Machine Learning a.y. 22-23

Homework 2: Report

Franesco Palandra
1849712

January 9, 2023

1 Introduction

ASL is a complicated and nuanced language used by millions of deaf people in the United States and around the world. It is made up of a sequence of gestures and hand shapes that can be joined in numerous ways to make words and phrases. Recognizing and correctly interpreting these signals is a critical undertaking because it allows persons who are deaf or hard of hearing to communicate with others and fully participate in society. The ASL classification problem involves building a machine learning model that is able to identify and classify individual ASL gestures from a set of images.

Figure 1: ASL Alphabet without J and Z

1.1 Dataset

The American Sign Language letter database of hand gestures represent a multi-class problem with 24 classes of letters (excluding J and Z which require motion). The dataset format is patterned to match closely with the classic MNIST. The training data (27,455 cases) and test data (7172 cases) are approximately half the size of the standard MNIST but otherwise similar with a header row of label, pixel1,pixel2....pixel784 which represent a single 28x28 pixel image with grayscale values between 0-255.

	label	pixel1	pixel2	pixel3	pixel4	pixel5	pixel6	pixel7	pixel8	pixel9	...	pixel775	pixel776	pixel777	pixel778	pixel779	pixel780	pixel781	pixel782	pixel783	pixel784
0	3	107	118	127	134	139	143	146	150	153	...	207	207	207	207	206	206	206	204	203	202
1	6	155	157	156	156	156	157	156	158	158	...	69	149	128	87	94	163	175	103	135	149
2	2	187	188	188	187	187	186	187	188	187	...	202	201	200	199	198	199	198	195	194	195
3	2	211	211	212	212	211	210	211	210	210	...	235	234	233	231	230	226	225	222	229	163
4	13	164	167	170	172	176	179	180	184	185	...	92	105	105	108	133	163	157	163	164	179

Figure 2: First five rows of the dataset

2 Data Preprocessing

By preprocessing the data, you may guarantee that the data is clean, consistent, and in a format that is acceptable for your machine learning assignment. This can boost the model's capacity to learn from data and result in improved performance.

2.1 Class Imbalance Detection

Class imbalance is a problem where the number of examples belonging to one class is significantly larger or smaller than the number of examples belonging to the other classes. This can cause problems when training a model, as the model may be biased towards the majority class and may not be able to accurately classify the minority class. In the case of ASL classification, if the data is imbalanced such that there are significantly more examples of one ASL gesture compared to others, the model may be biased towards classifying all gestures as the majority class. This would lead to poor performance on the minority class.
However, as we can see from the image below3, the data is equally distributed over all the classes so no change is needed.

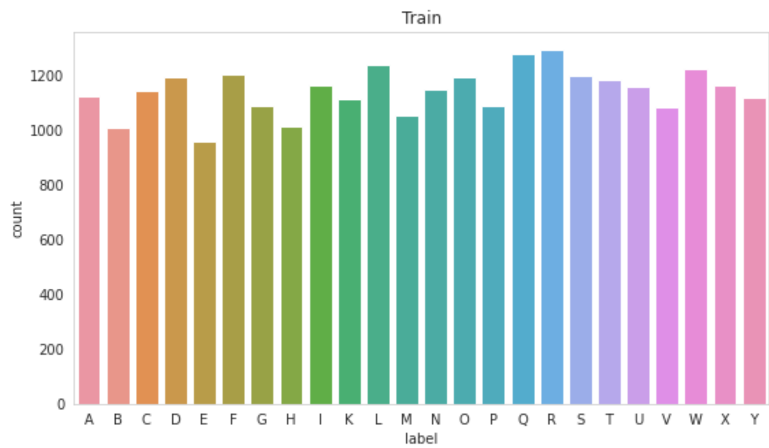


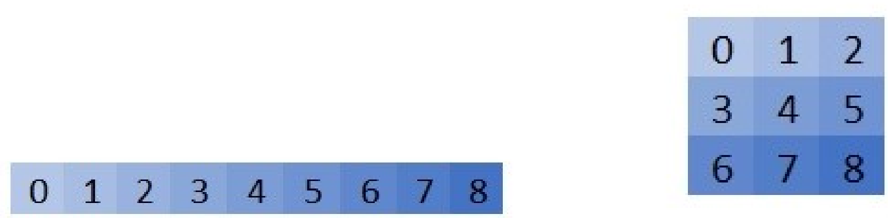
Figure 3: Data distribution over classes

2.2 Normalization of the Data

The data is normalized by dividing each column (each pixel) by 255, resulting in values that range from 0 to 1. Normalizing the input data is very important, especially when using gradient descent which will be more likely to converge faster when the data is normalized.

2.3 Reshapening into Images

The images will need to be reshaped in order to feed into the model. The output images will be 28x28x1 that corrisponds to a gray image



(a) Before (b) After

Figure 4: Reshapening

2.4 Data Augmentation

Data augmentation is a strategy for increasing the size of the dataset by generating altered copies of existing data. This is frequently done to increase the performance of a model. Generating additional data can help to reduce overfitting, in fact the model is exposed to more diverse examples and is less likely to rely on specific patterns in the training data that may not generalize to new examples. It can also help a model to learn more general features that are applicable to a wider range of data.

In particular some parameters have been selected to best represent the augmented data (if we apply a flip transformation on an ASL image it will change the meaning and probably it will create outliers). These are the parameters:

- Rotation
- Zoom
- Width Shift
- Height Shift

2.5 Data Visualization

Here is an example of the resulting input data after the preprocessing for each letter of the ASL.

3 Convectional Neural Network

A convolutional neural network is created by using keras to define each layer sequentially. To complete the model it is compiled using Adam as the optimiser and sparse categorical crossentropy and accuracy as the metrics for evaluation. The model is designed to process a batch of 32 images at a time. The first step in the model's architecture is a series of convolutional layers, which apply a 3x3x1 kernel to the input image and repeat this operation multiple times. After each convolutional layer, a max pooling layer with a window size of 2x2 is applied in order to downsample the image and reduce its dimensions. The resulting feature maps are then flattened and passed through two fully connected (dense) layers before being output as predictions for the 24 classes. By using this architecture, the model is able to learn complex patterns in the input data and make accurate classifications.

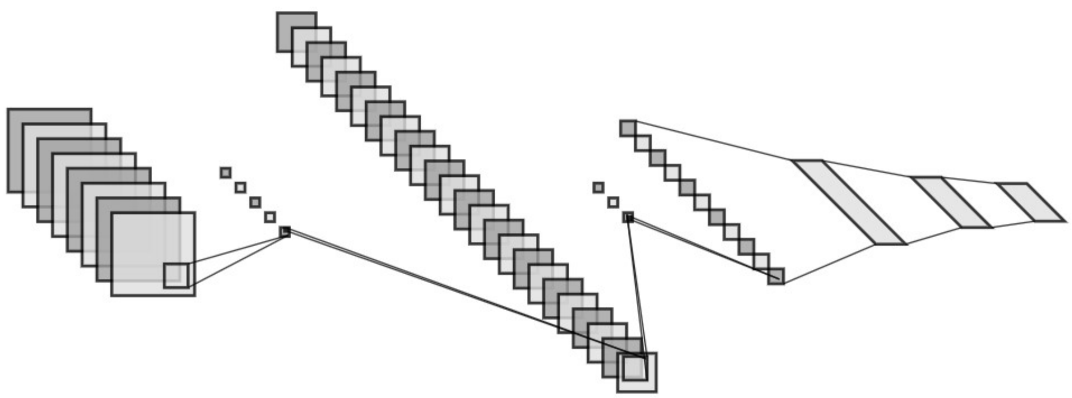


Figure 5: Architecture of Base Model

3.1 Regularizzation

An initial model has been created and have some regularization techniques have been isolated and analyzed.

3.1.1 Dropout

Dropout is a regularization technique that randomly sets a fraction of the input units to zero during training. This helps to prevent overfitting by reducing the complexity of the model and by forcing the model to learn multiple independent representations of the data.

3.1.2 Early Stopping

Early Stopping is a technique that involves stopping the training process before the model has fully converged, based on the performance of the model on a validation set. This can help to prevent overfitting and can lead to better generalization performance.

3.1.3 Learning Rate Reduction

Learning rate reduction is a technique that involves decreasing the learning rate of the optimization algorithm over time. This can help to prevent the optimization algorithm from overshooting the optimal solution and can lead to better generalization performance.

3.1.4 Data Augmentation

Data Augmentation as we have seen already is a technique that involves generating modified versions of the training data in order to artificially increase the size of the dataset. This can help to improve the generalization ability of the model by providing it with a more diverse set of training examples.

3.1.5 Batch Normalization

Batch Normalization is a technique that normalizes the activations of a layer within each mini-batch of data. This helps to improve the stability of the model during training and can accelerate the training process.

3.2 Model Fitting and Evaluation

For each of the models, we fit each model for 15 epochs with their unique characteristics and parameters, and then evaluate the model's performance. The results of the evaluation are shown in terms of accuracy (Figure 11) and loss (Figure 10). As we can see, all of the regularization techniques have improved the performance of the base model to some degree. However, Batch Normalization and Dropout appear to be the most effective techniques, as they have achieved the highest accuracy on the validation set. Additionally, when examining the loss, these two techniques seem to be the most successful at accurately describing the data.

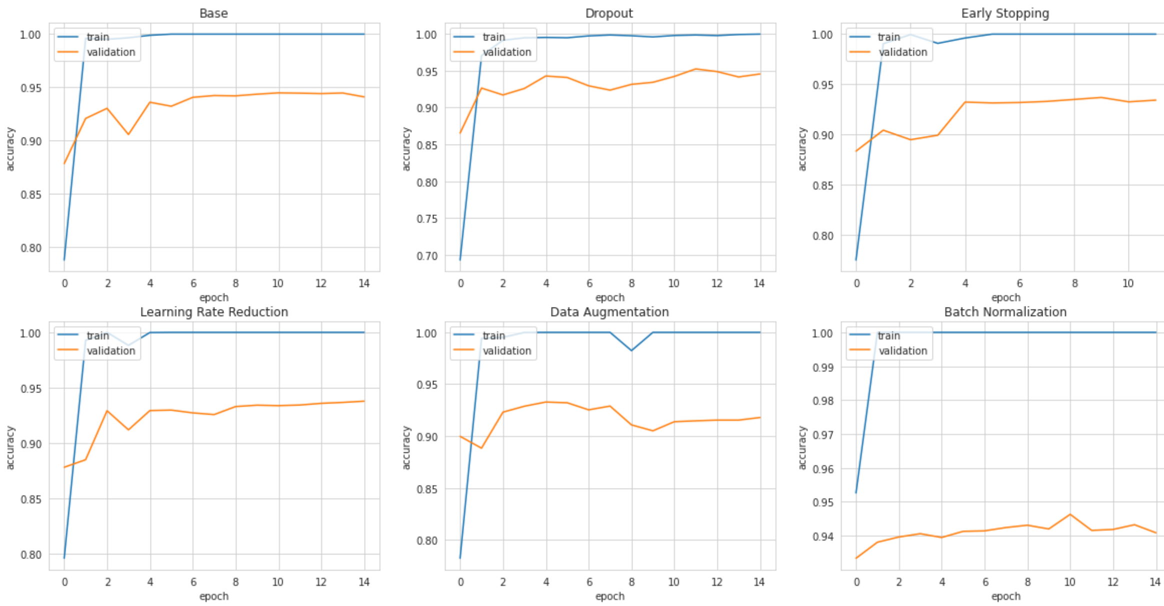
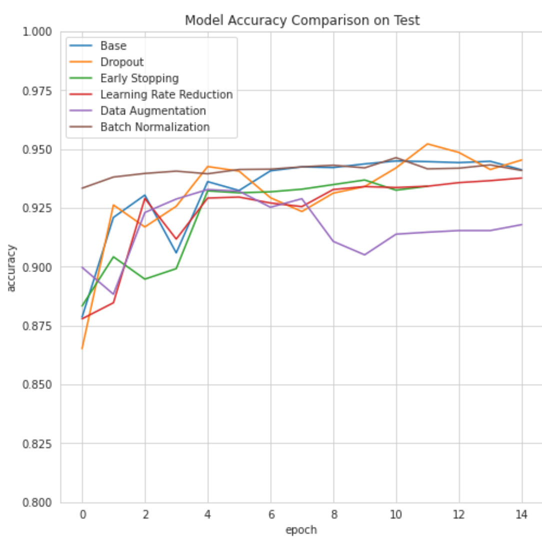


Figure 6: Accuracy of each Model



(a) Accuracy

(b) Loss

Figure 7: Models Comparison

4 Final Model

In constructing the final model, I decided to incorporate all of the regularization techniques that I analyzed, as they all showed some improvement in the model's performance. Specifically, it is included Dropout, Batch Normalization, learning rate reduction, Early Stopping, and Data Augmentation in the final model. In addition to incorporating these techniques, I also decided to increase the depth of the model by adding additional layers. This decision was based on the fact that deeper neural networks are generally able to learn more complex and nuanced features from the data, which can lead to improved performance. In the experiments, it has been found that increasing the depth of the model did indeed lead to a significant increase in accuracy.

4.1 Model Architecture

The model architecture is based on a convolutional neural network (CNN). The CNN processes input images in batches of size 32, and consists of two convolutional layers with a kernel size of 3x3x1. Each convolutional layer is followed by a Dropout layer and a Max Pooling layer. The output of the convolutional layers is flattened and passed through two dense layers, which ultimately lead to a classification output. This architecture is similar to the base model, with the addition of Dropout layers to regularize the model and improve its generalization performance.

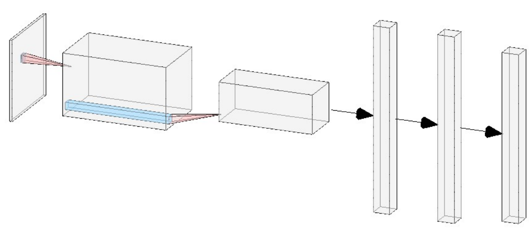


Figure 8: Architecture of the Final Model

4.2 Model Training and Evaluation

The model was trained for a total of 15 epochs using the augmented data and the learning rate reduction and Early Stopping methods. These methods were applied at each epoch in order to continually improve the accuracy of the model. The learning rate reduction method helped to prevent the optimization algorithm from overshooting the optimal solution, while the Early Stopping method stopped the training process if the model was not making substantial progress in the last few epochs, in order to prevent overfitting.

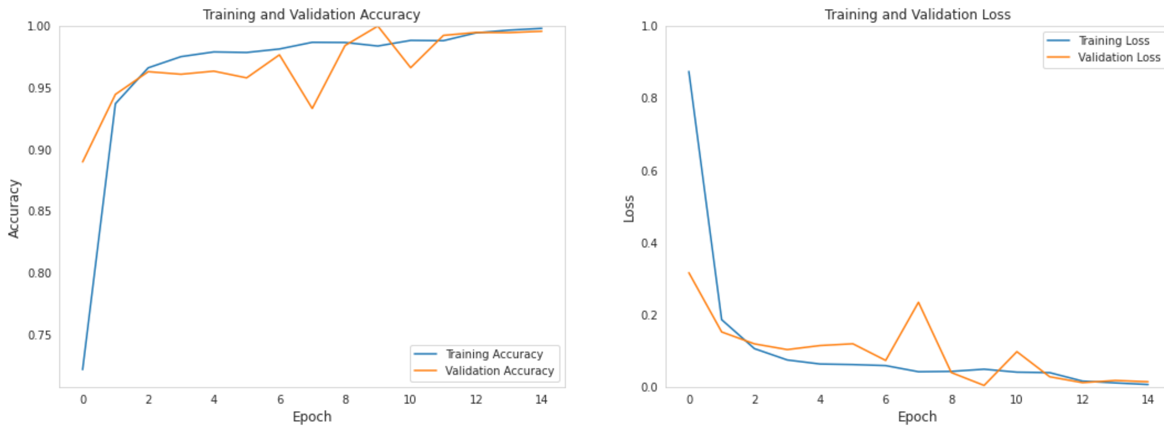
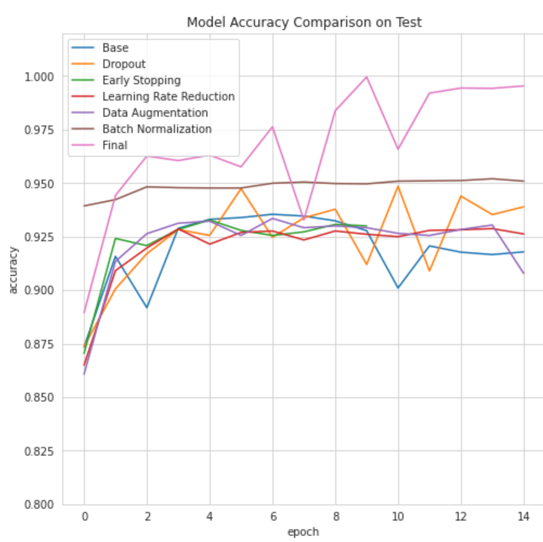
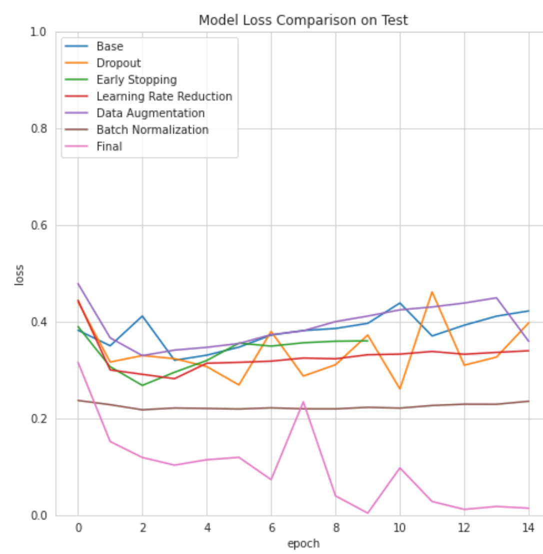


Figure 9: Accuracy and Loss of the Final Model



(a) Accuracy



(b) Loss

Figure 10: Final Model Comparison

The final model achieved significantly better performance than the base model due to the combination of various regularization techniques and the inclusion of an additional layer. The regularization techniques, which included Dropout, Batch Normalization, learning rate reduction, Early Stopping, and Data Augmentation, helped to improve the stability and generalization ability of the model. The additional layer also contributed to the improved performance of the final model, as it allowed the model to learn more complex and higher-level features from the data. Overall, the combination of these techniques and the additional layer led to significant performance improvements over the base model.

4.2.1 Evaluation Metric

We are going to use accuracy as the evaluation metric. Accuracy is the ratio of correctly classified samples to the total number of samples.

The overall accuracy of the model is 0.99, with a high f1-score and a precision of 1. This indicates that the model is performing exceptionally well on the classification task. Furthermore, the confusion matrix shows that the model was able to correctly classify all classes, indicating that there were no misclassification errors made by the model.

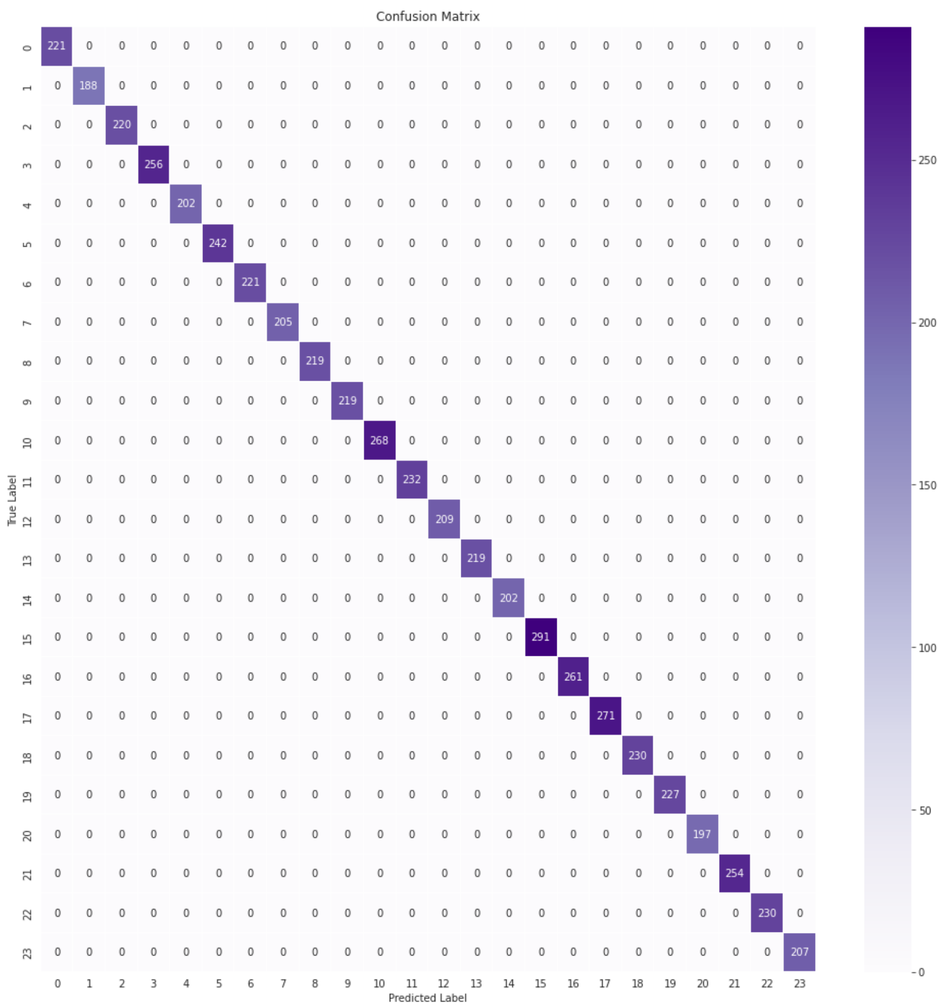


Figure 11: Confusion Matrix of the Final model

5 Conclusions

In conclusion, the results of this project show that the base model was already performing well on the classification task. However, by applying various regularization techniques we were able to further improve the performance of the model. In particular, the combination of these techniques led to a significant boost in accuracy, resulting in an overall accuracy of 0.99.

These results suggest that the problem of ASL is well-described by this model, and that the regularization techniques we applied were effective at improving the model's performance. It is worth noting that adding an additional layer to the model was particularly effective at boosting the model's accuracy.

Overall, this project demonstrates the effectiveness of CNNs and highlights the importance of carefully considering and applying regularization techniques in order to achieve optimal performance.