

**Course Title:Machine Learning**

**Assignment1: Fashion-MNIST Classification**

Github: <https://github.com/hadushtt/machineLearningAssigment1.git>

**Mekelle Institute of Technology**

**Department of Information**

Submission Date:5/2/2025 G.C

Submitted To: Ins Goitom A.

**Mekelle University**

# Fashion-MNIST Classification Report

**1.** Introduction

The objective of this project is to develop a machine learning model capable of accurately classifying images of fashion articles from the Fashion-MNIST dataset. Accurate classification of fashion items can be valuable for various applications, including e-commerce, inventory management, and fashion recommendation systems.

**2**. Data Exploration (EDA)

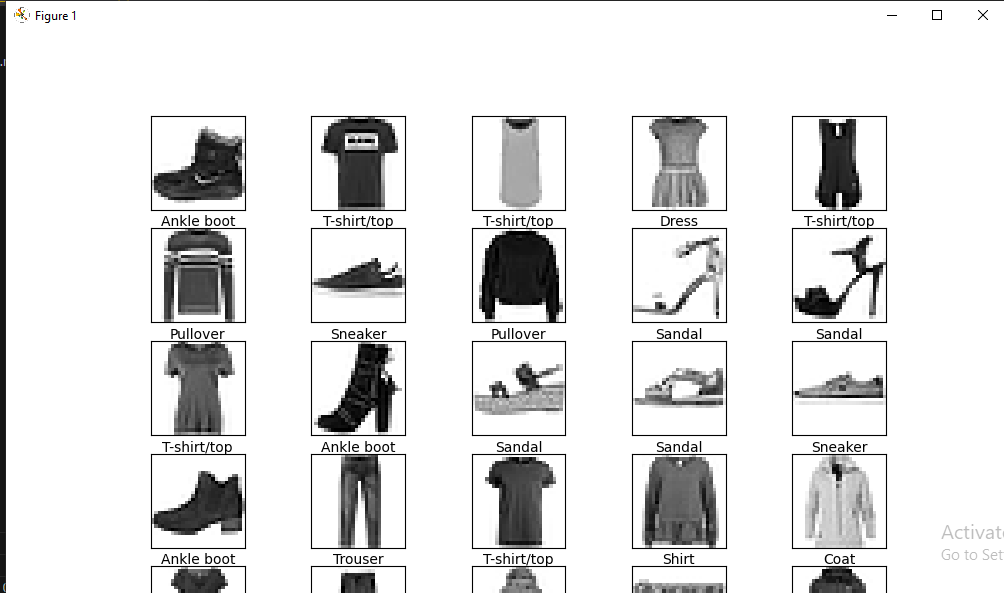
**2.1** Dataset Description

The Fashion-MNIST dataset comprises grayscale images of 10 different fashion article categories. Each image is 28x28 pixels. The dataset is split into a training set of 60,000 images and a test set of 10,000 images. For this project, a subset of 10,000 images from the training set was used, as per the assignment's recommendation. The 10 categories are:

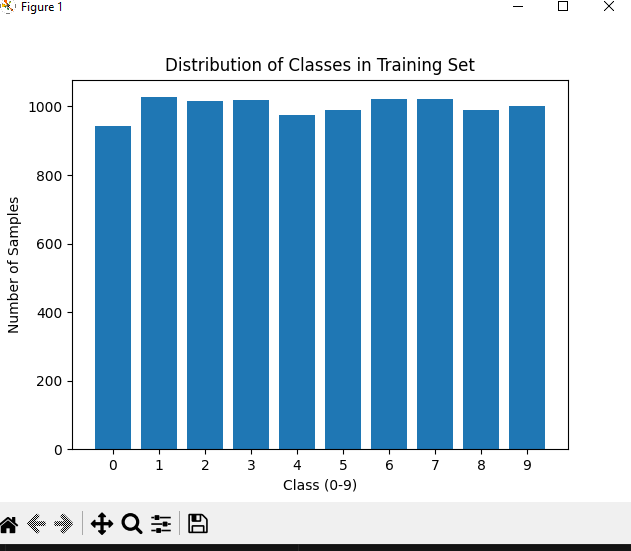
* 0: T-shirt/top
* 1: Trouser
* 2: Pullover
* 3: Dress
* 4: Coat
* 5: Sandal
* 6: Shirt
* 7: Sneaker
* 8: Bag
* 9: Ankle boot

**2.2** Data Visualization

* **Sample Images:**



* *This visualization provides a glimpse into the variety of fashion items present in the dataset.*
* **Class Distribution:**

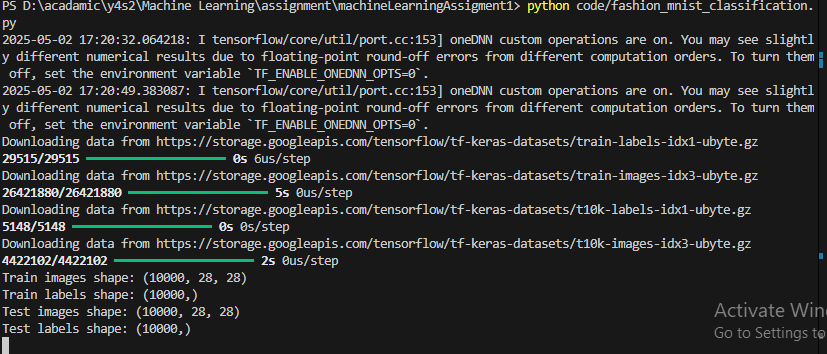


* *This chart illustrates the distribution of samples across the 10 classes. Ideally, the classes should be relatively balanced to avoid biasing the model towards dominant classes.*

**3.** Methodology

**3.1** Preprocessing Steps

The following preprocessing steps were applied to the Fashion-MNIST dataset:

* **Normalization:** Pixel values were normalized by dividing each pixel value by 255.0. This scaling ensures that all pixel values fall within the range of 0 to 1. Normalization helps to improve the training stability and convergence of the model.
* **Reshaping:** The training and testing images were reshaped to include a channel dimension, transforming them from (num\_samples, 28, 28) to (num\_samples, 28, 28, 1). This was necessary to feed the images into the Convolutional Neural Network (CNN) model, which expects input with a channel dimension, even for grayscale images. like the picture.

**3.2** Algorithm Selection and Implementation

A Convolutional Neural Network (CNN) was chosen as the classification algorithm for this project. CNNs are well-suited for image classification tasks due to their ability to automatically learn hierarchical features from images. The specific CNN architecture used is as follows:

* Convolutional Layer 1: 32 filters, 3x3 kernel, ReLU activation
* Max Pooling Layer 1: 2x2 pool size
* Convolutional Layer 2: 64 filters, 3x3 kernel, ReLU activation
* Max Pooling Layer 2: 2x2 pool size
* Flatten Layer: Flattens the 2D feature maps to 1D
* Dense Layer 1: 128 units, ReLU activation
* Dense Layer 2 (Output Layer): 10 units (for 10 classes), Softmax activation

The model was implemented using the TensorFlow/Keras library in Python.

**4.** Model Training

The CNN model was trained using the following hyperparameters and training strategy:

* **Optimizer:** Adam
* **Loss Function:** Sparse Categorical Crossentropy
* **Epochs:** 10
* **Batch Size:** 32 (This can be added)
* **Validation Split:** 20% of the training data was used for validation.

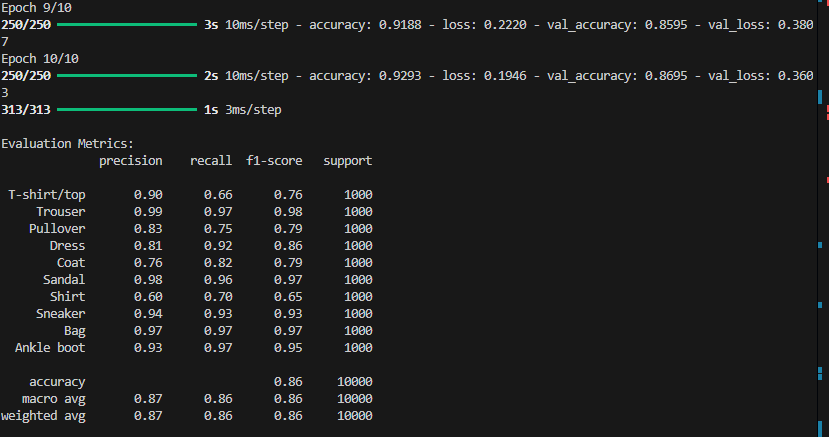
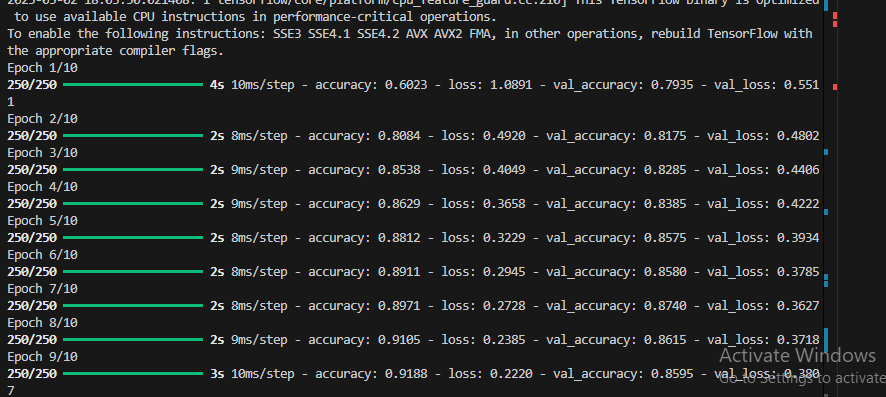
The Adam optimizer was chosen for its efficiency. Sparse categorical crossentropy was used as the loss function because the labels are integers. The model was trained for 10 epochs, with 20% of the training data held out as a validation set to monitor performance during training and prevent overfitting.

**5.** Results and Evaluation

**5.1** Evaluation Metrics and Performance Interpretation

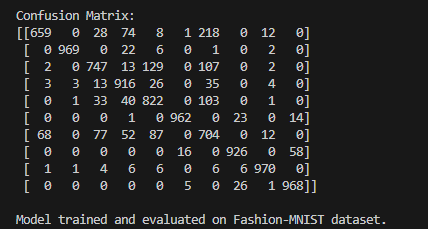
The performance of the trained model was evaluated on the test set using the following metrics:

* Accuracy, Precision, Recall, F1-score:



* *Accuracy measures the overall correctness of the model's predictions. Precision indicates the model's ability to correctly identify positive cases, while recall measures its ability to find all positive cases. The F1-score is the harmonic mean of precision and recall, providing a balanced measure of performance.*

**5.2** Confusion Matrix



* *The confusion matrix provides a detailed breakdown of the model's predictions, showing the number of correct and incorrect predictions for each class. It helps to identify which classes the model tends to confuse with each other.*

**6.** Conclusion and Reflections

**6.1** Summary of Findings

The CNN model achieved an accuracy of Report accuracy on the Fashion-MNIST test set. The classification report and confusion matrix provide further insights into the model's class-wise performance.

**6.2** Challenges Faced

* Computational resources limited the ability to experiment with more complex model architectures or longer training times.
* Balancing precision and recall across all classes proved to be a challenge.

**6.3** Possible Improvements

* Experiment with different CNN architectures, hyper parameters, and regularization techniques to potentially improve performance.
* Explore data augmentation techniques to increase the diversity of the training data and reduce overfitting.
* Investigate class weighting or other methods to address any class imbalance issues.