American Sign Language Classifier: An Application of Convolutional Neural Network Machine Learning Models using Sequential and VGG16 Architecture

Adarsh Makarla, John Hudson, Yousif Said, Tirth Patel

Abstract: This report discusses the development of an American Sign Language (ASL) classifier using Convolutional Neural Network (CNN) machine learning models, specifically the Sequential Model and VGG16 architecture. The goal was to accurately classify an ASL image into one of 29 categories, those being A-Z letters and special classes for SPACE, DELETE, and NOTHING. The Sequential Model employed Keras ImageDataGenerator for image augmentation and preprocessing. The VGG16 Model which has been pretrained on ImageNet, and was fine-tuned for ASL alphabet recognition. Both models' performances were evaluated using recall, precision and F1 scores. Plots such as loss graphs, accuracies and confusion matrices showcase insights into the model performance and were provided. The overall findings showcase the VGG16 Model performed better than Sequential regarding testing accuracy.

Key Terms: Convolutional Neural Network, Sequential Model, VGG16, Neural Network

Introduction

At the start of this project, we intended to do an American Sign Language detection model for our CSE5717 project. The goal was to decipher different letters of the American Sign Language Alphabet and accurately classify them into 29 categories. We tested multiple classifiers and identified which yields us the highest percentage in regards to accuracy, precision and recall. We came to the conclusion that the Sequential Model and VGG16 were the best possible choices with Sequential Model yielding 98.5% and VGG16 yielding 99.9% accuracy.

Data Set Description

Our dataset consists of 87,000 images which are 200x200 pixels of the ASL alphabet. There are 29 classes, of which 26 are for the letter A-Z and 3 classes for SPACE, DELETE and NOTHING. The images offer variety, some have different angles, lighting, skin tone, etc. This will prevent biases forming when training the model.

Dataset link: https://www.kaggle.com/datasets/grassknoted/asl-alphabet/data

Data Preparation

The data was split into 70% training, 15% validation, and 15% for testing. We ensured the shape of the images was 200x200 pixels, where each pixel has an RGB value to account for the color in the images as another dimension.

Sequential Model

The Sequential Model uses the Keras ImageDataGenerator to prepare the images for the training of the CNN model. This entails rescaling of the pixel values and allows for generating batches of augmented images during training. The architecture consists of convolutional layers, max-pooling layers, dropout layers, and dense layers. The model is compiled with the Adam optimizer and sparse categorical cross entropy loss all of which help multiclass classification.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 198, 198, 32)      896

 max_pooling2d_3 (MaxPoolin  (None, 99, 99, 32)        0
 g2D)

 dropout_4 (Dropout)         (None, 99, 99, 32)        0

 conv2d_4 (Conv2D)           (None, 97, 97, 64)        18496

 max_pooling2d_4 (MaxPoolin  (None, 48, 48, 64)        0
 g2D)

 dropout_5 (Dropout)         (None, 48, 48, 64)        0

 conv2d_5 (Conv2D)           (None, 46, 46, 128)       73856

 max_pooling2d_5 (MaxPoolin  (None, 23, 23, 128)       0
 g2D)

 dropout_6 (Dropout)         (None, 23, 23, 128)       0

 flatten_1 (Flatten)         (None, 67712)             0

 dense_2 (Dense)             (None, 256)               17334528

 dropout_7 (Dropout)         (None, 256)               0

 dense_3 (Dense)             (None, 29)                7453

=================================================================
Total params: 17435229 (66.51 MB)
Trainable params: 17435229 (66.51 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

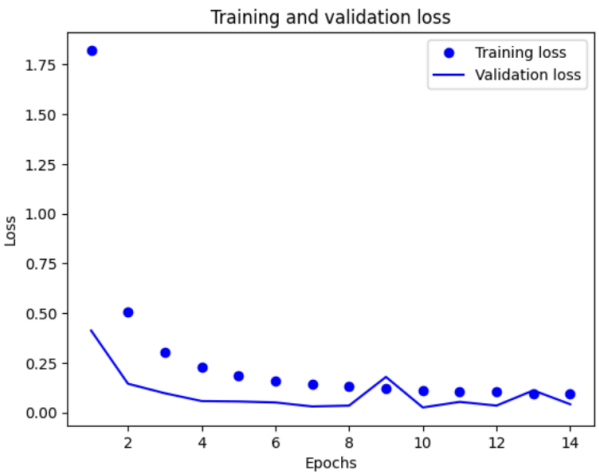
Model Summary

This model was trained with 20 epochs and early stopping set to 4. With these settings we are able to achieve

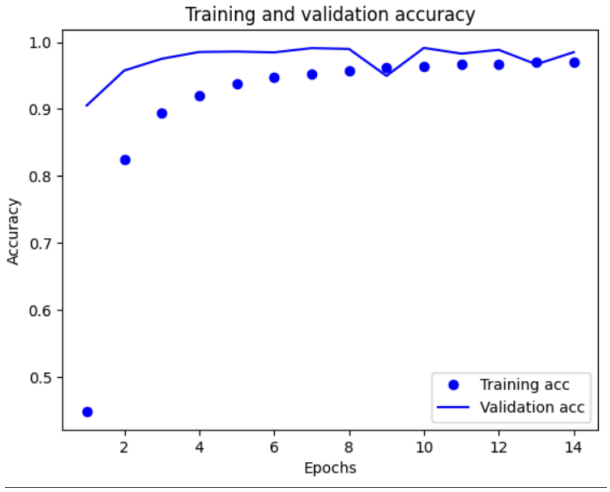
Training Accuracy: 97.01%
Validation Accuracy: 98.51%
Recall Score: 98.6%
Precision Score: 98.5%
F1 Score: 98.5%

We then utilized plots for further insight on the model and its performance

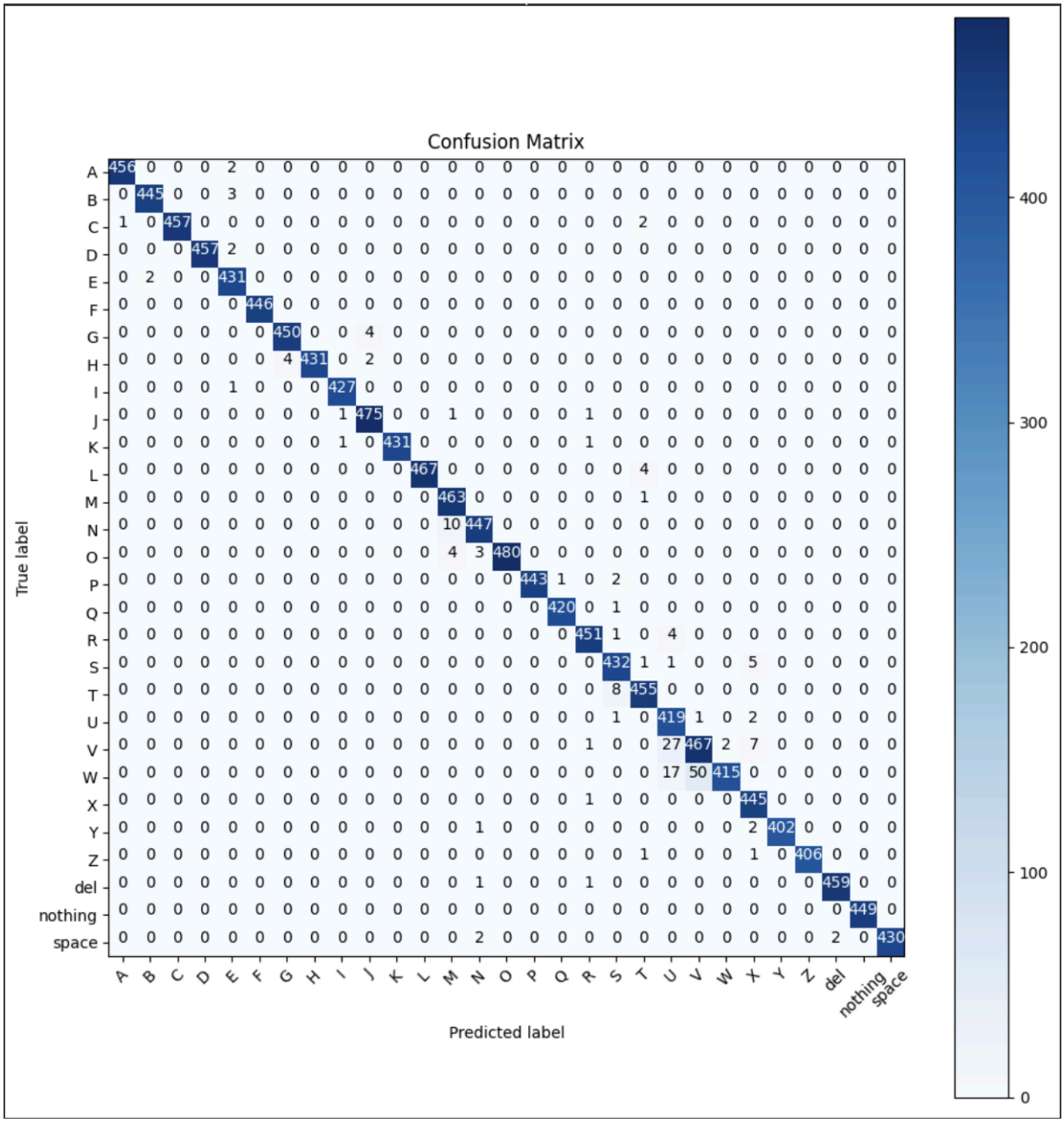
Loss for Training and Validation



Accuracies for Training and Validation



Confusion Matrix



With all these settings we yielded a **98.51%** testing accuracy.

VGG16 Model

After completing the Sequential model, we thought we would be able to get better numbers. So after some research, Big Data lectures, and meetings with Wei Wei we settled on using the VGG16 model. The VGG16 Model, which is pre-trained on ImageNet, was loaded and modified for ASL alphabet recognition. Additional layers, including Flatten, Dense, and Dropout are added to the VGG16 base to tailor it to our needs for classification. The model is compiled using the Adam optimizer and categorical cross entropy loss.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 200, 200, 3)]	0
block1_conv1 (Conv2D)	(None, 200, 200, 64)	1792
block1_conv2 (Conv2D)	(None, 200, 200, 64)	36928
block1_pool (MaxPooling2D)	(None, 100, 100, 64)	0
block2_conv1 (Conv2D)	(None, 100, 100, 128)	73856
block2_conv2 (Conv2D)	(None, 100, 100, 128)	147584
block2_pool (MaxPooling2D)	(None, 50, 50, 128)	0
block3_conv1 (Conv2D)	(None, 50, 50, 256)	295168
block3_conv2 (Conv2D)	(None, 50, 50, 256)	590080
block3_conv3 (Conv2D)	(None, 50, 50, 256)	590080
block3_pool (MaxPooling2D)	(None, 25, 25, 256)	0
block4_conv1 (Conv2D)	(None, 25, 25, 512)	1180160
block4_conv2 (Conv2D)	(None, 25, 25, 512)	2359808
block4_conv3 (Conv2D)	(None, 25, 25, 512)	2359808
block4_pool (MaxPooling2D)	(None, 12, 12, 512)	0
block5_conv1 (Conv2D)	(None, 12, 12, 512)	2359808
block5_conv2 (Conv2D)	(None, 12, 12, 512)	2359808
block5_conv3 (Conv2D)	(None, 12, 12, 512)	2359808
block5_pool (MaxPooling2D)	(None, 6, 6, 512)	0
flatten (Flatten)	(None, 18432)	0
dense (Dense)	(None, 512)	9437696
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 512)	262656
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 29)	14877

Total params: 24429917 (93.19 MB)
Trainable params: 9715229 (37.06 MB)
Non-trainable params: 14714688 (56.13 MB)

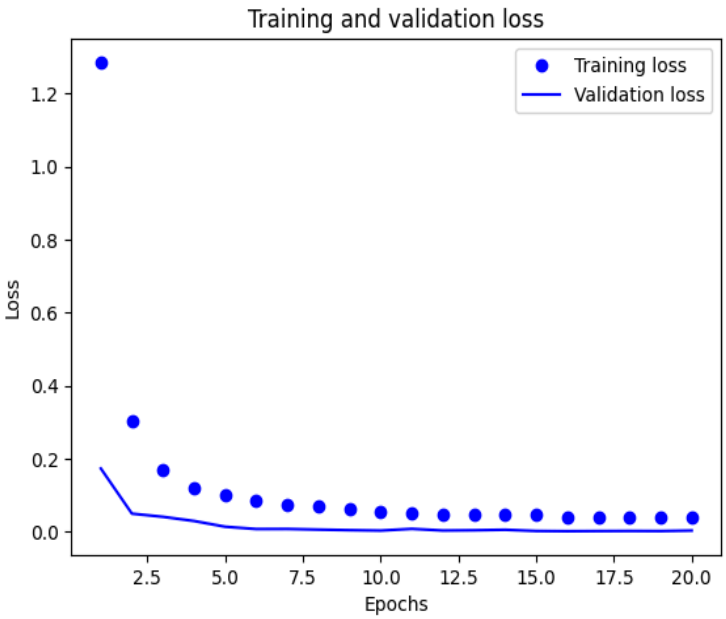
Model Summary

The model was trained with 20 epochs and early stopping set to 4. With these settings we were able to achieve:

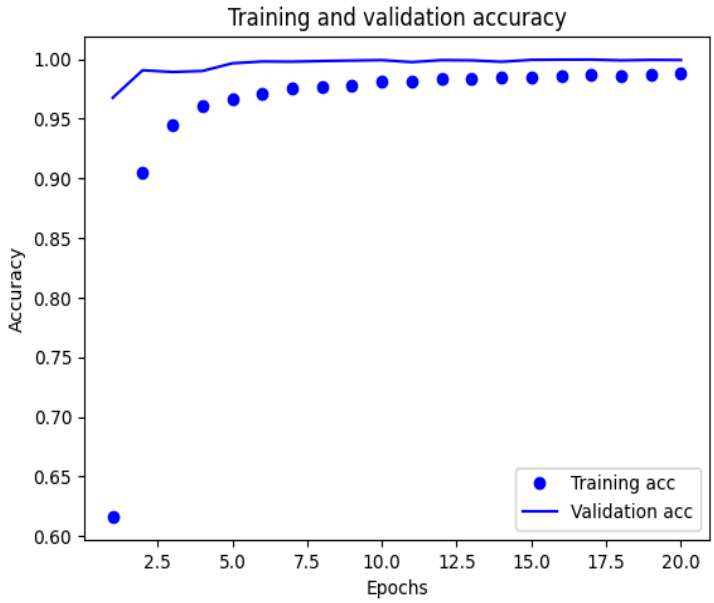
Training Accuracy: 98.81%
Validation Accuracy: 99.92363%
Recall Score: 99.92363%
Precision Score: 99.92337%
F1 Score: 99.92339%

We then utilized plots for further insight on the model and its performance

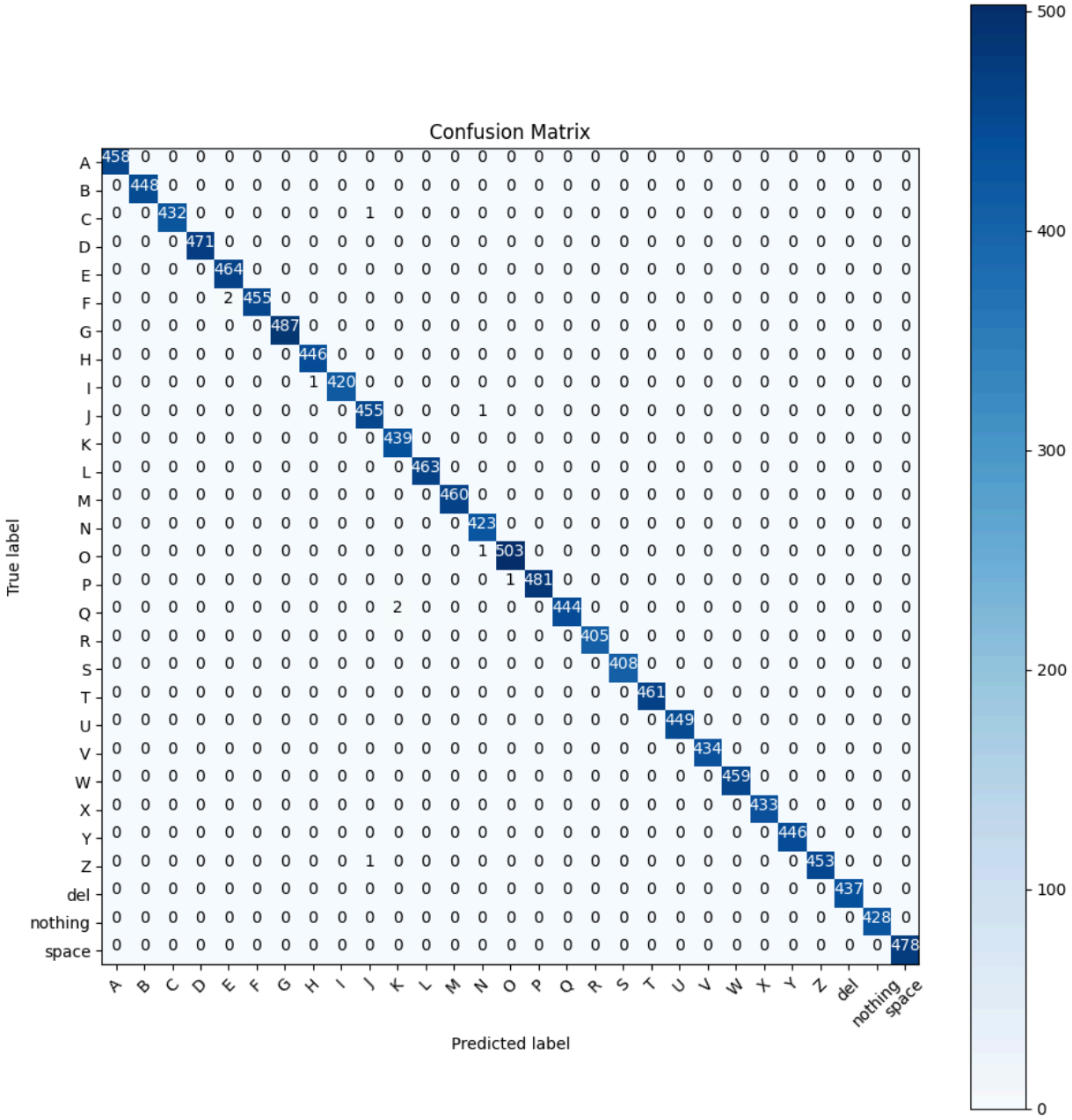
Loss for Training and Validation



Accuracy for Training and Validation



Confusion Matrix



With all these settings we yielded a **99.90% Testing Accuracy**

Conclusion

Overall, throughout the completion of this project we can conclude that CNNs will do the best job when it comes to classifying images, as evidenced by the test accuracy we achieved using the VGG16 model. Our findings command the fact that deep learning architectures are ideal for these kinds of dataset are they are equipped to handle the complexity of details that images tend to have.

References

https://nationaldeafcenter.org/faq/how-many-deaf-people-live-in-the-united-states/

https://www.kaggle.com/code/marwanabudeeb/vgg16-last-edit-x/notebook

https://datascience.stackexchange.com/questions/15135/train-test-validation-set-splitting-in-sklearn

https://stackoverflow.com/questions/63846109/how-to-splitting-training-data-into-smaller-batches-to-solve-memory-error

https://www.tensorflow.org/api_docs/python/tf/keras/Sequential

https://www.tensorflow.org/tutorials/images/classification

https://archive.org/details/handson-machine-learning-with-scikit-2-e-1/page/54/mode/2up?q=test_set&view=theater

https://tanthiamhuat.files.wordpress.com/2018/03/deeplearningwithpython.pdf

https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.interviewquery.com%2Fp%2Fdata-science-memes&psig=AOvVaw36XbwijzO9fS0BX4SqL0pJ&ust=1701822747196000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCNjGyPeF94IDFQAAAAAdAAAAABAI

https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.vectorstock.com%2Froyalty-free-vector%2Fhands-showing-sign-language-alphabet-vector-46696551&psig=AOvVaw39xj-ozJZ8f4W_jjB7wjIS&ust=1701828165066000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCLDsq4ya94IDFQAAAAAdAAAAABAX