Deeper Networks for Image Classifications

Izabella Kamila Ożóg
220997300
ECS795P QMUL (2023)
CW3 Mini-Project

In this study, the performance of two deep convolutional neural networks (DCNNs), ResNet and VGGNet, is compared on image classification tasks using the MNIST and Caltech101 datasets. The study evaluates the effectiveness of these networks with and without the use of transfer learning. Results show that both models perform very well on the MNIST dataset while the fine-tuned ResNet is superior on the Caltech101 dataset.

I. Introduction

Image classification is an important problem that has many real-world applications in fields such as medicine, manufacturing, and security, among many others. For many years it remained a huge challenge. Only recently, the advancements in deep learning led to a significant improvement in this area.

This report dvelves into the use of deep learning for image classification. It is focused on two popular networks: ResNet and VGGNet. The main objective is to evaluate the effectiveness of DCNNs on image classification task through the comparison of the two selected models using two different benchmarks, specifically, MNIST and Caltech101. These datasets differ in the complexity and provide insights into computational and predictive performance of the models. Additionally, this report explores the effects of transfer learning to the models training.

The "Related Work" section of this report provides a detailed introduction to ResNet, VGG, transfer learning, MNIST, and Caltech101. The next section, "Methodology", describes the experimental setup used for the training. Then finally, "Results and Analysis" section provides a discussion on the quantitative and qualitative performance of the evaluated models. The last section, "Conclusions", summarises the key findings of the report.

II. Related Work

This section provides an overview of the image classification task and the historical methods that have been used to address it. It explains the role of deep learning and focuses on two prominent architectures – VGG and ResNet. The section compares these models and highlights their strengths and weaknesses. Additionally, the section introduces the concept of transfer learning. It is a powerful technique used in deep learning to enhance model performance by using pre-trained models. Finally, the section provides an introduction to the datasets used in the experiments, specifically the MNIST dataset, a widely-used dataset for handwritten digit recognition, and the Caltech101 dataset, a more challenging dataset containing 101 different objects.

A. Image Classification and DCNNs

Image classification is a fundamental task in computer vision. It involves identification of objects, scenes or patterns in the images. Over the years, researchers developed a range of different methods for image classification, including rule-based systems, statistical models, and machine learning algorithms.

Historically, image classification relied on handcrafted features and shallow learning models. The early approaches to image classification used simple feature extraction techniques. Some of the popular ones included histogram of oriented gradients (HOG) [1], scale-invariant feature transform (SIFT) [2], and local binary patterns (LBP) [3]. These features were then fed into machine learning classifiers, such as support vector machines (SVMs) [4], decision trees [5], or random forests [6].

However, the performance of these approaches was limited by their reliance on domain expertise and the difficulty of feature engineering. The lack of scalability of these methods to large datasets and the need for manual feature engineering was soon to be addressed by DCNNs.

CNNs have been around since the late 1980s, but they were not widely used until the mid-2000s when advancements in computing power and the availability of large labeled datasets contributed to their development. In 2012 the AlexNet DCNN model have revolutionized image classification, achieving remarkable results on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [7]. DCNN is a more complex and powerful version of CNN, having many more layers and parameters. AlexNet used several new techniques, including the use of ReLU activation functions, data augmentation, and dropout regularization, which helped to improve its performance. The architecture of AlexNet consisted of five convolutional layers, followed by three fully connected layers.

DCNNs automatically learn the feature representations from raw image pixels, without the need for handcrafted features. This end-to-end learning approach has enabled DCNNs to achieve state-of-the-art results on various image classification tasks, such as

object recognition, scene understanding, and semantic segmentation.

DCNNs are composed of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers perform the feature extraction by convolving a set of learnable filters with the input image. The pooling layers downsample the feature maps to reduce the spatial dimensionality, while preserving the important features. The fully connected layers perform the classification by computing the softmax probability distribution over the output classes.

B. VGGNet and ResNet: A Comparative Analysis

VGGNet [8] and ResNet [9] are two popular architectures that have further improved the performance of DCNNs and achieved remarkable results on image classification benchmarks. [10]

VGGNet, introduced in 2014, is characterized by its use of very small convolutional filters (3x3) and its deep architecture (up to 19 layers). VGGNet improved DCNNs by increasing the depth of the network, which allowed for more abstract features to be learned. The strengths of VGGNet include its simplicity and ability to generalize well to new datasets. However, VGGNet also has weaknesses such as its high computational cost due to its deep architecture and the potential for overfitting when training on small datasets.

ResNet, introduced a year after VGGNet, addressed the problem of vanishing gradients by introducing skip connections that allowed the gradients to flow directly from one layer to another. This helped to prevent the gradients from becoming too small and allowed for the network to be trained even deeper. ResNet also improved the performance of CNNs and achieved even better results than VGGNet on image classification tasks. On the ImageNet dataset, an ensemble of ResNets achieved 3.57% error on the ImageNet test set, winning the 1st place on the ILSVRC 2015 classification task. ResNet's strengths include its ability to train very deep networks (up to 152 layers) and its superior performance on image classification tasks. In both VGGNet and ResNet the increased complexity of the architectures with more layers can lead to overfitting and a high computational cost.

C. Transfer Learning

Transfer learning is a machine learning technique that allows us to use knowledge gained from one domain to improve learning in another domain. [13] In some domains, it is difficult to construct a large-scale well-annotated dataset due to the expense of data acquisition and costly annotation, which limits its development. Transfer learning allows us to use knowledge gained from a related domain to improve learning in the target domain.

Transfer learning has been successfully applied to many domains, including computer vision, natural language processing, and speech recognition. It has the potential to improve learning performance, reduce the amount of labeled data required, and accelerate the training process. However, there are still many challenges in transfer learning, such as domain adaptation, negative transfer, and selecting appropriate pre-trained models.

D. MNIST and Caltech101 Datasets: Image Classification Benchmarks

MNIST [11] and Caltech101 [12] are two popular image classification benchmarks used to evaluate the performance of DCNNs [10].

MNIST is a dataset of handwritten digits (Fig. 1), consisting of 60,000 training images and 10,000 test images. The images are grayscale and have a resolution of 28x28 pixels. The goal of the dataset is to correctly classify the digits from 0 to 9. MNIST is widely used as a benchmark for evaluating the performance of DCNNs due to its simplicity and ease of use.

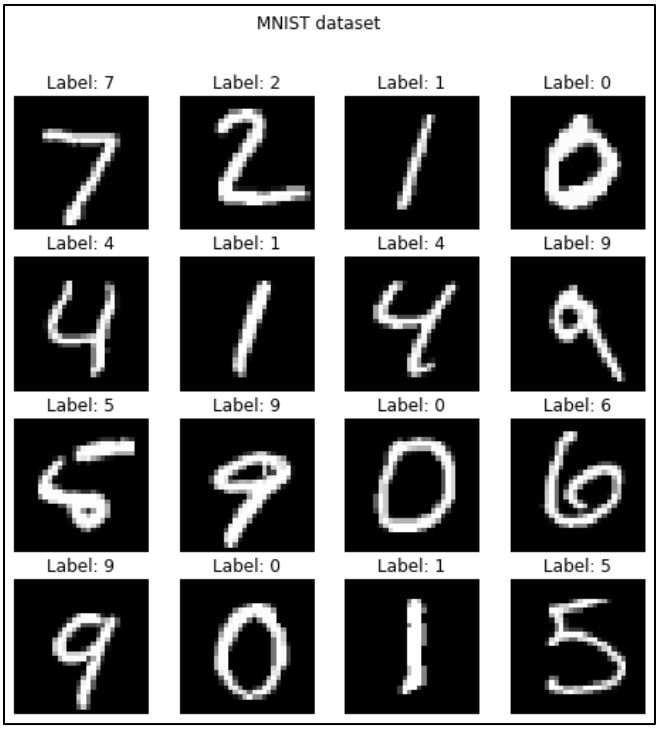

Fig. 1. Example images and labels inside the MNIST dataset

Caltech101, on the other hand, is a more complex dataset consisting of 101 object categories, such as airplanes, faces, and motorbikes (Fig. 2). The dataset contains 9,144 images, with an average of 50 images per category. The images are of varying sizes and resolutions, and the dataset is more challenging than MNIST due to the high variability in object appearance and the large number of categories. Caltech101 is commonly used as a benchmark for evaluating the performance of DCNNs on more complex image classification tasks.

Fig. 2. Example images and labels inside the Caltech101 dataset

In the following sections, a comparative analysis of VGGNet and ResNet models on the MNIST and Caltech101 datasets will be provided, accompanied by investigation of the benefits of transfer learning. The experiments will highlight the strengths and weaknesses of each model and demonstrate the effectiveness of deep learning in image classification.

III. Methodology

This section explains the implementation of model training and testing settings. Specifically, it is divided into three parts: "Datasets", "Model Architecture", and "Training". In the first part, the preprocessing steps for both datasets are presented. Then, in the "Model Architecture" section, a detailed description of VGGNet and ResNet implementation is provided. Finally, in the last section, additional information on the hyperparameters used is provided.

A. *Datasets*

a) MNIST

The MNIST dataset was downloaded from the Torchvision built-in dataset module torchvision.datasets.MNIST, with a standard split into training and testing datasets, as described in the "Related Work" section. The dataset was kept at a standard size of 28x28. The only preprocessing step performed for this dataset, was converting it to the Tensor. The dataset was set to be reshuffled during training after each epoch and not reshuffled during testing.

a) Caltech101

The Caltech101 dataset was also downloaded from the built-in dataset module (torchvision.datasets. Caltech101). This dataset was randomly split into 80% training and 20% testing sets. This resulted in 7315 training images and 1829 testing images. To make training with my model architecture possible, all images were transformed to a uniform size of 224x224 pixels since originally they had varying sizes. Additionally, to ensure the same number of channels, the images were converted to RGB (some images were originally in grayscale). All images were also converted to the Tensor and the same reshuffling system as with MNIST was set.

B. *Model Architecture*

a) ResNet-18

The ResNet-18 model was used as the base model, which consists of 18 layers and is the smallest version of the ResNet model available in the Torchvision module. A custom output layer was added to make it flexible to different numbers of outputs (10 for MNIST and 101 for Caltech101). Additionally, the first layer was modified to be flexible for the number of channels in the input image (1 for MNIST and 3 for Caltech101). The model was set up to be initialised in two ways: with pre-trained weights from the IMAGENET1K_V1 dataset or with untrained weights. This allows comparison of model training with and without transfer learning.

a) VGG-11

The VGG-11 model was used as a base model, which is the smallest version of the VGG model available in the Torchvision module, consisting of 11 layers. Similar to the ResNet-18 model, a custom output layer was added to it at the end to make it flexible to different numbers of outputs. The first layer was also modified to be flexible to the number of channels in the input image. The model can be initialized in two ways, the same as ResNet-18, to allow for transfer learning comparison. Additionally, in the forward function, the input image is resized to 224x224 using bilinear interpolation before being passed through the model.

C. *Training*

For all the trainings in this study, the learning rate was set to 0.001, the cross entropy loss function was used as the objective function, and the Adam optimizer was utilized for the training optimisation. Additionally, a random seed was set to 7 for reproducibility of each training run.

On the other hand, the batch size and number of epochs were varying across different trainings, as visible on the Table 1. They were selected based on the complexity of the datasets, the computational

performance of the models and the constraints of the computing resources.

TABLE I. TRAINING HYPERPARAMETERS

Dataset Name	Model Name	*Values Used*	
		Number of Epochs	*Batch Size*
MNIST	ResNet	5	32
	VGGNet	5	8
Caltech101	ResNet	10	32
	VGGNet	10	6

Originally, it was attempted to train all the models with the same parameters: five epochs and a batch size of 32. However, for the Caltech101 dataset, more epochs were necessary, as training for only five epochs did not produce valuable outputs. As for the VGGNet model, the batch size had to be significantly decreased to avoid memory errors. Otherwise the batch size was too large for the available computing resources.

IV. RESULTS AND ANALYSIS

The quantitative and qualitative outcomes of the conducted experiments are presented and analysed in this section, including effectiveness of models, impact of dataset complexity on performance and difference between failure cases of more and less accurate models.

A. *Quantitative Results*

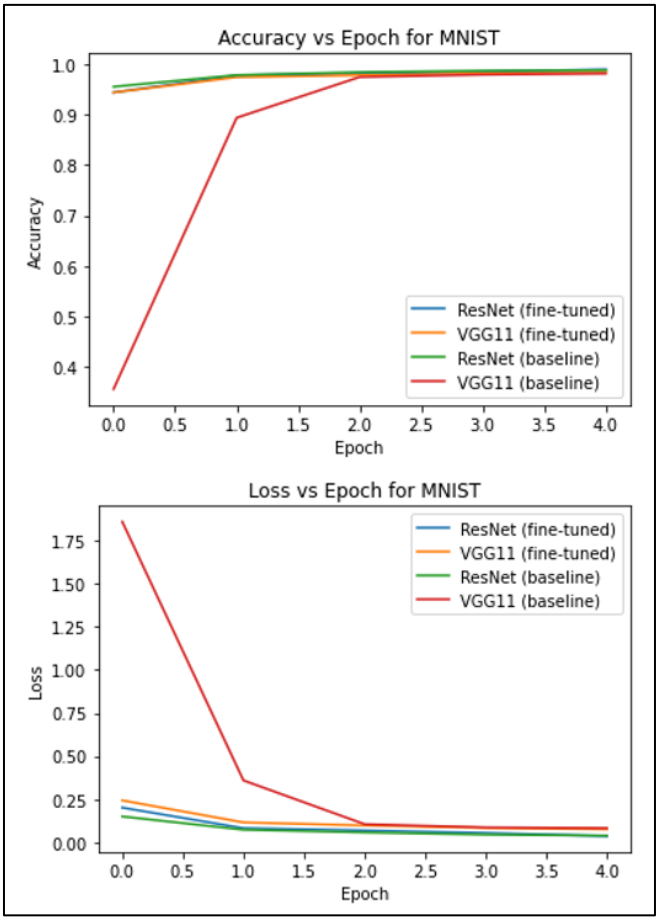

Fig. 3. Training progress over five epochs (*MNIST*)

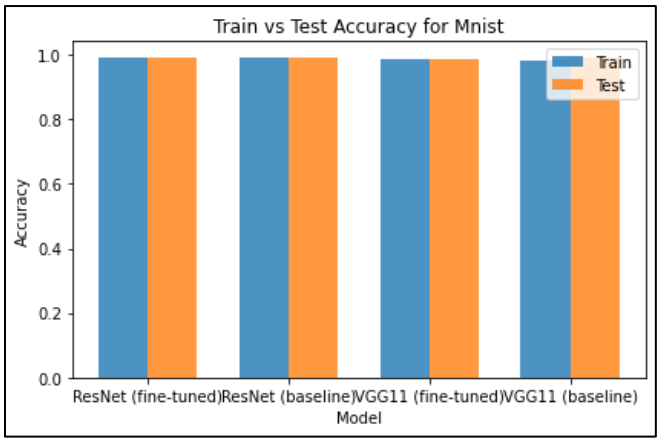

Fig. 4. Train and test accuracies compared (*MNIST*)

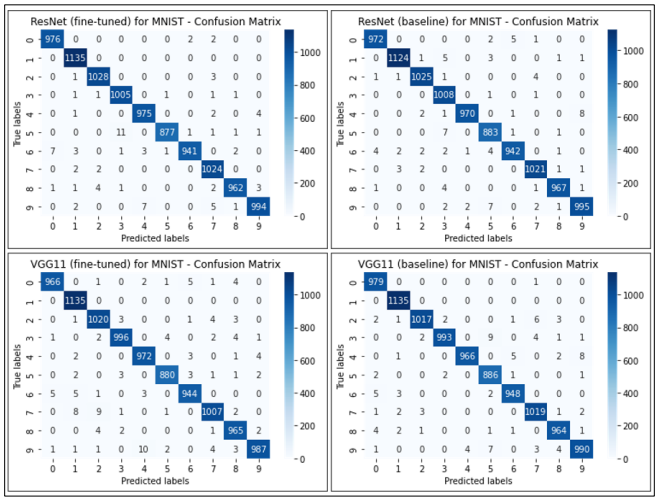

Fig. 5. Similar confusion matrix for each training variation (*MNIST*)

In the case of training the models on the MNIST dataset, all of them achieved very high scores very quickly (Fig. 3). The slowest one to converge was VGGNet trained from scratch, but it still achieved an accuracy around 98%. Moreover, as evidenced in Fig. 4, the models did not overfit, and the testing accuracy matched all the training accuracies. Fig. 5 shows similar mistakes made by the models, such as confusing 5s with 3s or 4s with 9s. They are confusing the most difficult cases. The results demonstrate excellent performance of both models on this dataset. Transfer learning helped the models converge to results even faster, which is visible in the VGGNet case in Fig. 3.

Training on Caltech101 was more challenging, and as a result, the superiority of some models was more evident. The fine-tuned ResNet achieved the highest accuracy fastest, followed by ResNet trained from scratch. The fine-tuned VGGNet achieved only 62%, while baseline achieved merely 9% accuracy. However, testing accuracies showed that ResNet had a greater tendency to overfit than VGGNet, perhaps due to small batch size. Results demonstrate the superiority of fine-tuning ResNet for complex tasks when there is low computational power available, and only short training is possible. Longer training would show if the VGGNet surpasses ResNet, and if its lack of overfitting persists.

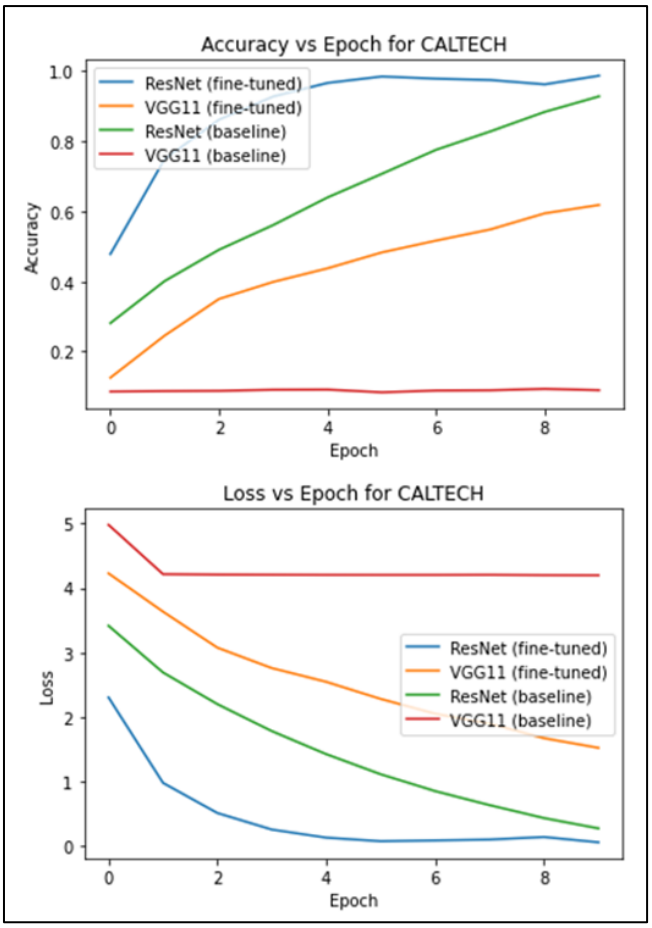

Fig. 6. Training progress over ten epochs (*Caltech101*)

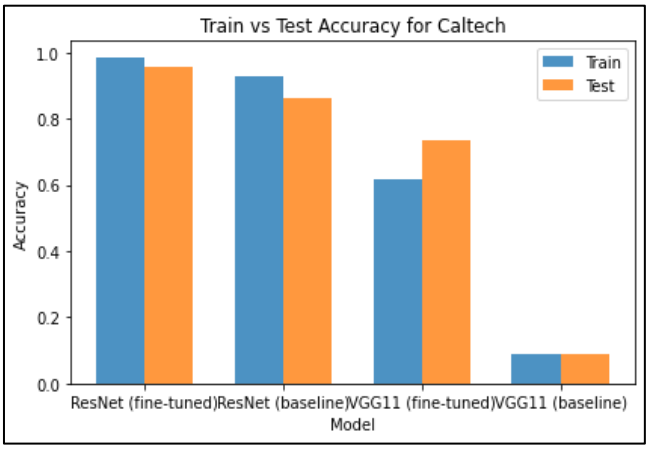

Fig. 7. Train and test accuracies compared (*Caltech101*)

TABLE II. PRECISION, RECALL AND F1-SCORE

Dataset Name	Model Name	Transfer Learning	*Metrics Scores*		
			Precision	*Recall*	*F1-Score*
MNIST	ResNet	yes	0.992	0.991	0.992
		no	0.991	0.991	0.991
	VGGNet	yes	0.987	0.987	0.987
		no	0.990	0.990	0.990
Caltech	ResNet	yes	0.940	0.934	0.931
		no	0.844	0.794	0.787
	VGGNet	yes	0.688	0.647	0.638
		no	0.001	0.010	0.001

For MNIST, all models achieved similar and very high precision and recall scores, as shown in Table II. On Caltech101, both ResNet models and the fine-tuned VGGNet model had a slightly lower rate of false positives than false negatives, whereas the opposite was true for VGGNet trained from scratch.

TABLE III. COMPUTATIONAL PERFORMANCE

Dataset Name	Model Name	Transfer Learning	*Average Training Performace*	
			Epoch Time	*It /Sec*
MNIST	ResNet	yes	55.442s	33.852
		no	55.092s	34.062
	VGGNet	yes	374.625s	20.020
		no	372.470s	20.142
Caltech	ResNet	yes	373.178s	0.582
		no	380.561s	0.571
	VGGNet	yes	420.067s	2.756
		no	402.724s	2.889

When analysing and comparing both models, it is important to keep in mind that the models were not trained in the same exact way, and VGGNet required smaller batch sizes. The difference in batch sizes might have affected the fact that ResNet always trained faster than VGGNet (Fig. 3; Fig. 6; Table III).

B. *Qualitative Results*

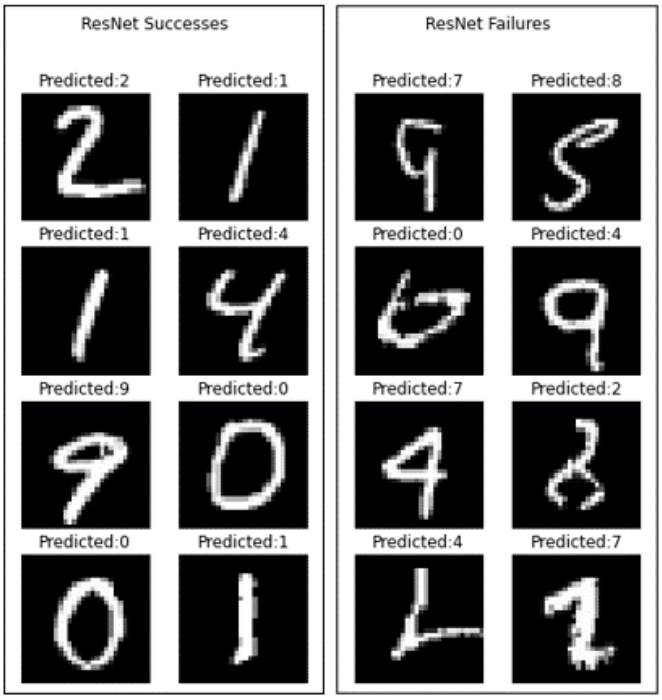

Fig. 8. Example cases with correct and with incorrect classification results of the fine-tuned ResNet model (*MNIST*)

The visualizations of successful and failed classifications for ResNet and VGG on MNIST (Fig. 8 and Fig. 9) show that both models make mistakes in more challenging cases, where the numbers start to look similar to each other. For example, both of them confused number six with number zero or number 7 with number 1.

With Caltech101 dataset, the failures become more apparent. The ResNet model (Fig. 10) made more justifiable errors eg. confusing yin yang with a panda, given that they both have similar black and white colors.

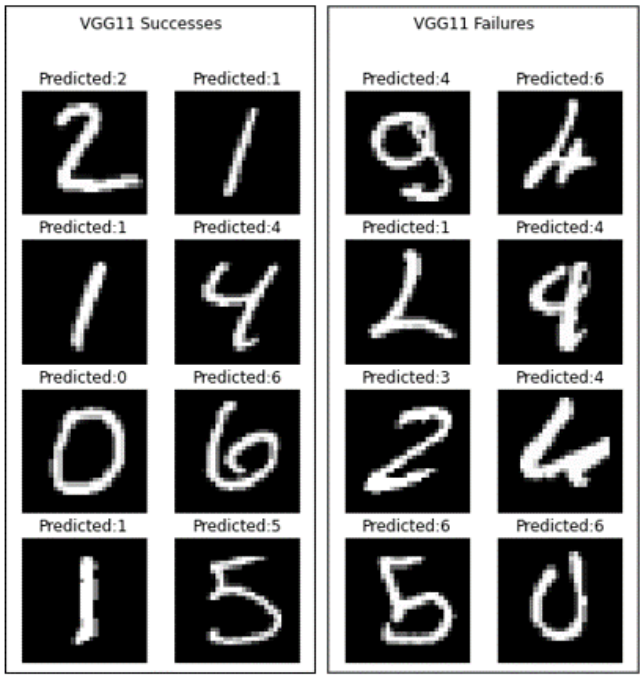

Fig. 9. Example cases with correct and with incorrect classification results of the fine-tuned VGGNet model (*MNIST*)

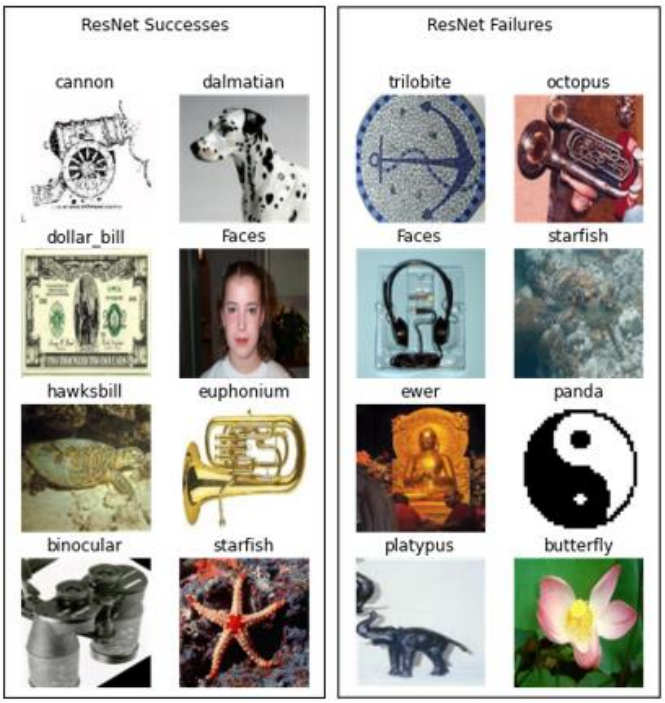

Fig. 10. Example cases with correct and with incorrect classification results of the fine-tuned ResNet model (*Caltech101*)

However, the failures of the VGGNet model are much more absurd. For example, as visible on Fig. 11, the model missclassified a flamingo as a sunflower, crab as a kangaroo, or starfish as an electric guitar. These examples clearly visualise the need for further training and improvement of this model.

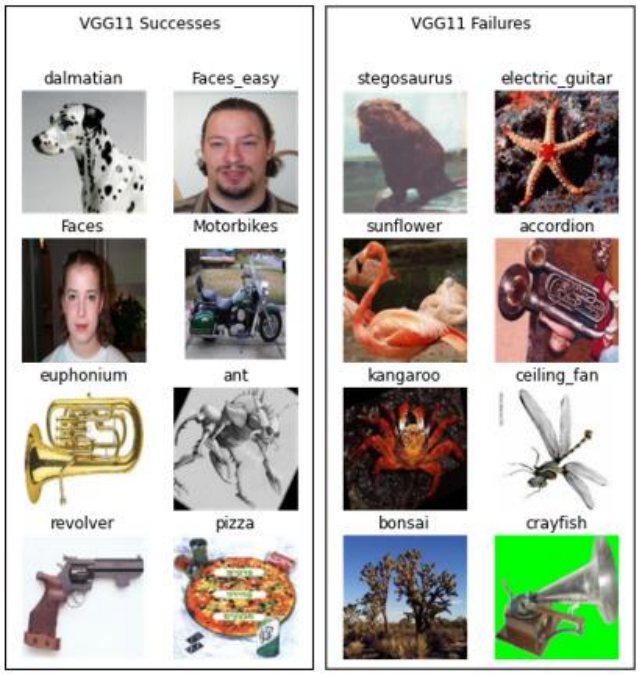

Fig. 11. Example cases with correct and with incorrect classification results of the fine-tuned VGGNet model (*Caltech101*)

V. Conclusions

In conclusion, the report evaluated the performance of ResNet and VGGNet models on two datasets, MNIST and Caltech101. The results demonstrate that both models performed well on the MNIST dataset, achieving high accuracy scores and not overfitting the data. However, when applied to the Caltech101 dataset, fine-tuning ResNet was found to be superior, especially when compared to training VGGNet from scratch. Although longer training would be necessary to determine whether VGGNet could surpass ResNet and maintain its lack of overfitting. Additionally, the batch size used in training may have affected the training speed and performance of the models. The study also revealed that both ResNet and VGGNet models made mistakes only on challenging cases on MNIST dataset, where the numbers start to look similar to each other. However, ResNet model made more justifiable errors than VGGNet on Caltech101 dataset, while the failures of the VGGNet model were much more absurd, highlighting the need for further training and improvement of this model. Overall, the study provides insights into the effectiveness of ResNet and VGGNet models in image classification tasks and proves the effectiveness of using DCNNs for this problem.

References

[1] Dalal, N. and Triggs, B. (2005) 'Histograms of Oriented Gradients for Human Detection', 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). doi:10.1109/cvpr.2005.177.

[2] Lowe, D.G. (1999) 'Object Recognition from Local Scale-Invariant Features', Proceedings of the Seventh IEEE International Conference on Computer Vision. doi:10.1109/iccv.1999.790410.

[3] Ojala, T., Pietikäinen, M. and Harwood, D. (1994) 'Performance evaluation of texture measures with classification based on Kullback discrimination of distributions', Proceedings of 12th International Conference on Pattern Recognition, 1, pp. 582–585. doi:10.1109/icpr.1994.576366.

[4] Chen, Y., Crawford, M.M. and Ghosh, J. (2004) 'Integrating Support Vector Machines in a Hierarchical Output Space Decomposition Framework', IEEE International IEEE International IEEE International Geoscience and Remote Sensing Symposium, 2004. IGARSS '04. Proceedings. doi:10.1109/igarss.2004.1368565.

[5] Webb, G.I. et al. (2011) 'Decision Tree', Encyclopedia of Machine Learning, pp. 263–267. doi:10.1007/978-0-387-30164-8_204.

[6] Breiman, L. (2001) 'Random Forests', Machine Learning, 45(1), pp. 5–32. doi:10.1023/a:1010933404324.

[7] Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2017) 'ImageNet classification with deep convolutional neural networks', Communications of the ACM, 60(6), pp. 84–90. doi:10.1145/3065386.

[8] Simonyan, K. and Zisserman, A. (2014) Very Deep Convolutional Networks for Large-Scale Image Recognition.

[9] He, K. et al. (2016) 'Deep Residual Learning for Image Recognition', 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) [Preprint]. doi:10.1109/cvpr.2016.90.

[10] Rawat, W. and Wang, Z. (2017) 'Deep convolutional neural networks for Image Classification: A Comprehensive Review', Neural Computation, 29(9), pp. 2352–2449. doi:10.1162/neco_a_00990.

[11] Deng, L., 2012. The MNIST Database of Handwritten Digit Images for Machine Learning Research. IEEE Signal Processing Magazine, 29(6), pp.141–142.

[12] Fei-Fei, L., Fergus, R. and Perona, P. (2004) 'Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories', 2004 Conference on Computer Vision and Pattern Recognition Workshop. doi:10.1109/cvpr.2004.383.

[13] Tan, C. et al. (2018) 'A Survey on Deep Transfer Learning', Artificial Neural Networks and Machine Learning – ICANN 2018, pp. 270–279. doi:10.1007/978-3-030-01424-7_27.