Faculty of Engineering & Technology

Electrical & Computer Engineering Department

COMPUTER VISION

ENCS5343

Course Project

Students' Name:

Lama Naser	1200190
Basheer Arouri	1201141

Instructor: Dr. Aziz Qaroush

Section: 1

Date: 30-1-2024

Abstract

This assignment aims to know how to recognize Arabic Handwritten Characters using Convolutional Neural Networks (CNNs). The main goals are to create a basic CNN for this task, covering the network's structure, how it learns (loss functions and optimizers), and how to train and assess its performance. It also aims to visualize how well the CNN is doing. The assignment explores advanced techniques like making the model more versatile using methods like random cropping and rotation. It introduces the idea of using pre-trained models (like VGG16) and adapting them for this task. Additionally, it briefly mentions more complex CNN types (ResNet or DenseNet) and their potential in recognizing Arabic Handwritten Characters.

Table of Contents

Abstract I

List of Figures 1

List of Tables 1

1- Introduction 2

3- Experimental Setup and results 3

3.1- Referenced research 3

3.2- Task one: Building CNN Model 4

3.2.1- Fine tuning the parameters 6

3.2.2- Final results 7

3.3- Task Two: Data Augmentation 8

3.3.1- Fine Tuning data Augmentation Techniques 8

3.3.2- Final Results 10

3.4- Task Three: Using Well-known CNN Architecture 11

3.5- Task Four: Using Pre-trained CNN with Transfer Learning 12

4- Conclusion 13

5- References 14

List of Figures

Figure 1: Task one, first trial 5
Figure 2: learning rate VS test error plot 6
Figure 3: Batch size VS test error plot 6
Figure 4: Task one final results 7
Figure 5: Effect of zoom ranges on both accuracy and loss for testing 8
Figure 6: Effect of width shift ranges on both accuracy and loss for testing 8
Figure 7: Effect of height shift ranges on both accuracy and loss for testing 9
Figure 8: Effect of rotation ranges on both accuracy and loss for testing 9
Figure 9: Data augmentation results 10
Figure 10: Task three results 11
Figure 11: pre-trained model with transfer learning results 12

List of Tables

Table 1: initial parameters 4

1- Introduction

Automatic handwriting recognition can be defined as the ability of a system to identify human handwritten input. The handwriting can be from many sources, such as paper documents, images, touch-screens, or other devices. Handwriting that is input through scanning is considered offline, while input through a pen tip is considered online. Handwriting recognition is considered a challenging problem in computer vision. Normally, there are variations in the handwriting of different individuals. In addition, the handwriting of a single writer may be slightly different.

Handwriting recognition problem has been studied using many methods, such as: support vector machines (SVMs), K-nearest neighbors (KNNs), neural networks (NNs), and, recently, convolutional neural networks (CNNs). However, previous research in this field has focused on the recognition of handwriting in Latin languages. Arabic language is the most widely spoken Semitic language, with an estimated 300 million speakers. It is ranked as the fifth most prevalent spoken language in the world. Handwriting recognition for Arabic is still an open problem for many reasons. Arabic is written with the cursive Arabic alphabet from right to left. The Arabic alphabet contains 28 letters.

In this report project, we present a dataset contains 16,800 images of Arabic character in one shape (different cases for each letter are not included). This dataset is divided into 80% as training set (13,440 images) and 20% as testing set (3,360 images). Then, different CNN’s were used on this dataset and compared with each other.

3- Experimental Setup and results

3.1- Referenced research

Arabic handwriting recognition system using convolutional neural network research by Najwa Altwaijry and Isra Al-Turaiki was selected to be as the reference of desgining our CNN model. This research was chosen since it is very close for our project objectives and specifications. Also the accuracy results for this research were very high for two different datasets. The first one is the Hija'a dataset that was collected from children handwritten characters. The second one is Arabic Handwritten Character Dataset (AHCD) dataset. The results of its model's performance is promising, achieving accuracies of 97% and 88% on the AHCD dataset and the Hijja dataset, respectively.

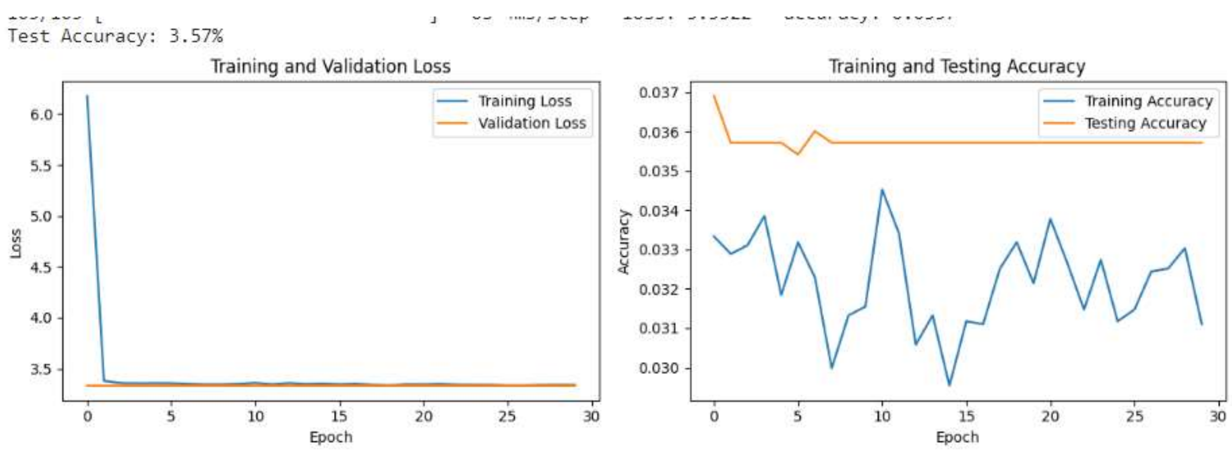
3.2- Task one: Building CNN Model

At first, the same parameters that were used in the research were used in our model to test its results on our dataset. These parameters were collected and recorded in the following table.

Table 1: initial parameters

Category	Parameter	Value/Type
Architecture	Number of layers	Seven layers
	Types of layers	Convolutional, Pooling, Batch Normalization, Flatten, Fully Connected and Dropout
	Activation functions	ReLU
Convolutional Layer Parameters	Number of filters	64, 128, 256
	Filter size	(3, 3)
	Stride	1
	Padding	Zero padding of 1.
Pooling Layer Parameters	Pool size	(2, 2)
	Pool type	Max Pooling
Fully Connected Layer Parameters	Number of neurons	4096, 1024, 512
Training Hyperparameters	Learning rate	0.001
	Batch size	32
	Epochs	30
	Optimizer	Adam (Loss function: categorical cross-entropy loss)
	Overfitting Handling	Dropout (0.8)

The results got from using the above parameters in the CNN model gave the following plots with final test accuracy = 3.57%.



```
Test Accuracy: 3.57%
```

Figure 1: Task one, first trial

IT was noticed that both training and validation losses remain constant, indicating difficulty in learning meaningful patterns. The consistently low training accuracy suggests the model struggles to capture underlying patterns. To resolve this proble, some parameters fine tuning were done as shown in the section below.

3.2.1- Fine tuning the parameters

1) Learning rate: as shown in the plot below, the best learning rate that can be used in this model is 0.0001.

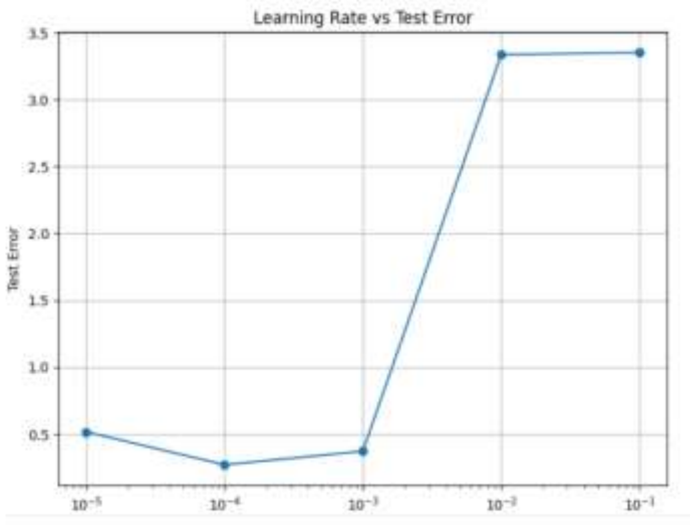


Figure 2: learning rate VS test error plot

2) Batch size: as shown in the plot below, the best batch size that can be used for this model is 16.

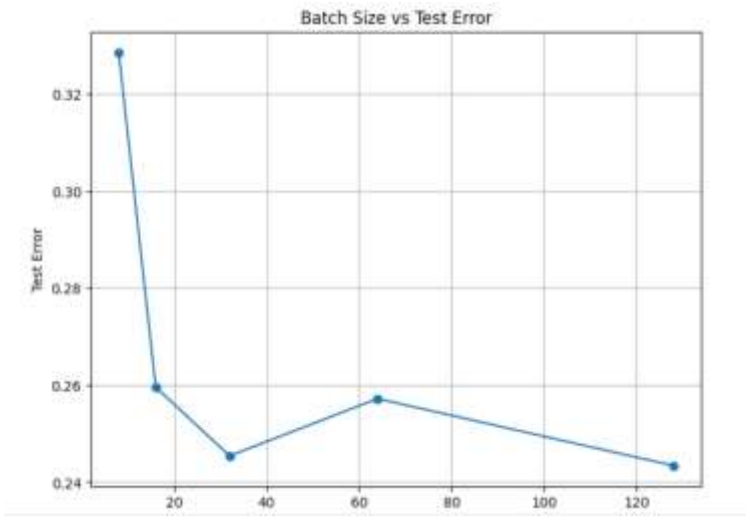


Figure 3: Batch size VS test error plot

3.2.2- Final results

After the fine tuning done above, it was determined to use all the fully connected layers with 60% dropout rate (experimentally) instead of 80% dropout rate. Also, it was determined to use learning rate of 0.001 instead of 0.001, batch size of 16 instead of 32 and number of epochs 20 (experimentally) instead of 30. By using all these updated parameters, the following results were got for (training and validation loss VS epochs) and (training and testing accuracy VS epochs) with final testing accuracy equals to 92.98%..

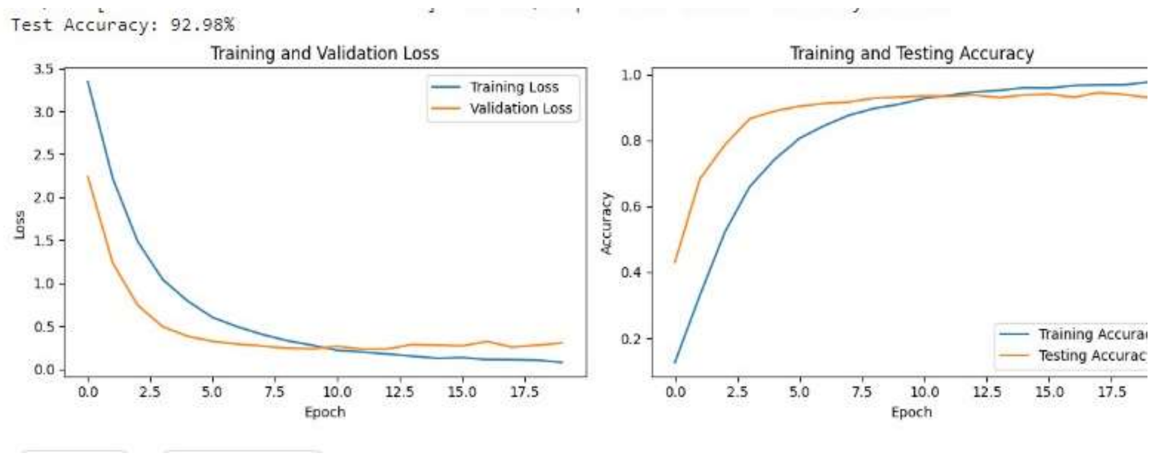

```
Test Accuracy: 92.98%
```

Figure 4: Task one final results

As shown in the plots above, the training loss VS epoch is decreasing which means that the model is effectively learning from the training data, reducing the error during training. The validation loss VS epoch also is decreasing which means that model is improving in generalization performance on unseen data. Moreover, the training accuracy increases over epochs, indicating that the model is becoming more accurate on the training data. The validation accuracy also increases over the epochs which means that the model's improvement in correctly classifying examples from the validation set.

3.3- Task Two: Data Augmentation

Three data augmentation methods were selected to increase the size of the dataset set as a try to improve our model performance. The methods are the zooming technique, rotation technique and transition technique includes horizontal and vertical shifting. These methods were suitable for the task of AHCR on the dataset provided for training and testing. In order to specify the parameters values of the selected methods, the following section explains the methodology used.

3.3.1- Fine Tuning data Augmentation Techniques

1) First, the following plots were obtained to determine which zoom range will result in highest testing accuracy and lowest testing loss. As shown below, the best value for the zoom range was 0.5 that gave 94.4% accuracy and 25.1% loss.

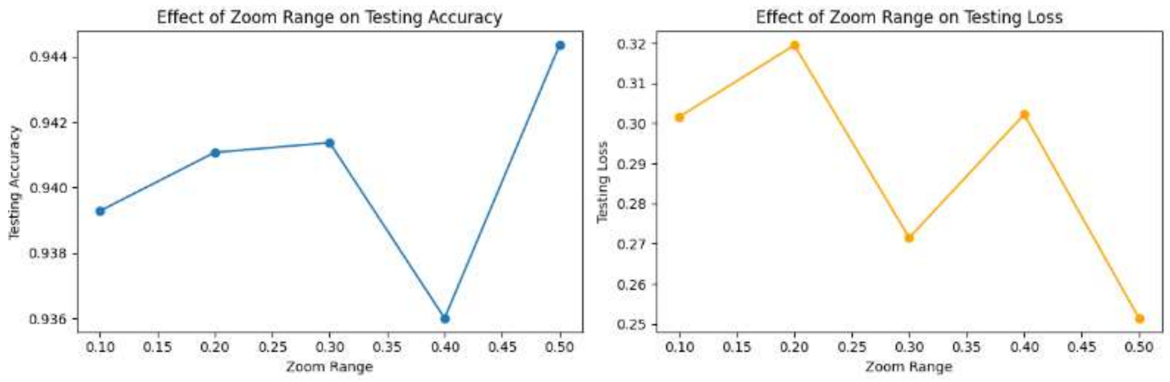


Figure 5: Effect of zoom ranges on both accuracy and loss for testing

2) Second, the following plots were obtained to determine which width shift range is better to use with the determined zoom range above (0.5). As shown below, the best value for the width shift range was 0.1 that gave 94.2% accuracy and 26.7% loss.

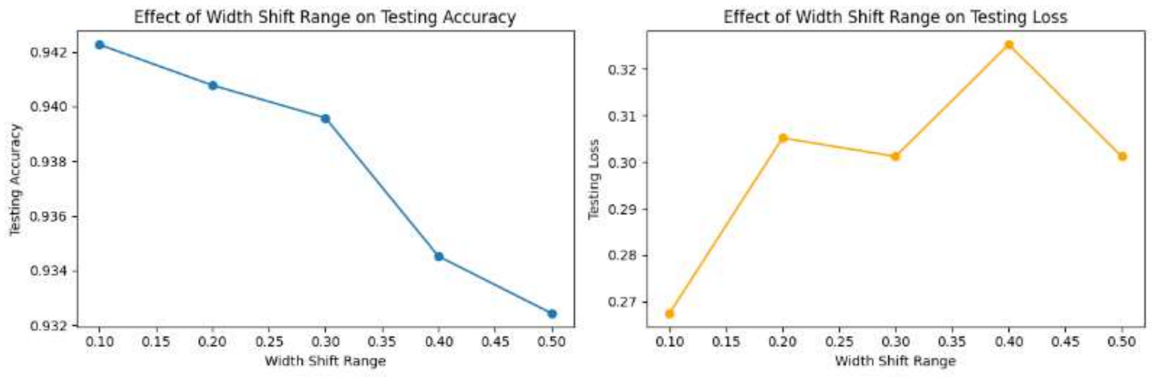


Figure 6: Effect of width shift ranges on both accuracy and loss for testing

3) Third, the following plots were obtained to determine which height shift range is better to use with the determined zoom range (0.5) and width shift range (0.1). As shown below, the best value for the height shift range was 0.2 that gave 94.4% accuracy and 25.4% loss.

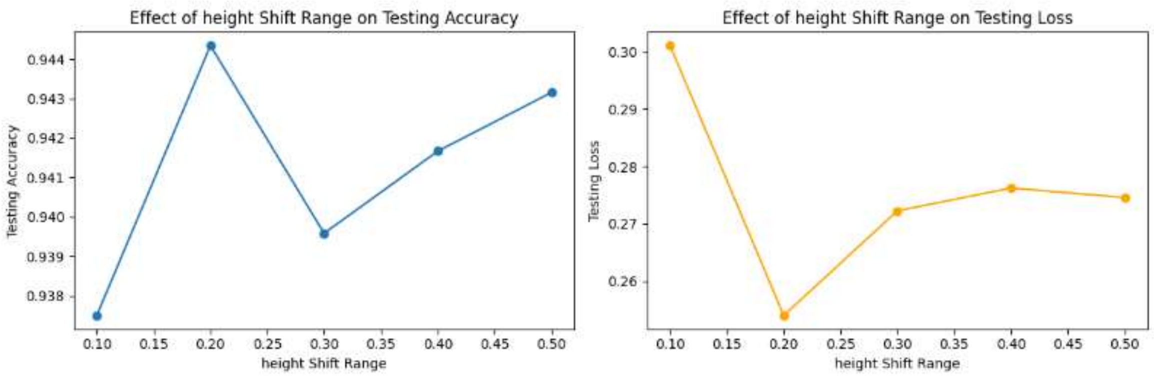


Figure 7: Effect of height shift ranges on both accuracy and loss for testing

4) Finally, the following plots were obtained to determine which rotation range is better to use with the determined zoom range (0.5), width shift range (0.1) and height shift range (0.2). As shown below, the best value for the rotation range was 20 that gave 94.5% accuracy and 26.2% loss.

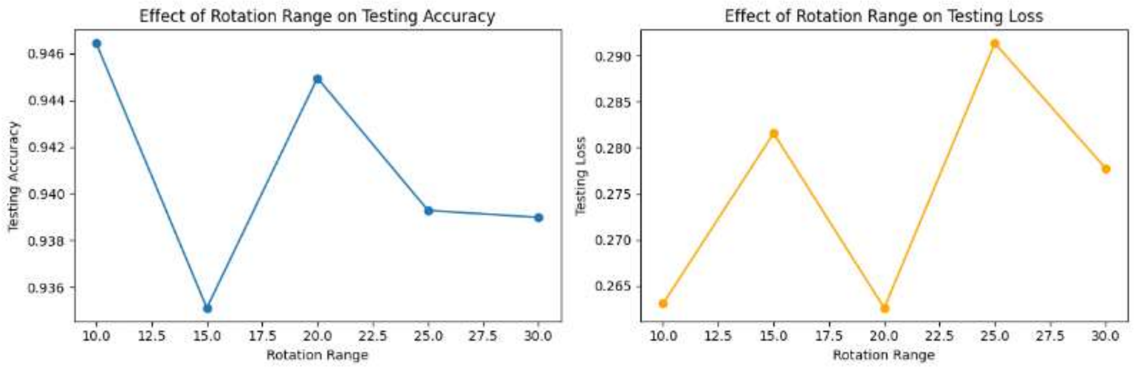


Figure 8: Effect of rotation ranges on both accuracy and loss for testing

3.3.2- Final Results

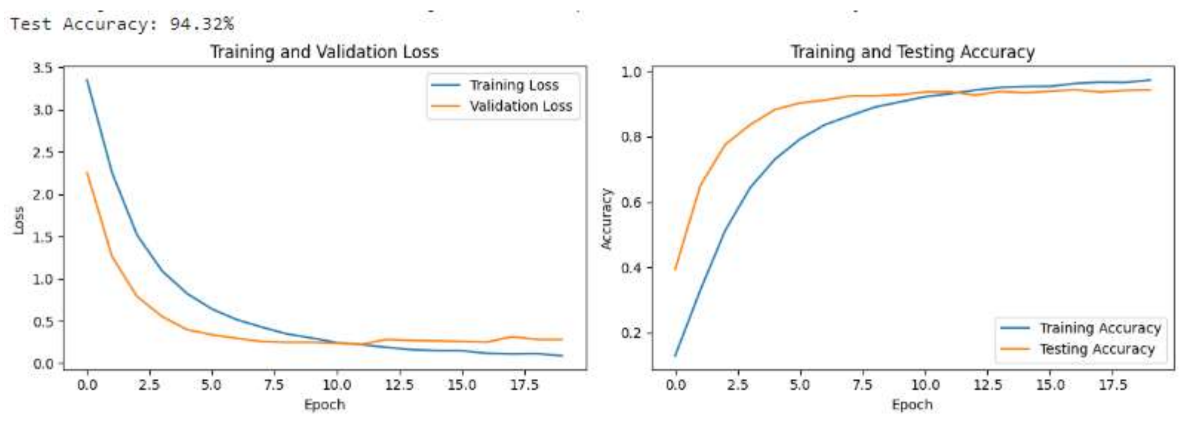

Test Accuracy: 94.32%

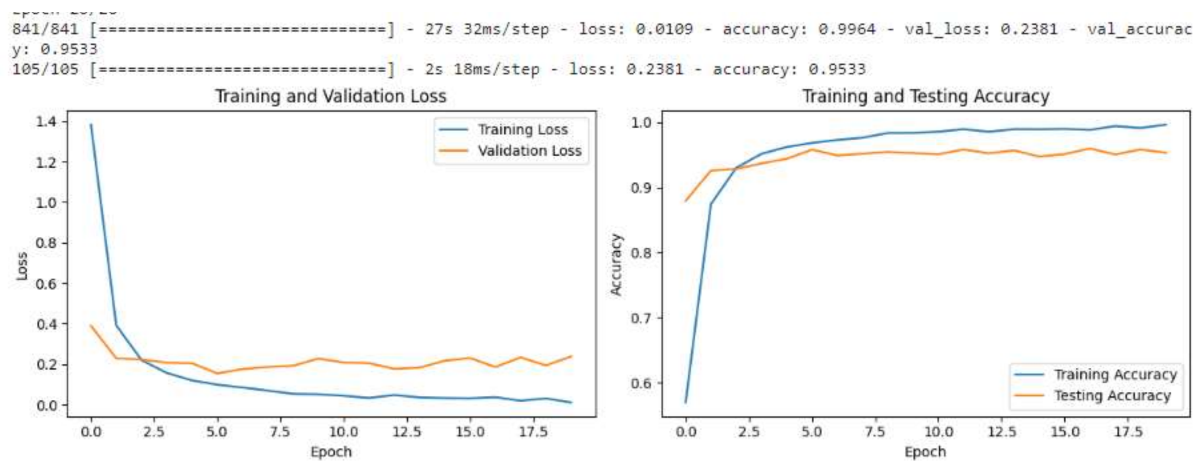
Figure 9: Data augmentation results

As shown in the above plots, both validation and training losses are decreasing over epochs which means that the model is effectively learning from the training data, reducing the error during training. Also it means that the model is improving in generalization performance on unseen data. Moreover, both of the training and testing accuracies are increasing over epochs, which means that the model is becoming more accurate on the training data and the model's improvement in correctly classifying examples from the validation set.

It was noticed that using data augmentation with the CNN model has improved the model performance by increasing the testing accuracy and decreasing the testing loss. Also by using the data augmentation, the fluctuations in the validation accuracy and loss plots were decreased and became smoother than the plots of the model that didn’t use data augmentation. This happened because data augmentation generates new images with slight variations of the original ones. This helps the model learn invariant features, making it more robust to changes leading to reducing the over fitting problem.

3.4- Task Three: Using Well-known CNN Architecture

In this section, the CNN selected was AlexNet architecture. This model is not provided by Kerase.application library so we built its architecture using Kerase functions. AlexNet consists of eight layers in total, including five convolutional layers and three fully connected layers. The selection of AlexNet was based on making balance between the complexity and the computational time. For example, the LeNet and VGG-16 architectures are simpler thanAlexNet and they don't need too much computational time. However, ResNet and Inception architectures are more complex and they need much more computational time. So that choosing AlexNet architecture was more suitable since it is in between those other architictures.



```
841/841 [==============================] - 27s 32ms/step - loss: 0.0109 - accuracy: 0.9964 - val_loss: 0.2381 - val_accurac
y: 0.9533
105/105 [==============================] - 2s 18ms/step - loss: 0.2381 - accuracy: 0.9533
```

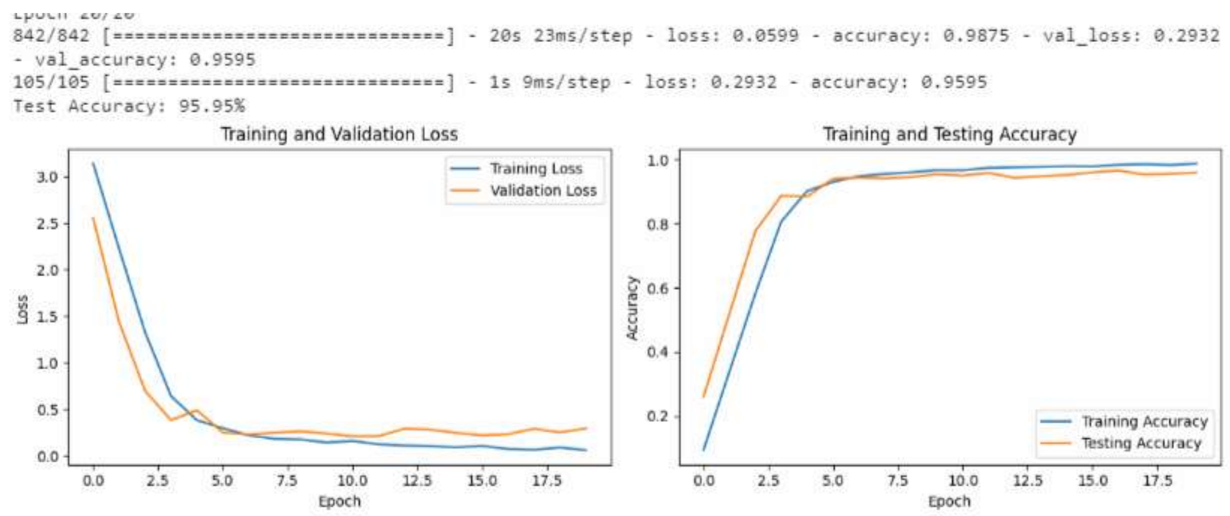
Figure 10: Task three results

As shown above, the results have improved comparing with results in Task 1 and Task 2. This can be explained and justified according to the strong of the Alexnet architecture that we chose. The idea of the Alexnet that we used is 8 layers (5 for training and 3 for output) to train and test our dataset. Test accuracy in the First Task is about 92.98%, Second Task is about 94.32% and when we applied Alexnet, Test accuracy is about 95.33%, which is very good. We can notice that there is a difference between the third task and the previous tasks in the accuracy.

3.5- Task Four: Using Pre-trained CNN with Transfer Learning

In this section, The VGG-16 weights were pre-trained in ImageNet. In our case, we used the VGG-16 model by freezing the weights of the first four groups and using the pre-trained convolutional base. In addition, we added our layers, namely, flattening layers and a fully connected layer with a ReLU activation function, a dropout layer with a ratio of 60%, and at least one Softmax layer as an output layer to distinguish between classes (28 classes). To train the proposed VGG-16, Adam was used as the optimizer. Finally, we trained the model using our dataset.

The transfer learning approach used in our code is “Feature extraction” which is a pre-trained VGG-16 model is used as a fixed feature extractor, and only a new classifier is trained on top of the pre-trained model.



```
842/842 [==============================] - 20s 23ms/step - loss: 0.0599 - accuracy: 0.9875 - val_loss: 0.2932
- val_accuracy: 0.9595
105/105 [==============================] - 1s 9ms/step - loss: 0.2932 - accuracy: 0.9595
Test Accuracy: 95.95%
```

Figure 11: pre-trained model with transfer learning results

As shown in the above plots, using a pre-trained CNN model with our task has achieved better results than the results got from the first, second and third tasks before. For example, this task has achieved a higher training and testing accuracy. Also it reduced the overfitting problem that both validation and training accuracies have the same or parallel increasing behavior as well as the training and validation losses have the same or parallel decreasing behavior.

4- Conclusion

In this project, we proposed deep learning models to recognize handwritten characters in Arabic. The first CNN model was implemented starting from a reference research then some improvements were added and achieved around 92.95% testing accuracy. The second models was implemented using the same model built in the first one but this time by using data augmentation as an enhancement technique to the original model and it achieved around 94.32% testing accuracy. The third CNN model was built with a well-known CNN model which is AlexNet with the use of the data augmentation, the selection of this model was based on make a balance between the network complexity and the computational time. This model achieved around testing accuracy 95.33%. The last model used is a pre-trained model VGG-16 with "Feature Extraction" transfer learning and it achieved around 95.95% testing accuracy.

5- References

[1] https://link.springer.com/article/10.1007/s00521-020-05070-8

[2] https://www.mdpi.com/2076-3417/13/3/1692