Image Classification on Different Types of Datasets Using Multiple Neural Network Architectures: A Comparative Study

Md. Saiful Bari Siddiqui*, Std. ID: N/A, BRAC University

***Abstract*- Perhaps the most crucial aspect of any task involving Neural Networks is recognizing exactly what sort of an architecture will produce the best results. This paper presents a comparative analysis on image classification tasks using different kinds of Neural Network architectures on datasets with different complexity levels. To carry out the task, Convolutional Neural Networks (CNN) with different structures and kernels and also simple Feed forward Neural Networks (ANN) were implemented using four famous datasets. In this study, CNNs consisting of 2 Convolutional layers followed by a Dense hidden layer achieved more than 99% accuracy and outperformed both simpler Feed forward networks and more complex CNNs with increased number of Convolutional layers and kernels in case of simple hand written digit classification using MNIST dataset. But as the complexity of the datasets increased, more and more complex neural network architectures were needed to achieve competitive results. The outcomes imply that the structure and complexity of data is to be taken into account to achieve the optimum performance. The code and implementations of this study are publicly available at the repository https://github.com/Saiful185/Model-Comparisons-on-Different-Datasets-for-Image-Classification .**

I. Background

From Object Recognition tasks to Natural Language Processing, Neural Networks are at the very center of Data Science and AI research works lately. One of the major domains that benefitted from the reemergence of Deep Neural Networks is OCR (Object Character Recognition) [1]. Handwritten Digit Recognition (HDR) is a snippet of OCR where instead of taking the whole character's data, HDR detects digits. Comparing to OCR, HDR is light and faster. In fields like medical, banking, student management, and taxation process, HDR possesses great flexibility.[2] HDR is a pretty basic yet tedious task to implement using traditional Machine Learning algorithms due to the distinctiveness in the style of writing. Neural Networks are particularly useful in these tasks because of the capacity to extract deep level features. The rise of CNNs [3] meant that the task is even simpler now. However, choosing the right architecture for a particular task is key to optimal performance. We discuss that in the context of HDR in this study with MNIST dataset.

Basic Image classification tasks can become much more complex compared to HDR due to the more diverse nature of the classes involved and also typically more closely spaced pixel values. We study that phenomenon using Fashion MNIST and then the colored CIFAR-10 dataset.

Classification tasks become even more tedious when the number of classes is increased. Not only does it become much more difficult to separate the classes from one another, but also the training sample size for each class becomes much smaller. We delve into that scenario too using CIFAR-100 dataset.

II. Datasets

Total 4 publicly available datasets were used in this study for comparative study. The datasets are: MNIST, Fashion-MNIST, CIFAR-10 and CIFAR-100.

The MNIST dataset (Figure 5) was used as a primary dataset to train the model, and it consists of 70,000 28X28 handwritten raster images from 250 different sources out of which 60,000 are used for training, and the rest are used for training validation. All the images are greyscale in MNIST dataset.

Fashion-MNIST [5] consists of 60,000 training images and 10,000 test images. It is an MNIST-like fashion product database (Figure 6). It was created as a direct replacement for MNIST dataset. Each image is in greyscale here too and associated with a label from 10 classes.

CIFAR-10 [6] is another popular dataset for image classification. It consists of 60,000 32x32 color images of 10 classes (each class is represented as a row). In total, there are 50,000 training images and 10,000 test images (Figure 7). The dataset is divided into 6 parts – 5 training batches and 1 test batch. Each batch has 10,000 images.

The CIFAR-100 [7] dataset is just like the CIFAR-10, except it has 100 classes containing 600 images each. There are 500 training images and 100 testing images per class. The 100 classes in the CIFAR-100 are grouped into 20 super classes. Each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs).

III. METHODOLOGY

A. Greyscale Handwritten Digit Recognition

Each image in MNIST [4] Dataset is of size 28X28. At the start of the task the images are reshaped and normalized as part of preprocessing. Then the model is constructed. We used 4 different Models to train this dataset. 2 of those models involved CNNs. One of the models used 2 and the other comprised of 3 Convolutional (8 to 64 kernels) and Maxpooling layer(2X2) pairs along with 1 dense hidden layer (512 or 256 neurons). The simple Feed forward model contained 2 hidden layers (512 & 256 neurons). One CNN and one MLP model summary is shown in Figure 1 and 2. After this we compile and train the model for 10-15 epochs using RMSprop as optimizing algorithm and 0.001 as learning rate. Finally, we measure accuracy, loss and other metrics to evaluate the model.

B. Greyscale Fashion/Clothing Classification

The Fashion MNIST dataset is identical to the MNIST dataset in terms of training set size, testing set size, number of class labels, and image dimensions. MNIST cannot represent modern computer vision tasks. That's why Fashion MNIST was created. Despite the similarities, Fashion MNIST is a bit more challenging compared to MNIST. So, we followed the same procedures as we did in case of Digit recognition. We used the same data preprocessing techniques. We used 3 different Models to train this dataset. 2 of those models involved CNNs. One of the models used 2 Convolutional (8 to 64 kernels) and Maxpooling layer(2X2) pairs along with 1 dense hidden layer (512 or 256 neurons). Another one of the models used 2 Convolutional (8 to 64 kernels) X2 and Maxpooling layer(2X2) triads along with 1 dense hidden layer (512 neurons). The simple Feed forward model contained 2 hidden layers (512 & 256 neurons). Due to the added complexity, more convolutional layers were added in one of the models to extract hidden features. We followed the VGG structure (two Conv layers followed by one Maxpooling). The other two models were used for comparison purposes. After this we compile and train the model for 10-15 epochs using RMSprop as optimizing algorithm and 0.001 as learning rate. Finally, we measure accuracy, loss and other metrics to evaluate the model.

C. Colored Object Recognition with 10 Classes

Modern computer vision tasks mostly involve colored images and videos. Colored images represent a much more complex scenario with 3 layers in each of the images for RGB colors. This new dimension means the classification task is much more complicated. CIFAR-10 dataset is great for testing the models under these circumstances. Each image in CIFAR-10 Dataset is of size 32X32X3. Usually, RGB images need more preprocessing compared to simple greyscale images. CIFAR-10 dataset is suitable for many data augmentation techniques too. But, for comparison purposes, we used the same data preprocessing techniques as we used before. We used 3 different Models to train this dataset too. 2 of those models involved CNNs. One of the models used 2 Convolutional (8 to 64 kernels) X2 and Maxpooling layer(2X2) triads along with 1 dense hidden layer (512 neurons) like we used in the Fashion MNIST case. The simple Feed forward model contained 2 hidden layers (512 & 256 neurons) like before. To deal with the added complexity with RGB images, we train another model (Figure 8 contains model summary) with 3 Convolutional (8 to 64 kernels) X2 and Maxpooling layer(2X2) triads along with 1 dense hidden layer (512 neurons). We also specify kernel initializer to he transform to take advantage of Relu activation, which was not needed in case of previous two datasets. After this we compile and train the model for 10-15 epochs using RMSprop as optimizing algorithm and 0.001 as learning rate. Finally, we measure accuracy, loss and other metrics to evaluate the model.

D. Colored Object Recognition with 100 Classes

Image classification tasks become much more difficult to learn when there are more classes within a limited dataset. CIFAR-100 presents an interesting scenario in this case. This is exactly the same dataset as CIFAR-10, but there are 100 classes instead of 10. So, there are only 500 images per class in the training set which makes the task a whole lot tougher. For comparison purposes, we used the same data preprocessing techniques as we used before. We used 2 different Models to train this dataset. 1 of those models involved CNNs. It used 3 Convolutional (8 to 64 kernels) X2 and Maxpooling layer(2X2) triads along with 1 dense hidden layer (512 neurons). We also specify kernel initializer to he transform to take advantage of Relu activation, like we did in the CIFAR-10 case. The simple Feed forward model contained 2 hidden layers (512 & 256 neurons) like before. After this we compile and train the model for 10-15 epochs using RMSprop as optimizing algorithm and 0.001 as learning rate. Finally, we measure accuracy, loss and other metrics to evaluate the model.

IV. RESULTS & OBSERVATIONS

A. Greyscale Handwritten Digit Recognition

We measured the training and validation accuracy for each epoch and plotted it to tune the model. A CNN consisting of 2 Convolutional layers followed by a Dense hidden layer achieves 99.11% accuracy with 0.0325 log loss on validation set. This is up there with the top performing models on this dataset. CNNs with 3 convolutional layers reach decent accuracy too but slightly less than that consisting of 2 Convolutional layers. The interesting fact is Simple Feed forward NN models also produce quite high accuracy scores (more than 98.5%) despite overfitting a bit more compared to CNN models. The model with no hidden layers, expectedly shows the worst accuracy scores by some margin. The overall performance metrics for different NN architectures are summarized in Table 1 & Figure 2 & 3.

It's evident that Neural networks, particularly CNNs perform very well at the task of Handwritten Digit Recognition due to the ability of extracting complex and sparse features, as we can see from the outputs of CNN layers in Fig. 4. It also makes sense that comparatively shallow CNNs perform better when it comes to relatively simple greyscale images like those in MNIST as the size is only 28X28 pixels, and more Maxpooling with no padding means losing pixel information in deeper layers. Also, 2 Convolutional and 1 Dense layer is enough to get almost all the features from these images.

B. Greyscale Fashion/Clothing Classification

Again, we measured the training and validation accuracy for each epoch and plotted it to tune the model. The CNN consisting of 2 Convolutional layers followed by a Dense hidden layer struggles to achieve even 90% accuracy with on validation set of this dataset. This shows the added difficulty in classifying a more complex dataset. To address the issue a more complex Neural network architecture is applied as we mentioned in methodology. This VGG like shallow model shows improvement as it achieves 92.04% accuracy with nearly 0.255 log loss. On the other hand, the feed forward model shows an accuracy of just over 86%.

These results show that despite having an exactly identical dataset like MNIST in terms of color, size and shape, Fashion MNIST needs a more complex CNN model to extract enough information to be classified properly due to its more diverse features relating to each class and closely spaced pixel values. It also demonstrates the fact that as image classification tasks get more and more complicated, simple feed forward models start performing worse.

C. Colored Object Recognition with 10 Classes

We started following the same approach as before. The same model that performed best in case of Fashion MNIST achieves only around 71% accuracy in CIFAR-10 dataset. This demonstrates the new dimension that colored images bring to the table and the added complexity. To tackle this, we not only add more Conv layers with zero padding to create a more sophisticated model that can extract the necessary features from the images, but also set he transform as the kernel initializer to take advantage of Relu activation. This new model achieves an accuracy of 75.23%, a significant improvement from previous approaches without using any data augmentation or regularization techniques. The simple feed forward model was also implemented. Its performance dropped drastically with the RGB image dataset, returning a poor accuracy of just 41.92%.

These results further show the need for deeper neural network architectures and the effectiveness of convolutional neural networks in modern image classification tasks. At this point simple feed forward models no longer produce decent results. Complex convolutional neural network architectures with more and more layers are needed as the dataset and classification task becomes more complicated. Added to that, proper optimization algorithms along with other techniques help a lot in generalizing the model and decreasing overfitting.

D. Colored Object Recognition with 100 Classes

In this final set of experiment, we once again follow the same procedures as before for comparison. The same model that achieved more than 75% accuracy in CIFAR-10 dataset could only achieve a 38.64% accuracy in case of CIFAR-100. This isn't so poor considering this is a 100-class classification. But this also implies the difficulty associated with classification tasks that involve a lot of discrete classes. We also implemented the simple feed forward model once again and it resulted in an unacceptable 17% accuracy.

These results show how more classes make the classification task much more complicated. Dropout, Regularization and Data augmentation techniques are needed along with much more complex CNN architectures for extracting deep level features in the image in order to achieve competitive results. This opens the door for further studies in this area. Transfer learning and training with added data are also some obvious candidates to build great classification models using these datasets.

V. CONCLUSIONS

Throughout this study, we experimented with different datasets that are used for very similar classification tasks. Still, all these datasets posed different levels of challenges and thus achieved optimum level of performance using models with different levels of complexity and depth. The more difficult the task is, the more complex a model is needed to carry out the classification properly. The complexity is not only in terms of more hidden layers, but also in using techniques like data augmentation, transfer learning, regularization etc. Our study focused on basic models with different datasets without using these techniques. Future research works can focus on the effect of using these techniques on different datasets and image classification tasks. Table-2 summarizes the performance of all the models using different datasets.

REFERENCES

[1] Junli Gu, Maohua Zhu, Zhitao Zhou, Feng Zhang, Zhen Lin, Qianfeng Zhang, Mauricio Breternitz. "*Implementation and evaluation of deep neural networks (DNN) on mainstream heterogeneous systems*" APSys '14: Proceedings of 5th Asia-Pacific Workshop on Systems, June 2014 Article No.: 12 Pages 1–7.

[2] S. Aly and A. Mohamed, **"***Unknown-Length Handwritten Numeral String Recognition Using Cascade of PCA-SVMNet Classifiers***"** in IEEE Access, vol. 7, pp. 52024-52034, 2019

[3] Y. Yin, W. Zhang, S. Hong, J. Yang, J. Xiong, and G. Gui, *"Deep Learning-Aided OCR Techniques for Chinese Uppercase Characters in the Application of Internet of Things,"* in IEEE Access, vol. 7, pp. 47043-47049, 2019.

[4] L. Deng, *"The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]"* in IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 141-142, Nov. 2012.

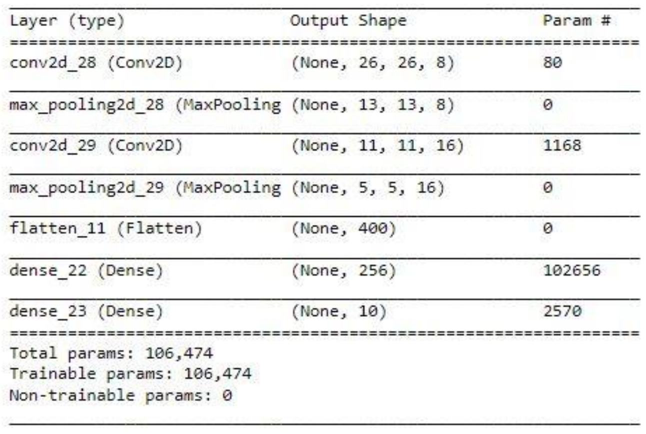
[5] H. Xiao, K. Rasul, and R. Vollgraf., *"Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms",* *http://arxiv.org/abs/1708.07747 ,2017*.

[6] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, *"CIFAR-10 & CIFAR-100"*, https://www.cs.toronto.edu/~kriz/cifar.html

[7] https://www.kaggle.com/benhamner/popular-datasets-over-time/code www.kaggle.com. Retrieved 2017-12-11

Models	**Training Accuracy**	**Validation Accuracy**	**Training Loss**	**Validation Loss**	**ROC_AUC Score**	**Cohen Kappa Score**	**F1 Score**	**Matthews Correlation Coefficient**
CNN with 2 Conv layers	**0.9972**	**0.9911**	**0.0096**	**0.0325**	**0.9999**	**0.9901**	**0.9910**	**0.9901**
CNN with 3 Conv layers	0.9954	0.9861	0.0143	0.0639	0.9997	0.9845	0.9861	0.9846
ANN with 2 hidden layers	0.9931	0.9797	0.0343	0.1805	0.9991	0.9774	0.9795	0.9775
ANN with no hidden layer	0.9299	0.9273	0.2807	0.3033	0.9932	0.9192	0.9263	0.9192

Table 1: Performance Metrics Summary for all Models using MNIST Dataset



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_28 (Conv2D)           (None, 26, 26, 8)         80
_________________________________________________________________
max_pooling2d_28 (MaxPooling (None, 13, 13, 8)         0
_________________________________________________________________
conv2d_29 (Conv2D)           (None, 11, 11, 16)        1168
_________________________________________________________________
max_pooling2d_29 (MaxPooling (None, 5, 5, 16)          0
_________________________________________________________________
flatten_11 (Flatten)         (None, 400)               0
_________________________________________________________________
dense_22 (Dense)             (None, 256)               102656
_________________________________________________________________
dense_23 (Dense)             (None, 10)                2570
=================================================================
Total params: 106,474
Trainable params: 106,474
Non-trainable params: 0
_________________________________________________________________
```

Figure 1: Model summary for CNN with 2 Conv layers

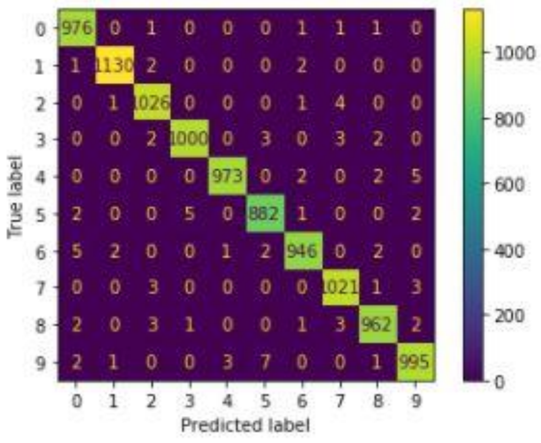


Figure 2: Confusion Matrix for CNN with 2 Conv layers (MNIST)

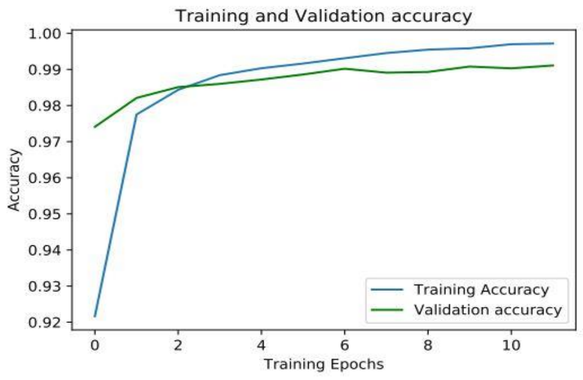


Figure 3: Validation Accuracy vs Epochs for CNN with 2 Conv layers

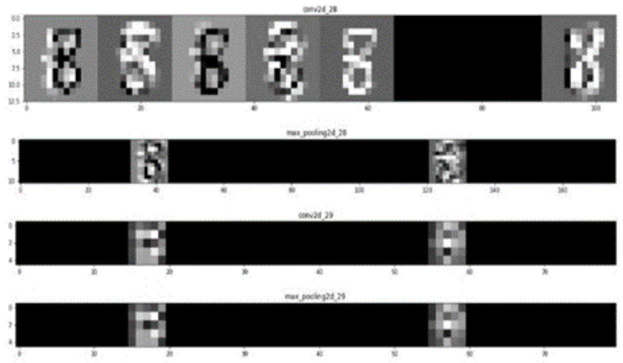


Figure 4: CNN Conv layers extracting sparse features from MNIST

Models \ Datasets	MNIST	Fashion MNIST	CIFAR-10	CIFAR-100
3 layered FC ANN Model	0.9796	0.869	0.4192	0.1641
CNN with 2 Conv layers	**0.9911**	0.8974	---	---
CNN with 2 2XConv layers	---	**0.9204**	0.7176	---
CNN with 3 2XConv layers	---	---	**0.7523**	**0.3864**

Table 2: Accuracy for Different Models on all the Datasets

Figure 5: MNIST Dataset Samples

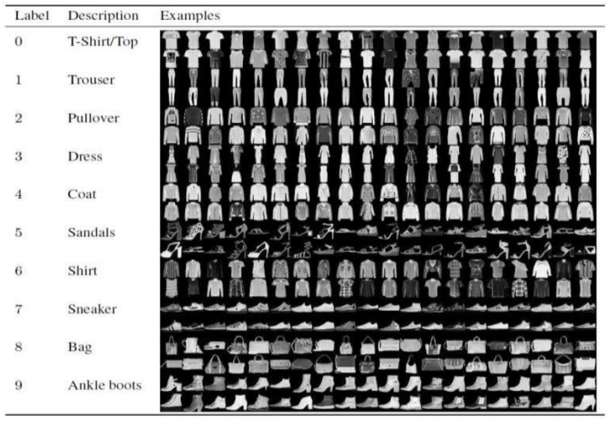
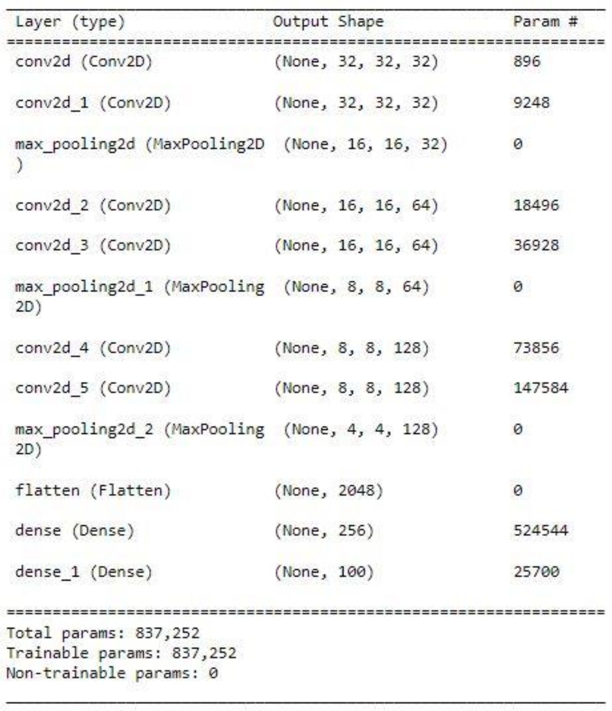


Figure 6: Fashion MNIST Dataset

Figure 7: CIFAR-10 Dataset



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 32)        896

conv2d_1 (Conv2D)            (None, 32, 32, 32)        9248

max_pooling2d (MaxPooling2D   (None, 16, 16, 32)       0
)

conv2d_2 (Conv2D)            (None, 16, 16, 64)        18496

conv2d_3 (Conv2D)            (None, 16, 16, 64)        36928

max_pooling2d_1 (MaxPooling   (None, 8, 8, 64)         0
2D)

conv2d_4 (Conv2D)            (None, 8, 8, 128)         73856

conv2d_5 (Conv2D)            (None, 8, 8, 128)         147584

max_pooling2d_2 (MaxPooling   (None, 4, 4, 128)        0
2D)

flatten (Flatten)            (None, 2048)              0

dense (Dense)                (None, 256)               524544

dense_1 (Dense)              (None, 100)               25700

=================================================================
Total params: 837,252
Trainable params: 837,252
Non-trainable params: 0
_________________________________________________________________
```

Figure 8: Best Performing Model on CIFAR Datasets