fig. 4: Prediction of the Letter B

As shown in figure 4, the model was able to correctly predict the letter signed due to the augmentation done in the code.

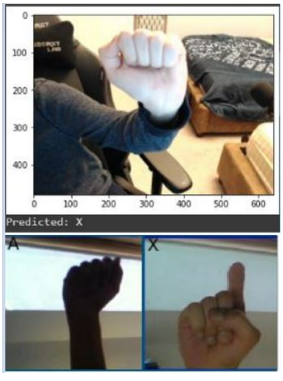

Fig. 5: Prediction of the Letter A

In figure 5, the letter shown is not correctly detected, this maybe because both the letters, A and X resemble each other very closely. The model may have gotten confused between the two signs.

VII. Future Works

This project can be developed into many different ways.

This model can be further developed and deployed onto a microcontroller. When a person signs at the camera of the microcontroller, the letter is translated into the corresponding English letter, making it easy for the official or the employee of the store to understand the person.

Instead of a microcontroller, this model can be developed into an application, to be used on a mobile device which is possible because of the small size of the MobileNet-V2. A person can sign into the camera connected to the device which is sent to the application as the input. The application can detect the letter and translate it into the corresponding English letter.

The training dataset can be expanded by adding several other languages such the British Sign Language, the French Sign Language, the Italian Sign Language, the Japanese Language etc. This can generalize the model as a person can sign in any sign language which he is most comfortable with and the model detects it and translates it for the listener.

The model we used for our training consists of only alphabets. The model can be trained on words also instead of only alphabets. This will definitely make conversations easier and shorter as the person can sign complete words rather that spell out each and every word to the camera or the microcontroller. The model in this case will actually behave as a translator from the sign language into English.

VIII. What We have Learned

While choosing a network to train, we should keep the application of the project in mind. Based on that only can we choose the model. In our case, for the project, even though we trained the model on three different models, we chose the MobileNet-V2 due to the number of parameters it has and its storage size.

We must be aware of the dataset we are training our model on. Our training dataset only covered the alphabets, and the due to the background of the training images, we had to augment our code in order to get the results we were expecting.

IX. Conclusions

In this project, we learned that we need to look at different parameters of a network and see what suits our application the best. The network we choose does not need to have a very high accuracy, there are other parameters we need to look into while selecting the model, such as the storage size. We also realized that the biggest model does not always have the highest accuracy. One unexpected experience we faced that even the model with a 100% can look fishy and suspicious.

Working on this project made us realize that the potential of Deep Learning and Machine Learning is so much, that it can not only be used in tech and development but also can be used to solve so many social problems and bridge so many social issues.

REFERENCES

[1] Surinta Enkvetchakul. Effective data augmentation and training techniques for improving deep learning in plant leaf disease recognition. 2022.

[2] Mohammad Jahromi, Pau Buch-Cardona, Egils Avots, Kamal Nasrollahi, Sergio Escalera, Thomas Moeslund, and Gholamreza Anbarjafari. Privacy-constrained biometric system for non-cooperative users. *Entropy*, page 1033, 2019.

[3] Aakash Kaushik. Understanding resnet50 architecture.

[4] Connor Shorten. Introduction to resnets, 2019.

[5] SMITA SINHA. Why google's mobilenetv2 is a revolutionary next gen on-device computer vision network, 2018.

[6] WikiPedia. Vanishing gradient problem.