Assignment #04

Applied Artificial Intelligence

Models Evaluation Report of CNN and ML Models (KNN, SVM)

Name: Abdul Ahad Tariq
Roll Number: 22L-7866

Section: SE-6A

Submitted To: Miss Rushda Muneer

Traditional ML Models

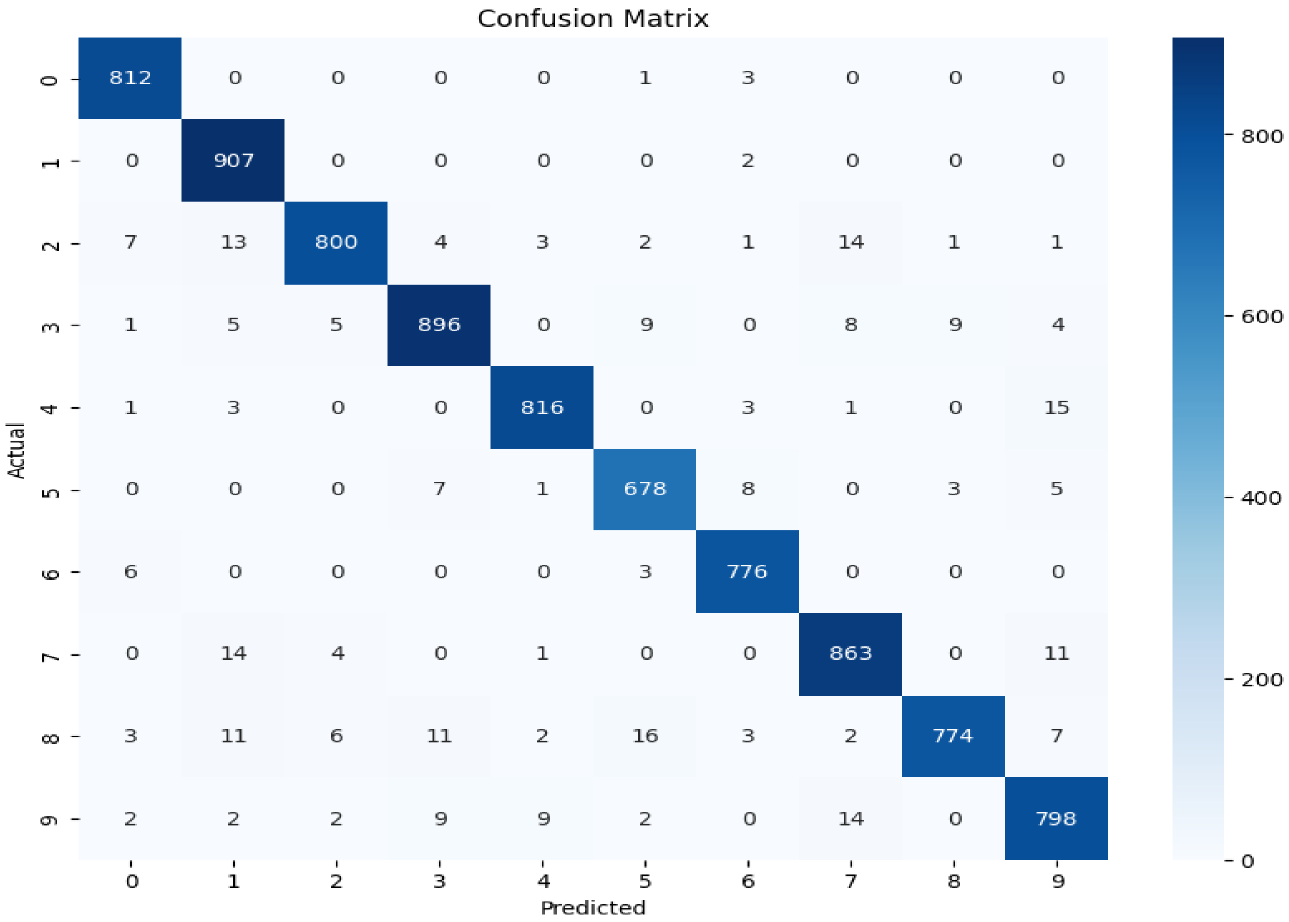
KNN Model

1. Model Architecture

- **Model**: K-Nearest Neighbors
- **Hyperparameter**: n_neighbors = 3

2. Performance Analysis

- **Validation Accuracy**: 0.96
- **F1-score**: 0.97
- **Confusion Matrix**:

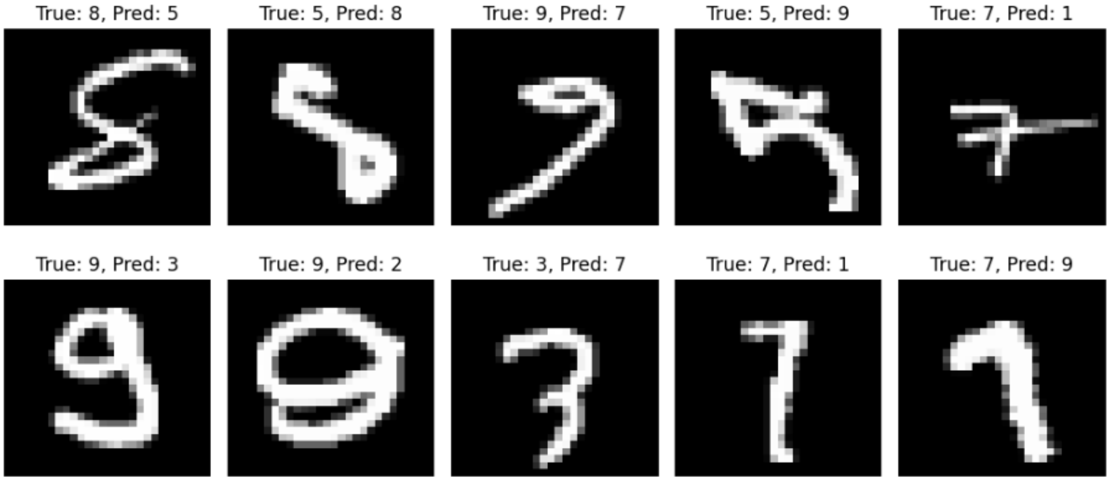


Confusion Matrix

	0	1	2	3	4	5	6	7	8	9
0	812	0	0	0	0	1	3	0	0	0
1	0	907	0	0	0	0	2	0	0	0
2	7	13	800	4	3	2	1	14	1	1
3	1	5	5	896	0	9	0	8	9	4
4	1	3	0	0	816	0	3	1	0	15
5	0	0	0	7	1	678	8	0	3	5
6	6	0	0	0	0	3	776	0	0	0
7	0	14	4	0	1	0	0	863	0	11
8	3	11	6	11	2	16	3	2	774	7
9	2	2	2	9	9	2	0	14	0	798

3. Error Analysis

- Examples of misclassified digits are visualized.
- Common confusion: Digits like 8 vs 5, 7 vs 1, etc.



4. Summary

- **Accuracy**: 0.96
- **Training Speed**: 0.03
- **Resource Usage**: High RAM usage and slow inference on large datasets.

5. Recommendations

- Not ideal for real-time use without optimization.

SVM Model

1. Model Architectures

- **Algorithm Used**: Support Vector Machine (SVM)
- **Kernel**: RBF (Radial Basis Function)
- **C (Penalty Parameter)**: 5.0

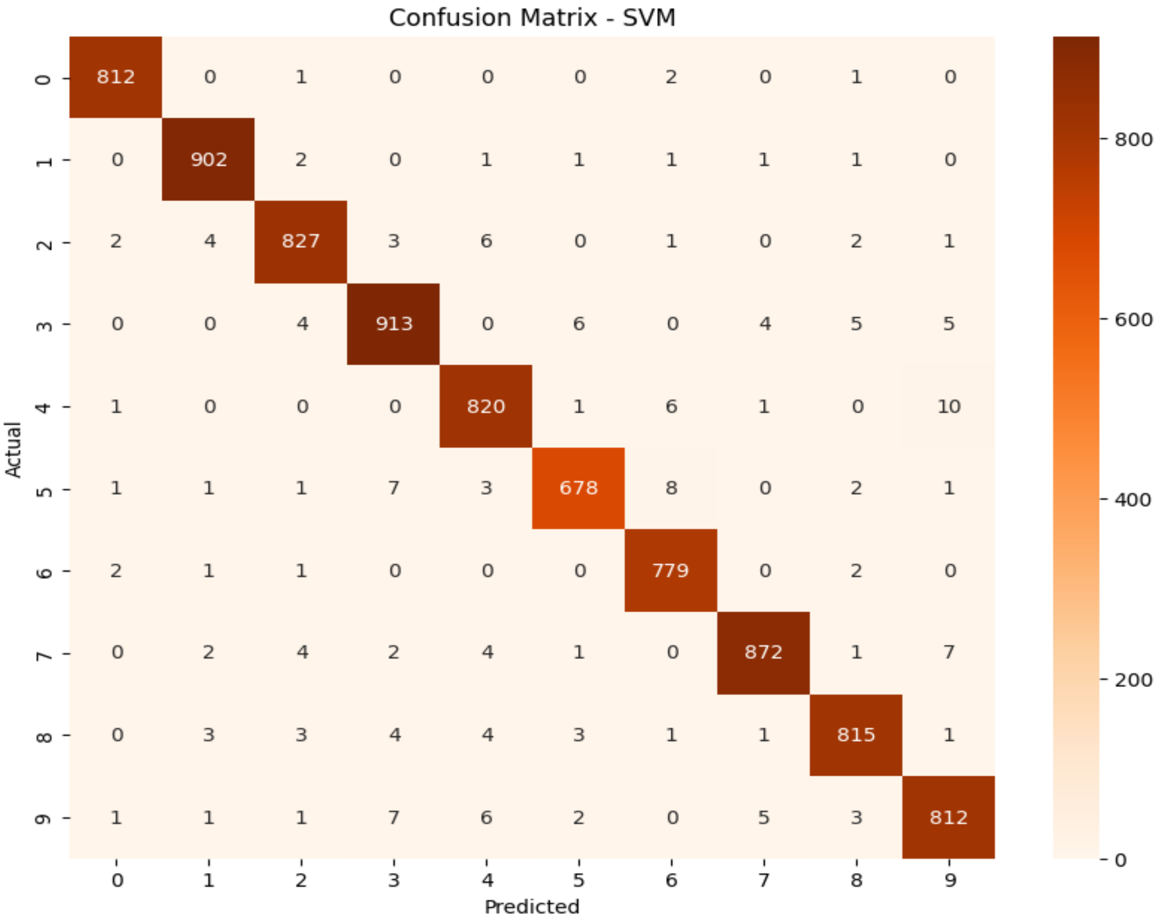
2. Hyperparameters

- C = 5.0 (Controls trade-off between smooth decision boundary and classification accuracy on training data).

- kernel = 'rbf' (RBF kernel is used to handle non-linear digit images).

3. Performance Analysis

- **Validation Accuracy**: 0.97
- **F1-score**: 0.98
- **Confusion Matrix**:

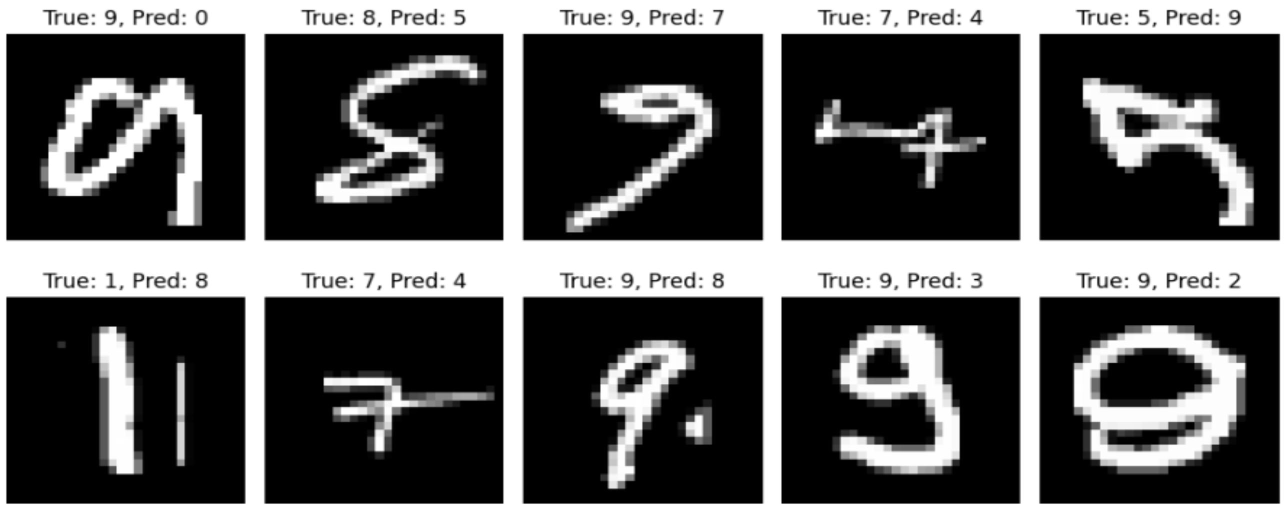

Confusion Matrix - SVM

	0	1	2	3	4	5	6	7	8	9
0	812	0	1	0	0	0	2	0	1	0
1	0	902	2	0	1	1	1	1	1	0
2	2	4	827	3	6	0	1	0	2	1
3	0	0	4	913	0	6	0	4	5	5
4	1	0	0	0	820	1	6	1	0	10
5	1	1	1	7	3	678	8	0	2	1
6	2	1	1	0	0	0	779	0	2	0
7	0	2	4	2	4	1	0	872	1	7
8	0	3	3	4	4	3	1	1	815	1
9	1	1	1	7	6	2	0	5	3	812

4. Error Analysis

- **Misclassified Digits** SVM struggles slightly with visually similar digits.

4. Summary



- **Accuracy**: 0.97
- **Training Speed**: 101.24
- **Resource Usage**: Medium-High

5.Recommendations:

- SVM is useful for small to medium datasets with high accuracy.

CNN Model

1. Model Architecture

Custom **Convolutional Neural Network (CNN)**:

- **Conv2D(32, (3,3)) + ReLU**: feature extraction from 28x28 grayscale images.
- **MaxPooling2D(2x2)**: downsample feature maps.
- **Dropout(0.25)**: prevent overfitting.
- **Conv2D(64, (3,3)) + ReLU** + **MaxPooling2D** + **Dropout**
- **Flatten**: convert 2D feature maps to 1D vector.
- **Dense(128, ReLU)**: fully connected layer.

- **Dropout(0.5)**
- **Dense(10, Softmax)**: output layer for 10 digit classes.

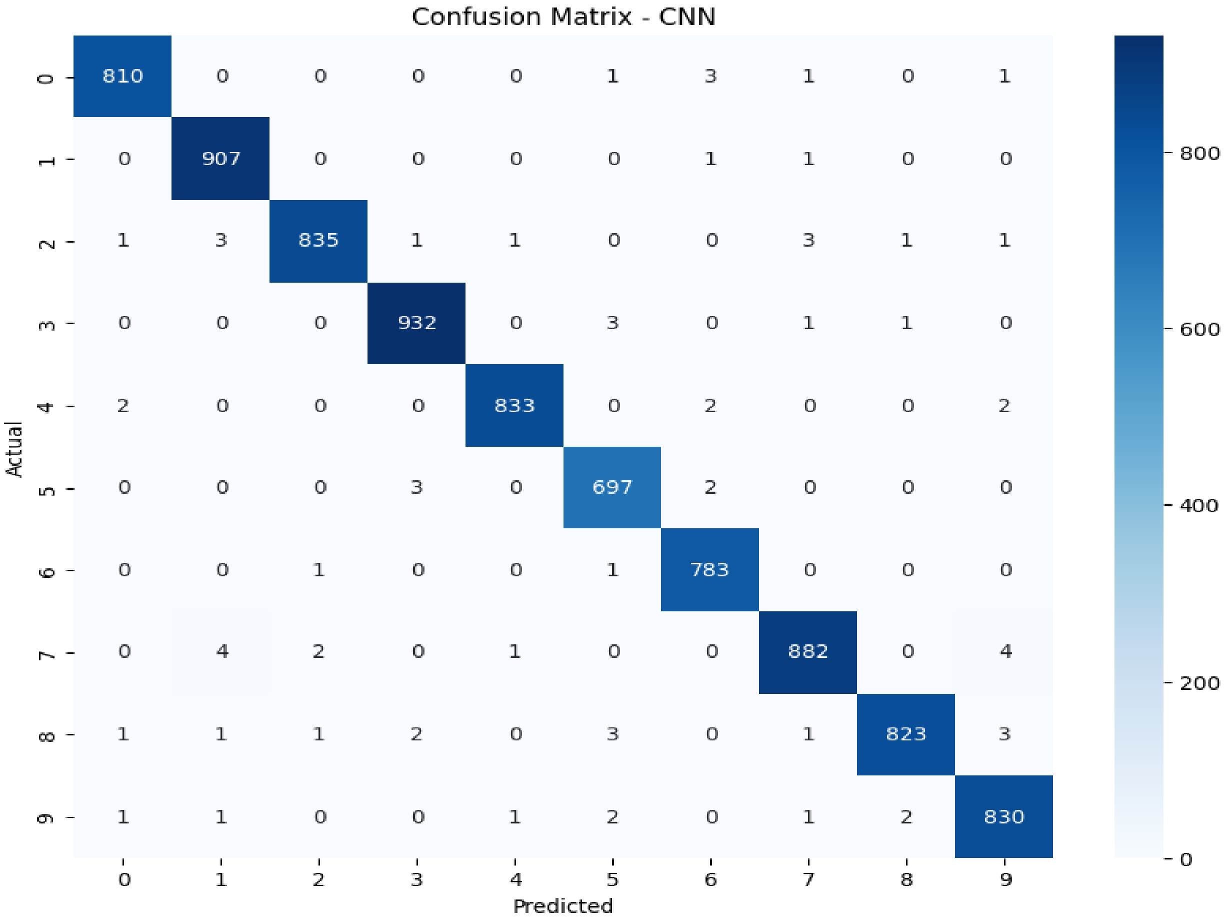
2. Hyperparameters

- **Learning Rate:** 0.001
- **Optimizer:** Adam
- **Epochs :** 15
- **Batch Size:** 64
- **Loss Function:** Categorical Crossentropy

3. Performance Analysis

- **Training Accuracy/Loss** and **Validation Accuracy/Loss** are plotted across 15 epochs.
- **Validation Accuracy**: 0.99

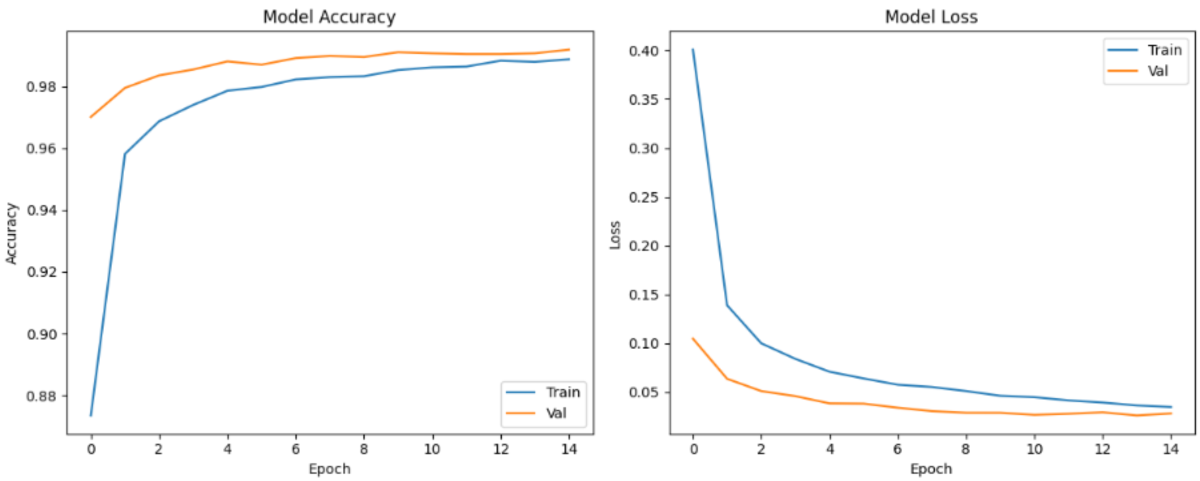
- **Confusion Matrix**:



Confusion Matrix - CNN

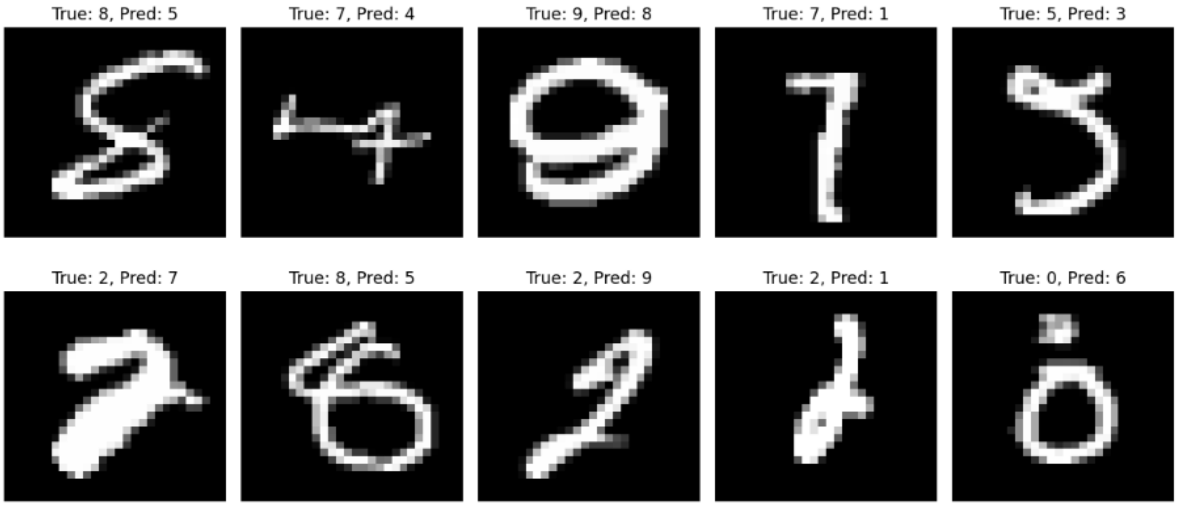
Actual \ Predicted	0	1	2	3	4	5	6	7	8	9
0	810	0	0	0	0	1	3	1	0	1
1	0	907	0	0	0	0	1	1	0	0
2	1	3	835	1	1	0	0	3	1	1
3	0	0	0	932	0	3	0	1	1	0
4	2	0	0	0	833	0	2	0	0	2
5	0	0	0	3	0	697	2	0	0	0
6	0	0	1	0	0	1	783	0	0	0
7	0	4	2	0	1	0	0	882	0	4
8	1	1	1	2	0	3	0	1	823	3
9	1	1	0	0	1	2	0	1	2	830

- **Training Curve:**



4. Error Analysis

- Confusion in similar shapes like 7& 4, 8&5 etc.



5. Summary

- **Accuracy**: 0.99
- **Training Speed**: 221.77
- **Resource Usage**: Too much CPU load

Recommendations for Real-World Deployment:

- CNNs are highly accurate and generalizable.

Comparative Summary:

- **KNN** is easy to implement but not scalable for large datasets like MNIST. Accuracy is decent but suffers from memory and speed issues.
- **SVM** offers strong accuracy but can be slow to train; good for datasets that don’t require GPU.
- **CNN** is **the most accurate**, scalable, and production-ready model, especially when trained with TensorFlow and optimized on GPU.