ECE 9202: Advanced Image Processing - Assignment 1

Amina Tabassum - 251390976
atabass4@uwo.ca

Thisali Senarath - 251469334
tsenarat@uwo.ca

Xiang Li - 251130633
xli2824@uwo.ca

Feb 8th, 2025

1 Introduction

This assignment aims to explore image classification using a Multilayer Perceptrons (MLP) on the Fashion MNIST dataset, a collection of images representing various clothing items and accessories. Two different approaches for training an MLP are explored: using SIFT feature extraction and using raw image pixels as direct input to the model.

2 Dataset

The Fashion MNIST dataset is used for this assignment . It contains 70,000 images classified into 10 categories i.e., T-shirt/top, Trouser, Pullover, Dress, Coar, Sandal, Sneaker,Bag,Ankle boot . The dataset is divided into 60,000 images for training and 10,000 images for testing.

3 Methodology

First , the dataset is loaded and few images are visualized for dataset exploration as shown in Figure 1. It is follwed by features extraction using Scale Invarient Feature Transform (SIFT).

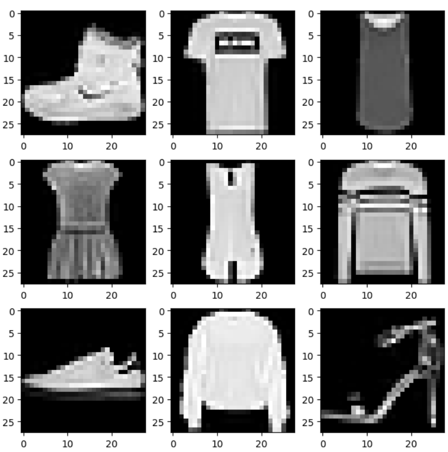


Figure 1: Visualization of nine random images from training dataset

3.1 Task 1 : SIFT Features Based MLP

3.1.1 SIFT

SIFT is a computer vision technique used for extracting features that are robust to scale and rotation invariance. It works by identifying keypoints based on their local intensity extrema and computing descriptor vectors to capture information around those keypoints. To extract features, images are first loaded. Since these are already grayscale, there is no need to change their color. OpenCV's SIFT-create() object is initialized, and the 'detectAndCompute()' function is used to extract keypoints and descriptors. This is followed by computing the average of the descriptor vectors to form a 128-dimensional feature vector. If keypoints are not detected, the descriptor vector is filled with zeros. The extracted SIFT feature vectors are standardized using StandardScaler() from scikit-learn by scaling them to zero mean and unit variance to ensure that features are centered and scaled, which helps improve model stability and training efficiency. These features are then fed into MLP arachitecture discussed next.

3.1.2 MLP Architecture

MLP models are designed to classify based on SIFT features and raw input data. The model architecture is as follows: the first hidden layer contains 256 neurons, and the second hidden layer has 128 neurons. Finally, the output layer consists of 10 neurons, corresponding to the number of classes in the dataset.Dropout layers are used to avoid overfitting.

3.2 Task 2 : Train MLP using Raw Input Data

Each 28×28 grayscale image is flattened into a 784-dimensional vector and then the pixel values are normalized by dividing by 255. MLP architecture as discussed in 3.1.2 is used except that first layer flattens image that is fed to Dense layer.

3.3 Hyperparameter Combinations

To observe the effects of hyperparamters, four different hyperparameter combinations were tested:

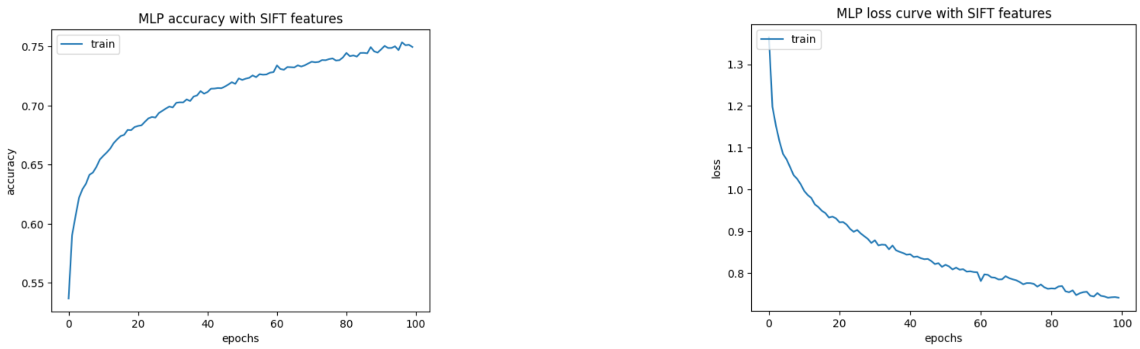
Combination	Learning Rate	Batch Size	Optimizer	Activation	Loss Function
1	0.001	32	ADAM	ReLU	RMSE
2	0.1	4	SGD	Sigmoid	MAE
3	0.01	16	RMSProp	Tanh	MSE
4	0.001	8	SGD	Sigmoid	Huber
5	0.001	16	Adam	ReLU	Categorical Cross Entropy
6	0.01	8	SGD	ReLU	MSE
7	0.001	16	Adam	Tanh	Huber

Table 1: Hyperparameter Combinations Used for Experiments

The models are trained for 100 epochs. Test set accuracy , precision, recall and F1 score are used as evaluation matrices along with confusion matrix to visualize classification results.

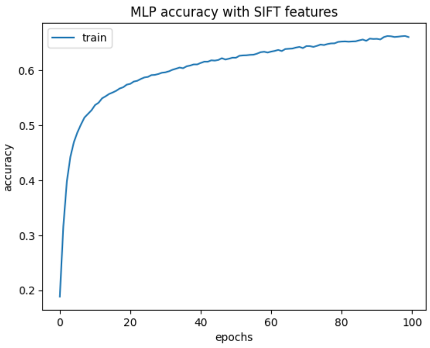
4 Results

The models using 2 different input representation and 7 different hyperparameter combinations(table 1) were evaluated using training accuracy, test accuracy, and loss values to determine their effectiveness.

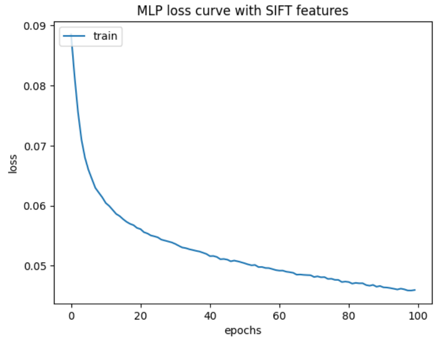


(a) Accuracy plot (b) Loss Plot

Figure 2: MLP accuracy and loss plot using SIFT features (Learning rate = 0.01, Batch size = 16, Optimizer = Adam, Activation = ReLU, Loss = Categorical Cross Entropy)

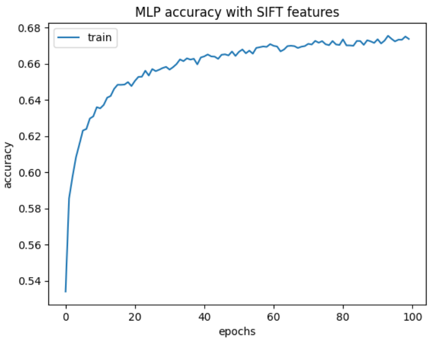


(a) Accuracy plot

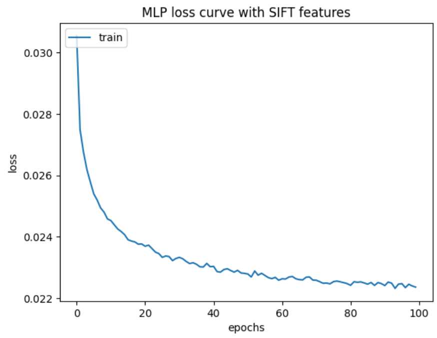


(b) Loss Plot

Figure 3: MLP accuracy and loss plot using SIFT features (Learning rate = 0.01, Batch size = 8, Optimizer = SGD, Activation = ReLU, Loss =MSE)

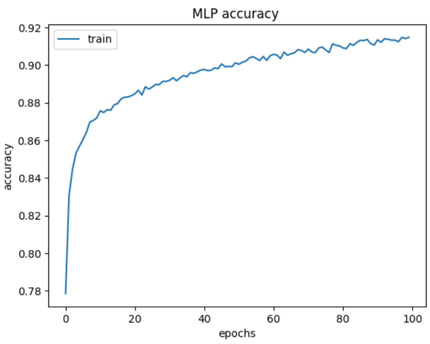


(a) Accuracy plot

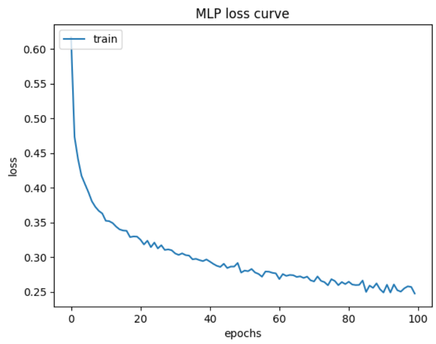


(b) Loss Plot

Figure 4: MLP accuracy and loss plot using SIFT features (Learning rate = 0.001, Batch size = 16, Optimizer = Adam, Activation = Tanh, Loss = Huber)

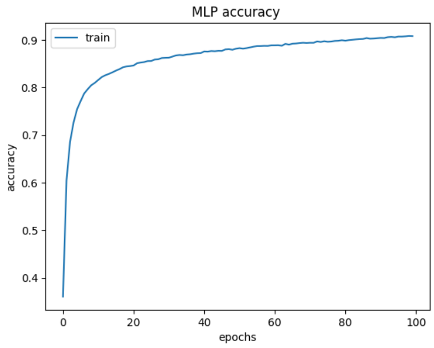


(a) Accuracy plot

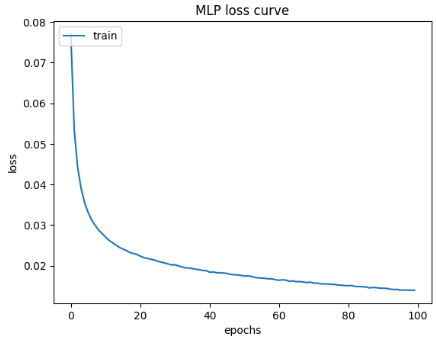


(b) Loss Plot

Figure 5: MLP accuracy and loss plot using raw data (Learning rate = 0.01, Batch size = 16, Optimizer = Adam, Activation = ReLU, Loss = Categorical Cross Entropy)

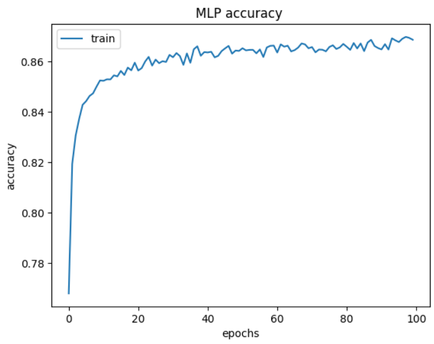


(a) Accuracy plot

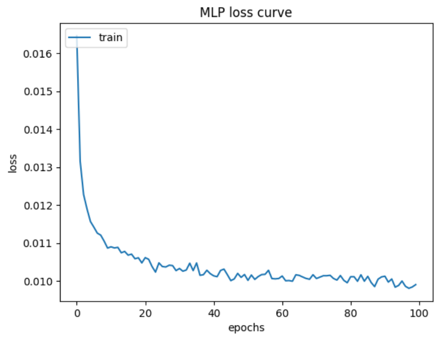


(b) Loss Plot

Figure 6: MLP accuracy and loss plot using raw data (Learning rate = 0.01, Batch size = 8, Optimizer = SGD, Activation = ReLU, Loss =MSE)



(a) Accuracy plot



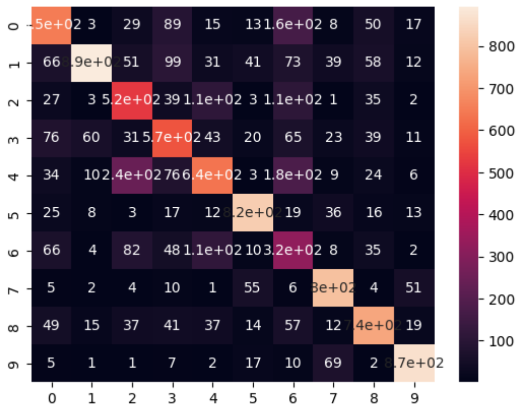
(b) Loss Plot

Figure 7: MLP accuracy and loss plot using raw data (Learning rate = 0.001, Batch size = 16, Optimizer = Adam, Activation = Tanh, Loss = Huber)

classification report:

	precision	recall	f1-score	support
0	0.65	0.63	0.64	1035
1	0.89	0.66	0.76	1364
2	0.52	0.62	0.57	850
3	0.57	0.61	0.59	942
4	0.64	0.52	0.57	1214
5	0.82	0.85	0.84	973
6	0.32	0.47	0.38	690
7	0.80	0.85	0.82	933
8	0.74	0.72	0.73	1018
9	0.87	0.88	0.88	981
accuracy			0.68	10000
macro avg	0.68	0.68	0.68	10000
weighted avg	0.70	0.68	0.69	10000

(a) Classification Report



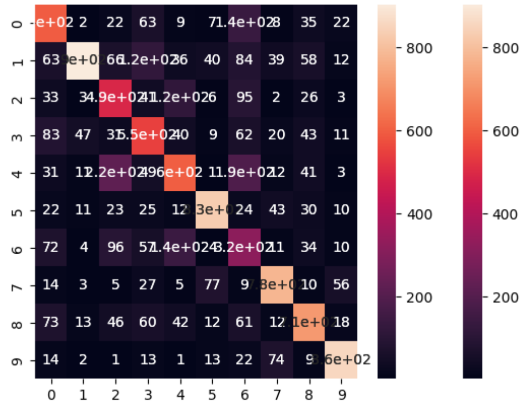
(b) Confusion Matrix

Figure 8: MLP classification report and confusion matrix using SIFT feature (Learning rate = 0.01, Batch size = 16, Optimizer = Adam, Activation = ReLU, Loss = Categorical Cross Entropy)

classification report:

	precision	recall	f1-score	support
0	0.59	0.66	0.63	899
1	0.90	0.64	0.75	1421
2	0.49	0.60	0.54	813
3	0.55	0.61	0.58	892
4	0.60	0.52	0.56	1154
5	0.83	0.81	0.82	1031
6	0.32	0.43	0.37	750
7	0.78	0.79	0.78	985
8	0.71	0.68	0.70	1051
9	0.85	0.85	0.85	1004
accuracy			0.66	10000
macro avg	0.66	0.66	0.66	10000
weighted avg	0.69	0.66	0.67	10000

(a) Classification Report



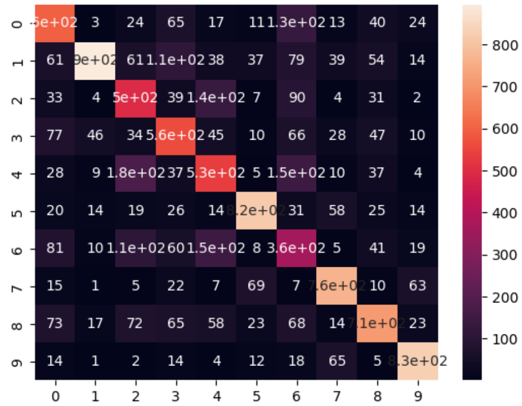
(b) Confusion Matrix

Figure 9: MLP classification report and confusion matrix using SIFT feature using SIFT features (Learning rate = 0.01, Batch size = 8, Optimizer = SGD, Activation = ReLU, Loss =MSE)

classification report:

	precision	recall	f1-score	support
0	0.60	0.65	0.62	924
1	0.90	0.64	0.75	1389
2	0.49	0.59	0.54	842
3	0.56	0.61	0.58	924
4	0.53	0.54	0.54	990
5	0.82	0.79	0.80	1039
6	0.36	0.43	0.40	844
7	0.76	0.79	0.78	963
8	0.71	0.63	0.67	1123
9	0.83	0.86	0.84	962
accuracy			0.66	10000
macro avg	0.66	0.65	0.65	10000
weighted avg	0.67	0.66	0.66	10000

(a) Classification Report



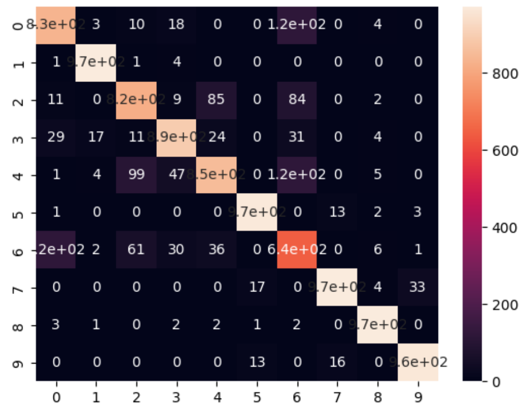
(b) Confusion Matrix

Figure 10: MLP classification report and confusion matrix using SIFT feature using SIFT features (Learning rate = 0.001, Batch size = 16, Optimizer = Adam, Activation = Tanh, Loss = Huber)

classification report:

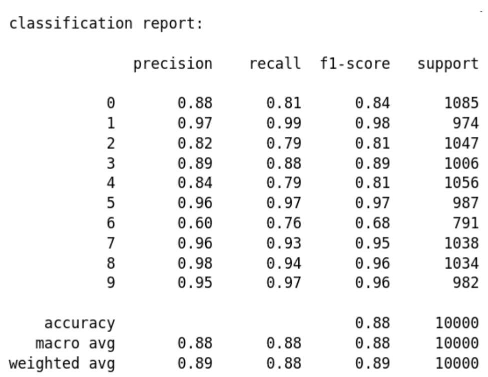
	precision	recall	f1-score	support
0	0.60	0.65	0.62	924
1	0.90	0.64	0.75	1389
2	0.49	0.59	0.54	842
3	0.56	0.61	0.58	924
4	0.53	0.54	0.54	990
5	0.82	0.79	0.80	1039
6	0.36	0.43	0.40	844
7	0.76	0.79	0.78	963
8	0.71	0.63	0.67	1123
9	0.83	0.86	0.84	962
accuracy			0.66	10000
macro avg	0.66	0.65	0.65	10000
weighted avg	0.67	0.66	0.66	10000

(a) Classification Report



(b) Confusion Matrix

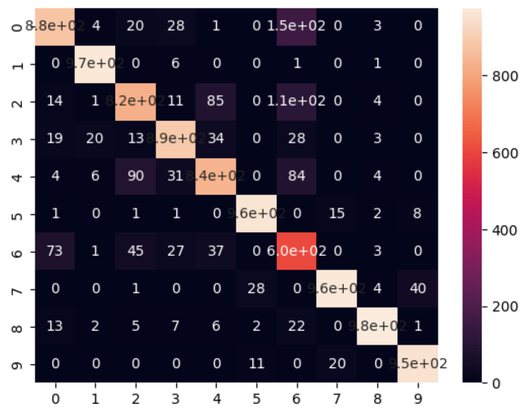
Figure 11: MLP classification report and confusion matrix (Learning rate = 0.01, Batch size = 16, Optimizer = Adam, Activation = ReLU, Loss = Categorical Cross Entropy)



classification report:

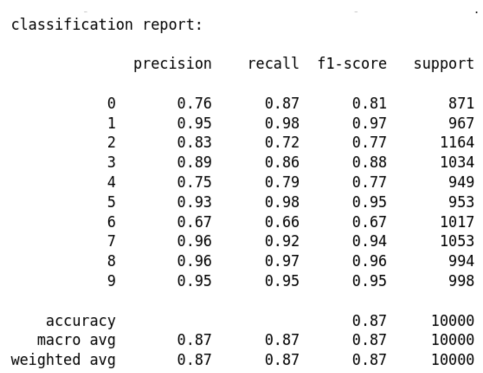
	precision	recall	f1-score	support
0	0.88	0.81	0.84	1085
1	0.97	0.99	0.98	974
2	0.82	0.79	0.81	1047
3	0.89	0.88	0.89	1006
4	0.84	0.79	0.81	1056
5	0.96	0.97	0.97	987
6	0.60	0.76	0.68	791
7	0.96	0.93	0.95	1038
8	0.98	0.94	0.96	1034
9	0.95	0.97	0.96	982
accuracy			0.88	10000
macro avg	0.88	0.88	0.88	10000
weighted avg	0.89	0.88	0.89	10000

(a) Classification Report



(b) Confusion Matrix

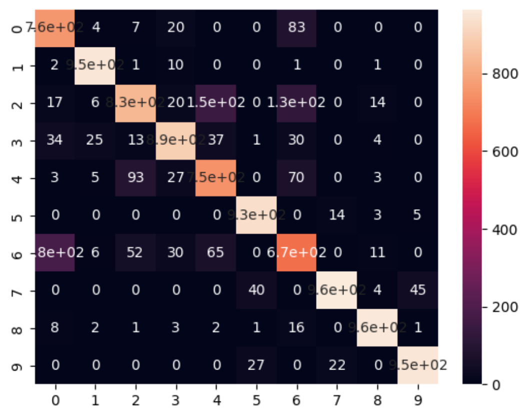
Figure 12: MLP classification report and confusion matrix (Learning rate = 0.01, Batch size = 8, Optimizer = SGD, Activation = ReLU, Loss = MSE)



classification report:

	precision	recall	f1-score	support
0	0.76	0.87	0.81	871
1	0.95	0.98	0.97	967
2	0.83	0.72	0.77	1164
3	0.89	0.86	0.88	1034
4	0.75	0.79	0.77	949
5	0.93	0.98	0.95	953
6	0.67	0.66	0.67	1017
7	0.96	0.92	0.94	1053
8	0.96	0.97	0.96	994
9	0.95	0.95	0.95	998
accuracy			0.87	10000
macro avg	0.87	0.87	0.87	10000
weighted avg	0.87	0.87	0.87	10000

(a) Classification Report



(b) Confusion Matrix

Figure 13: MLP classification report and confusion matrix(Learning rate = 0.001, Batch size = 16, Optimizer = Adam, Activation = Tanh, Loss = Huber)

Model	**Hyperparameter**	**Test Acc.**	**Train Acc.**	**Loss**
SIFT-Based MLP	1	64.45%	70.9%	0.1983
SIFT-Based MLP	2	32.73%	29.64%	0.1429
SIFT-Based MLP	3	51.54%	47.55%	0.0755
SIFT-Based MLP	4	12.18%	9.8%	0.0456
SIFT-Based MLP	5	68.18%	80.91%	0.5814
SIFT-Based MLP	6	66.31%	69.88%	0.0414
SIFT-Based MLP	7	65.67%	71.07%	0.0201
Raw Image MLP	1	87.06%	88.09%	0.1331
Raw Image MLP	2	29.97%	29.24%	0.1420
Raw Image MLP	3	55.34%	53.02%	0.0677
Raw Image MLP	4	14.37%	9.8%	0.046
Raw Image MLP	5	88.88%	93.51%	0.2005
Raw Image MLP	6	88.49%	91.96%	0.0122
Raw Image MLP	7	86.58%	91.96%	0.0086

Table 2: Comparison of Training and Testing Performance Across Hyperparameter Combinations

5 Discussion

The results presented in section 4 indicate that different hyperparameter settings significantly impact model performance.The learning rate determines how model's weights are updated with respect to loss gradient. A higher learning rate such as in combination 1 can cause the model to converge too quickly while a lower learning rate such as in combination 1,5 and 7 requires more epochs to converge but results in stable performance as shown in Figure 2, 4,5 and 7.
The optimizer determines how the models weights are updated based on computed gradies. Adaptive Moment Estimation (Adam) achieves faster convergence and higher test accuracy as shown in Figure 2,4,5 and 7. While Stochastic Gradient Descent (SGD) can provide better generalization over long training durations, it suffers from slow convergence and sensitivity to learning rate selection as shown in Figure 6 and 3. In contrast, Adam's adaptive learning rates allow for more stable training, reducing the risk of vanishing gradients. Overall, Adam is the preferred optimizer due to its efficiency and robustness in handling different feature representations.
Batch size represents the number of samples used in one forward and backward pass through the network and has a direct impact on the accuracy and computational efficiency of the training process. The batch size can be understood as a trade-off between accuracy and speed. Large batch sizes can lead to faster training times but may result in lower accuracy and overfitting, while smaller batch sizes can provide better accuracy, but can be computationally expensive and time-consuming.
An activation function is a mathematical function applied to the output of a neuron. It introduces non-linearity into the model, allowing the network to learn and represent complex patterns in the data. Without this nonlinearity feature, a neural network would behave like a linear regression model, no matter how many layers it has. The activation function decides whether a neuron should be activated by calculating the weighted sum of inputs and adding a bias term. This helps the model make complex decisions and predictions by introducing nonlinearities to the output of each neuron.
Huber Loss combines the advantages of both MSE and MAE. When the error is small, it is similar to MSE, which can smoothly optimize and avoid the gradient problem of MAE. But when the error is large, it is similar to MAE and is insensitive to outliers. Because of that balance, it can handle the SIFT-based models with multiple zeros efficiently compared to the other models.Figure 8,9,10,11,12 , and 13 demonstrate performance of both SIFT and raw input data based models in terms of precision, recall and F1-score.
Following table elaborates comparison between best performing models for both SIFT and MLP.

Model	**Best Test Accuracy (%)**
SIFT	68.18%
Raw Image	88.88%

Table 3: Comparison of SIFT and Raw Image MLP Models

It can be observed from Table 2 and 3 that raw input data MLP performed better than SIFT-based models. One possible reason could be that the small image size, 28x28 pixels, poses a challenge for SIFT feature extraction. When we extract SIFT features, many of them are filled with zeros because keypoints were not extracted. However, inputting a lot of zeros to the MLP interferes with the backpropagation process, slowing down the updating of weights, as the gradient updates depend on the activation values of the input data.

6 Conclusion

It can be concluded that raw input based MLP performed better in terms of accuracy and class performance.While SIFT is robust in terms of rotation, scaling, and affine transformations, it did not outperform simple MLP. In terms of activation function since we are dealing with a classification problem, Adam optimizer with ReLU and Tanh provided better performance. In terms of loss function, RMSE performed similar to MSE, emphasizing large errors, while MAE is not sensitive to outliers and is suitable for tasks with less noise. MSE is more sensitive to large errors (outliers), making it suitable for regression tasks with a uniform error distribution. Huber Loss combines the advantages of MAE and MSE, so it performs better. Since, the focus was classification problem, categotical cross entropy performed better with it's ability to handle probability distributions better.