Intel Unnati Industrial Training - "Conquering Fashion MNIST with CNNs using Computer Vision"

Team : Vision
Authors : Vishnu T, Akhil Ashok R
Date of Submission : 14/07/2023

Abstract

The project aims to develop a convolutional neural network (CNN) model for accurately classifying images from the Fashion MNIST dataset. Computer vision techniques and concepts will be employed to enable the model to interpret and understand visual data, specifically focusing on clothing items.

1 Introduction

Our objective was to accurately classify fashion items into their respective categories. To optimize the model's performance and inference time, we incorporated Intel optimization techniques. We present the methodology, experiments, and outcomes, highlighting the benefits of leveraging CNN algorithms and Intel optimization in fashion item classification. The subsequent sections delve into the details of our approach, results, and analysis, aiming to inspire advancements in computer vision and deep learning models.

2 Motivation behind the problem

The Fashion MNIST dataset presents a suitable alternative to the traditional MNIST dataset, which mainly focuses on handwritten digit recognition. Fashion MNIST specifically targets the fashion industry, providing a more relevant and challenging dataset for fashion item classification. By utilizing this dataset, we can explore the applicability of CNNs to real-world fashion-related tasks.

3 Our Approach

3.1 Data augmentation

Certain data augmentation technique that involves applying random transformations to the training data, such as rotation, zooming, shearing, shifting, and flipping are used. By introducing these variations, data augmentation helps increase the size of the training dataset and improves the model's ability to generalize and handle real-world variations. It also acts as a form of regularization, preventing overfitting and improving the model's performance and robustness.

3.2 Training

We used TensorFlow and Keras frameworks for efficient neural network development. The model is a sequential neural network architecture designed for image classification tasks. It starts with a series of convolutional layers with increasing filter sizes and uses the ReLU activation function to introduce non-linearity. Batch normalization is applied after certain layers to normalize the outputs and improve training stability. Dropout layers are included to prevent overfitting by randomly deactivating a fraction of neurons during training. Max pooling layers are used for spatial downsampling. The flattened output is then passed through fully connected layers with ReLU activation. Batch normalization and dropout are again applied before the final dense layer with softmax activation, which produces the class probabilities. This model architecture leverages the power of convolutional neural networks for feature extraction from images and incorporates regularization techniques for better generalization and accuracy.

4 Model Architecture

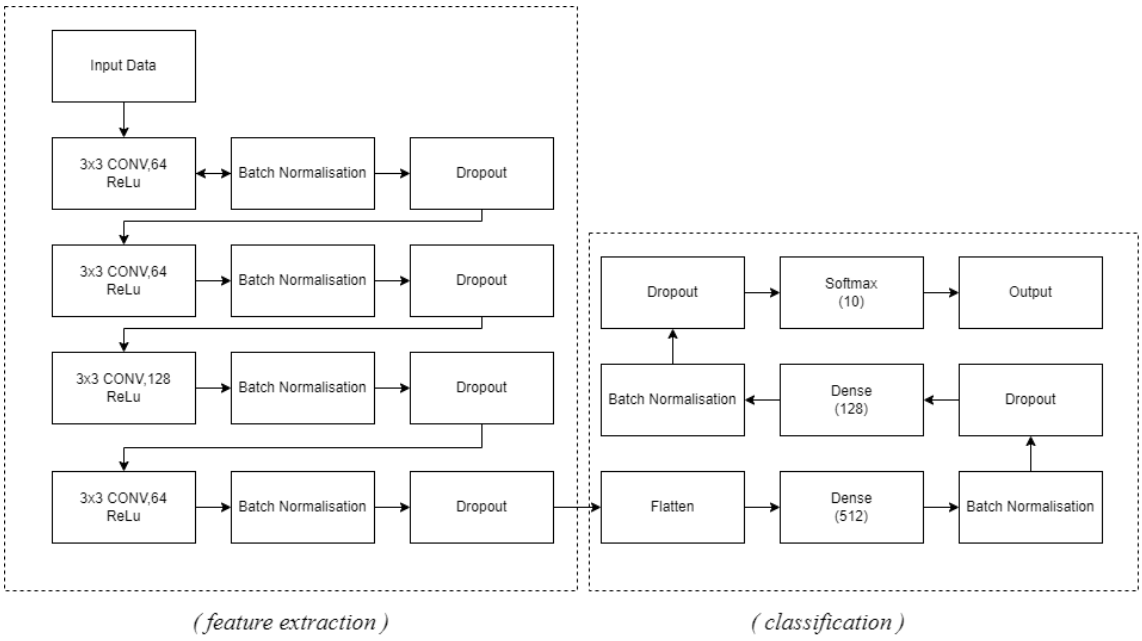


Figure 1: CNN Model architecture

5 Results

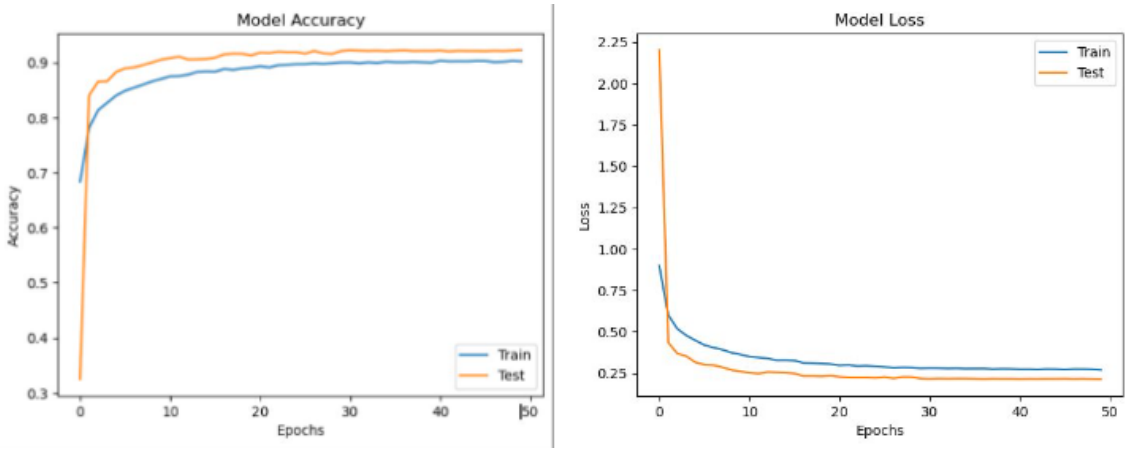
Our model achieved an impressive accuracy of 92% on the test dataset, demonstrating its effectiveness in accurately classifying the images. This high accuracy reflects the model's ability to learn and generalize well from the training data to make accurate predictions on unseen instances. The achieved accuracy serves as a strong indicator of the model's performance and its capability to handle the image classification task effectively.

5.1 Classification Report

The classification report provides important metrics like precision, recall, F1-score, and support for each class, giving insights into the model's ability to correctly classify instances. Analyzing the report helps identify the model's strengths and weaknesses for different classes, highlighting areas of good performance and possible challenges.

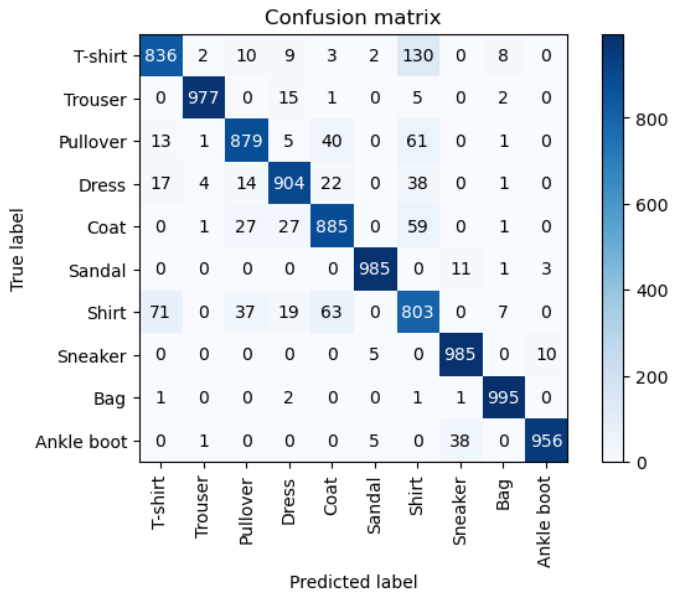
	precision	recall	f1-score	support
T-shirt	0.89	0.84	0.86	1000
Trouser	0.99	0.98	0.98	1000
Pullover	0.91	0.88	0.89	1000
Dress	0.92	0.90	0.91	1000
Coat	0.87	0.89	0.88	1000
Sandal	0.99	0.98	0.99	1000
Shirt	0.73	0.80	0.77	1000
Sneaker	0.95	0.98	0.97	1000
Bag	0.98	0.99	0.99	1000
Ankle boot	0.99	0.96	0.97	1000
accuracy			0.92	10000
macro avg	0.92	0.92	0.92	10000
weighted avg	0.92	0.92	0.92	10000

5.2 Learning curves



In the learning curves, the train curves and validation curves closely align with each other. This indicates that the model is performing well and not overfitting to the training data. When the train accuracy and test accuracy are close, it suggests that the model has learned to generalize well to unseen data, which is crucial for its performance on real-world scenarios.

5.3 Confusion Matrix



The confusion matrix provides a concise summary of the model's predictions compared to the actual labels. By analyzing the confusion matrix, we gain valuable insights into the model's performance for each class. In particular, we observe that our model encounters some difficulty in distinguishing between shirts and t-shirts. This indicates that there might be similarities in the visual characteristics of these two classes, leading to occasional misclassifications.

6 Intel Optimisations

6.1 Intel Extension for TensorFlow

The inclusion of Intel Extension for TensorFlow resulted in a significant reduction in training time for the model. Without the extension, each epoch took approximately 250 seconds to complete, whereas with the extension, the training time was reduced to just 40 seconds. This corresponds to an impressive improvement of 84% in training time.

6.2 OpenVINO Model Optimizer

The utilization of the OpenVINO Model Optimizer resulted in a significant reduction in the inference time of the model. Prior to converting the model into the Intermediate Representation (IR) format using the OpenVINO Model Optimizer, the inference time for each instance was approximately 5.2 seconds. However, after the conversion, the inference time was drastically reduced to just 1.5 seconds. This corresponds to a remarkable improvement of approximately 70% in inference time

7 Refferences

- https://www.tensorflow.org/tutorials/images/cnn
- https://www.tensorflow.org/datasets/catalog/fashion_mnist
- https://intel.github.io/intel-extension-for-tensorflow/latest/get_started.html
- https://docs.openvino.ai/2023.0/home.html