Machine Leaning Models Comparison and Training on Fashion MNIST Dataset

1st Zeiad Moawad
Computer Science
University of Prince Edward Island, Cairo Campus
Cairo, Egypt
zeiadmoawad@gmail.com

2nd Ahmed Salem
Data Analytics
University of Prince Edward Island, Cairo Campus
Cairo, Egypt
ahmedkhaledmohamedsalem@gmail.com

***Abstract*—This paper aims to find a machine learning model that can achieve the highest testing accuracy while minimizing the computational time for the fashion MNIST dataset. Such models can help other researchers find a suitable model not just for fashion MNIST datasets or fashion datasets in general but also for other Image based datasets, as Image classification is one of the fundamental bases for Artificial Intelligence and can be used for multiple purposes like self-driving vehicles and face and object recognition. There are two models that were found close to this objective, which is the CNN model with (94.38%) testing accuracy and the MobileNetV2 +GAP-Dense classifiers model with (93.76%) testing accuracy. It was found that CNN deals with Fashion MNIST well and produces high accuracy. Computation time is less than that of other machine learning models, but MobileNetV2 + GAP-Dense classifiers may be less prone to overfitting and perform better on unseen data.**

***Index Terms*—Image classification, Convolutional Neural networks, MobileNetV2, Fashion MINST**

I. Introduction

First, we need to understand what the dataset contains. The Fashion MNIST dataset [1] contains 70000 images, 60000 for training, and 10000 for testing. Each image is 28 x 28 pixels and on a grey scale. The dataset consists of 10 class labels (shirt, T-shirt, Sandal, pullover, dress, coat, sneaker, bag, ankle boot, trousers). Each pixel contains values from 0 to 255. Then, we need to understand why we are doing a machine learning model, which model we choose, and how to implement it. We are implementing a Machine Learning model to classify each image to the class label it belongs from the 10 class labels. This paper utilized two models: a multilayer CNN model and the MobileNetV2 + GAP-Dense classifiers model. The objective is to find the model with the best testing accuracy and the lowest computational time while avoiding overfitting. The MobileNetV2 model is meant to be less computationally expensive, but the setup we utilized to achieve higher accuracy was found to take a longer time to train, which makes the multi-layer CNN model easier and take a shorter amount of time to train. 1

.

Fig. 1. Dataset images sample

II. Literature Review

In previous papers, we indicated that MobileNetV2 achieves higher accuracy than CNN but takes a little longer regarding computational time. In [2], it was found that on the fashion Minst dataset, an accuracy of (88.91%) and it took 12 seconds for runtime using the CNN model while MobileNetV2 achieved an accuracy of (92.91%) and took 98 seconds for runtime. It was also indicated in [2] that resizing was essential for the MobileNetV2 model to ensure that data loss is minimized as the input dimension in the original model of MobileNetV2 is 224x224 [2]. In another paper [3], it was also found that adding batch normalization to the CNN model increased the test accuracy of the model, as the accuracy of a CNN model with two convolutional layers and each layer followed by one maxpooling layer and then adding a dense, dropout and dense layers was (91.17%) but when adding batch normalization before each convolutional layer increases the test accuracy to (92.53%).

III. Background

A. Convolutional Neural Networks (CNNs)

This special type of neural network is a class of deep learning models that excel in visual data analysis; CNNs are particularly effective for tasks involving image classification, object detection, etc. [4]

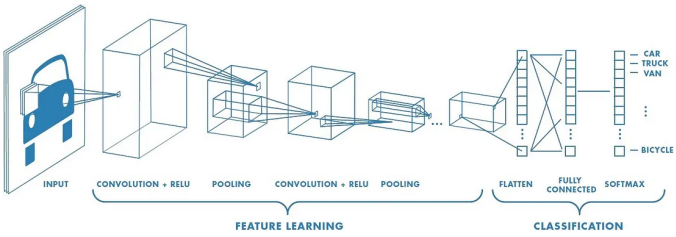

Fig. 2. CNN arch

1) Convolutional layers: The main component of a CNN is the convolutional layer, which performs the convolution operation on the inputted training dataset. Using filters (kernels) features like edges, textures, and patterns are extracted in the input image, creating feature maps as abstractions of the original image by learning spatial hierarchies in images.

2) Activation Functions: Complex patterns result in better model training; that's where an activation function after each convolution is applied, as it introduces non-linearity into the network, which causes CNNs to understand complex patterns and make decisions based on the non-linear relationships. ReLU (Rectified Linear Unit), GELU (Gaussian error linear unit), and Sigmoid are examples of activation functions, with ReLU being the most frequently used.

3) Pooling layers: A technique used to downsample feature maps, reducing the spatial dimensions while retaining essential features minimizes computational requirements, preventing overfitting. It is most commonly used as max-pooling, which selects the maximum value from the region.

4) Fully connected (Dense) layers: Once the convolutional and pooling layers have extracted the relevant features and passed through a flattening layer before passing them to the fully connected layers, they interpret learned features and output a prediction for the classification task.

B. MobileNetV2

A lightweight convolutional neural network architecture designed for resource-constrained devices like mobile phones and IoT devices, with a primary focus on achieving high accuracy at low computational complexity and memory usage. [5]

1) Key Design Principles:

- Depthwise Convolution: A single convolution filter per input channel is applied to reduce the computational cost.
- Pointwise COnvolution: A 1x1 convolution is used to recombine the depthwise features into the desired output channels.

2) Inverted Residuals (BottleNeck Blocks):

- Expansion phase: This input is first expanded (using pointwise convolution) by a factor of 4 or 6, typically to a higher-dimensional feature space, which aids in the network extracting rich feature representations.
- Depthwise COnvolution: Following, the depthwise convolution inputs the spatial features from the expanded space.
- Projection Phase (Linear Bottleneck): At the end, to preserve the compact representation nature, the output is projected back into a lower-dimensional space. Using linear bottlenecks (without ReLU) guarantees no loss of critical information in low-dimensional spaces.

3) Classification Head: The pre-built model is then passed through a global average pooling layer, followed by a fully connected (Dense) layer for classification.

IV. Results and Analysis

Both utilized models were set to a normalization scale of 0 - 1 and have been transformed from grayscale to RGB, as well as resizing the images from 28x28 to 64x64, which are requirements for the MobileNetV2-based model since it does not support images less than 32x32 and none RGB images. The Multi-Layer CNN model has also seen better results when normalization, transformation, and resizing are applied to the data. Adam optimizer was utilized in the setup to aid in the training with a learning rate of 0.001, and the categorical cross-entropy with label smoothing of 0.1 loss function was used for multiclass classification to encourage the model to predict probabilities closer to the actual distribution.

A. Multi-layer CNN model

When implementing our CNN model we built 3 convolutional layers with relu activation function each layer is followed by batch normalization and maxpooling of (2,2) and dropout. First layer has 64 neuron and dropout value of 0.2. Second layer has 128 neurons and dropout value of 0.3. third layer has 256 neurons and dropout value of 0.4. then those 3 layers are connected to flatten layer to reshape the multidimensional values to a vector fo outputs. this vector is connected to the dense layer of 128 units with relu activation function followed by batch normalization and dropout of 0.5 the output values are then connected to the final layer (output layer) we used dense layer of 10 units using softmax activation function to ensure that the values in the last neurons are 10 probability values representing the probability of an image belonging to each of the 10 class labels.dropout and batch normaization helps with generalizing the model and avoiding overfitting. The model training reached the best outcome achieved is at epoch 113.

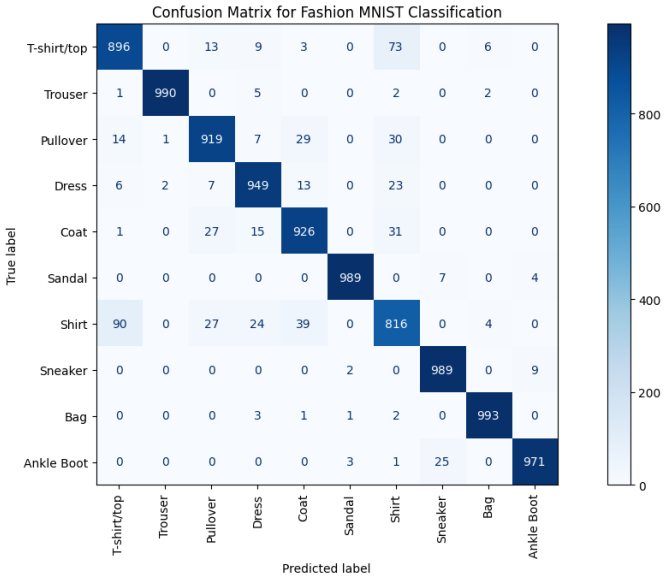

Fig. 3. CNN Confusion Matrix

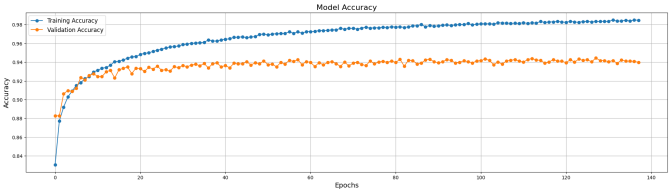

Fig. 4. CNN Loss Plot

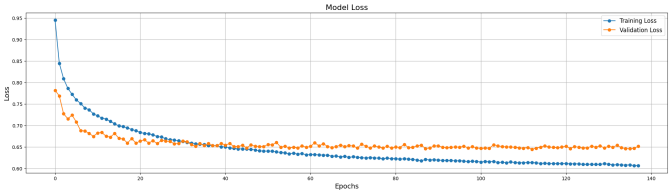

Fig. 5. CNN Loss Plot

B. *MobileNetV2 + Gap Dense Classifier*

MobileNetV2 was set as the base model, and a Gap Desne layer was added to translate the outputs into the 10 dataset classes. The trainer was set with an early stop of 30, 150 epochs, with a batch size of 16, and has trained until the 51st epoch, where the early stop has ended the training of the model and selected the best weights at the 21st epoch. The GELU activation function was utilized in the GAP dense classifier later, which yielded better accuracy than using ReLU, but using GELU made it more complex, which negatively impacted the computational complexity, making it take longer to train, followed by batch normalization and dropout of 0.5 the output values are then connected to the final layer (output layer) we used dense layer of 10 units using softmax activation function to ensure that the values in the last neurons are 10 probability values representing the probability of an image belonging to each of the 10 class labels [6].

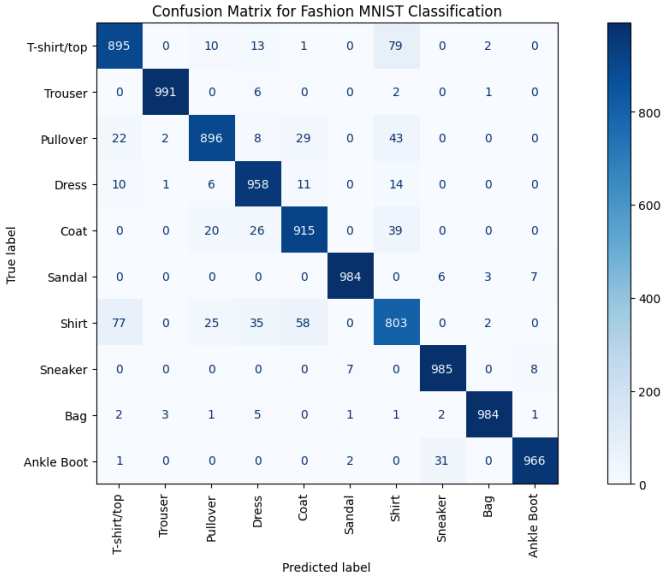

Fig. 6. MobileNetV2 + Gap Dense Classifier Confusion Matrix

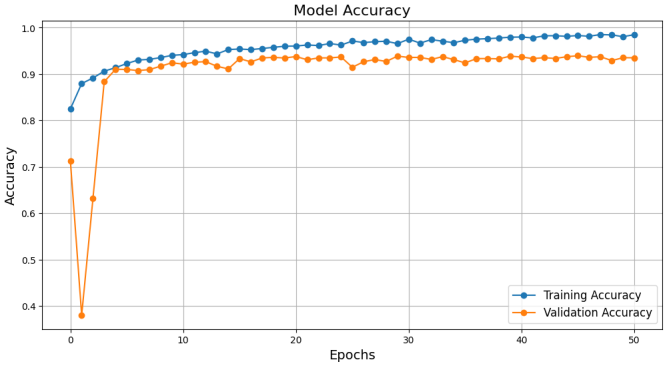

Fig. 7. MobileNetV2 + Gap Dense Classifier Accuracy Plot

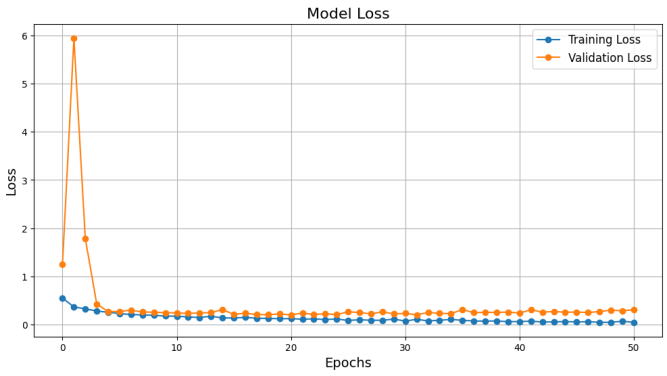

Fig. 8. MobileNetV2 + Gap Dense Classifier Loss Plot

V. Conclusions and Discussion

A. model conclusion

1) CNN model: The CNN model explained in this paper achieved (94.38%) test accuracy and f1 score of 0.94) and Took 53 mins to train 138 epochs on NVIDIA GeForce RTX 4060 GPU.

2) MobileNetV2+GAP-dense classifier: the MobileNetV2+GAP-dense classifier model achieved (93.76%) test accuracy and f1 score of 0.94 and Took 35 mins to train 51 epochs on NVIDIA GeForce RTX 4060 GPU.

3) conclusion: the implemented CNN model achieved higher accuracy than MobileNetV2+GAP-dense classifier model and has less computational power but, it's more prone to overfitting than MobileNetV2+GAP-dense classifier model.

B. Insights and Recommendation

The chosen models performed very well, considering the complexity of the training dataset, which yielded great results. However, there is room for improvement. The dataset has a few wrongly labeled examples, confusing the model; developing a way to detect incorrectly labeled images considered outliers would yield better results.

C. Future Research

It was noticed that the data may has misclassified images which contribute in model accuracy therefore Implementing an outlier-detecting algorithm to correct or remove mislabeled images in the dataset will significantly improve the performance of the models during training and testing.

References

[1] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[2] W. Di, "A comparative research on clothing images classification based on neural network models," in *2020 IEEE 2nd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, 2020, pp. 495–499.

[3] S. Bhatnagar, D. Ghosal, and M. H. Kolekar, "Classification of fashion article images using convolutional neural networks," in *2017 Fourth International Conference on Image Information Processing (ICIIP)*, 2017, pp. 1–6.

[4] K. O'Shea, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015.

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[6] M. Lee, "Gelu activation function in deep learning: A comprehensive mathematical analysis and performance," 2023. [Online]. Available: https://arxiv.org/abs/2305.12073