Machine Learning Research Internship Assignment

Submitted by

Muhammad Burhan Ahmed

Date : 17/05/2024

Introduction

Digit recognition is the process of providing the ability to machines to recognize digits. Among many datasets available, MNIST(Modified National Institute of Standards and Technology) is the most popular dataset which provides a large database of handwritten digits that is commonly used for training various image processing systems. Common applications of digit recognition include postal mail sorting and bank check processing.
Key Features of the dataset are as follows;

- 60,000 **training images**
- 10,000 **testing images**
- Grayscale images of size 28x28 pixels.
- The images are normalized to fit into a 28x28 pixel bounding box and anti-aliased.

Objective

To create a model which is capable of accurately identifying digits.

Literature Review

Fabien Lauer uses an approach to solve this problem of digit recognition by consider the feature extractor as a black box model trained to give relevant features as outputs with no prior knowledge on the data. He introduces a trainable feature extractor (TFE) based on the LeNet5 convolutional network architecture. Amongst all the classifiers that have been applied to character recognition, neural networks were popular due to the performance obtained by LeNet family of neural networks. These are convolutional neural networks that are sensitive to the topological properties of the input. Another approach studied to the recognition of n different digits is to use a single n-class SVM instead of n binary SVM subclassified with the one-against-all method, thus solving a single-constrained optimization problem. Multi-class SVMs have been studied by different authors. But this method is not very popular in digit recognition applications yet and did not yield better performance than other classifiers.

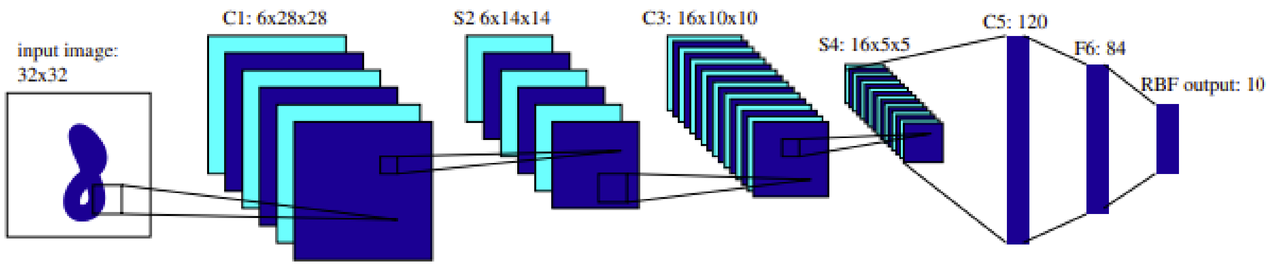


Figure 1: LeNet5 Architecture

Moreover, translations of one pixel in the 8 directions were used to generate new samples and multiply the size of the training set by 9.

New findings of handwritten digit recognition were obtained by Cheng-Lin Liu and Kazuki Nakashima via utilizing cutting-edge feature extraction techniques and classification algorithms. Not every strategy was employed; instead, only those that are generally accepted to yield excellent results were taken into consideration. The databases under test include MNIST, CEDAR, and CENPARMI. Eight classifiers and 10 feature vectors are combined to yield 80 recognition accuracies on the test data set of each database. Every outcome was acquired using normalized pictures without the use of heuristic feature extraction.

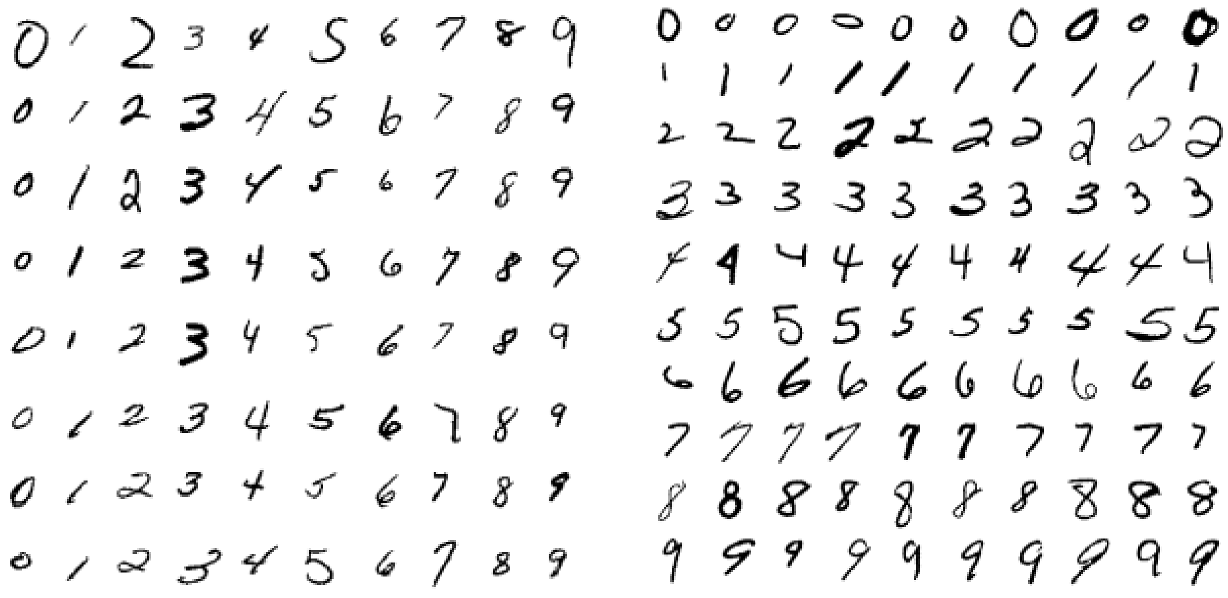


Figure 2: (a) CENPARMI Dataset (b) CEDAR Dataset

The features used in our experiments are the variants of direction feature: chain code, gradient feature with Sobel and Kirsh operators, and peripheral direction contribution. The direction feature was extracted in three steps:

1. Image normalization
2. Direction assignment
3. Feature measuring.

Figure 3: MNIST Dataset

Design decisions

Data Preprocessing

- **One-Hot Encoding**:
  - **Decision**: Convert labels to one-hot encoding.
  - **Reason**: One-hot encoding is necessary for the categorical cross-entropy loss function.

Model Architecture

- **Input Layer**:
  - **Decision**: Use a `Flatten` layer to convert the input images into 1D arrays.
  - **Reason**: The model needs to work with 1D input vectors.
- **Hidden Layer**:
  - **Decision**: Use a `Dense` layer with 128 units and ReLU activation.
  - **Reason**: The hidden layer captures complex patterns in the data, and ReLU activation helps in dealing with non-linearities and prevents vanishing gradient problems.
- **Output Layer**:
  - **Decision**: Use a `Dense` layer with 10 units and softmax activation.
  - **Reason**: The output layer with softmax activation produces probability distributions over the 10 digit classes.

Model Compilation

- **Optimizer:**
  - **Decision:** Use the Adam optimizer.
  - **Reason:** Adam is a popular optimization algorithm that adapts the learning rate and works well in practice for a wide range of problems.
- **Loss Function:**
  - **Decision:** Use categorical cross-entropy.
  - **Reason:** Categorical cross-entropy is suitable for multi-class classification problems where the output is a probability distribution over classes.

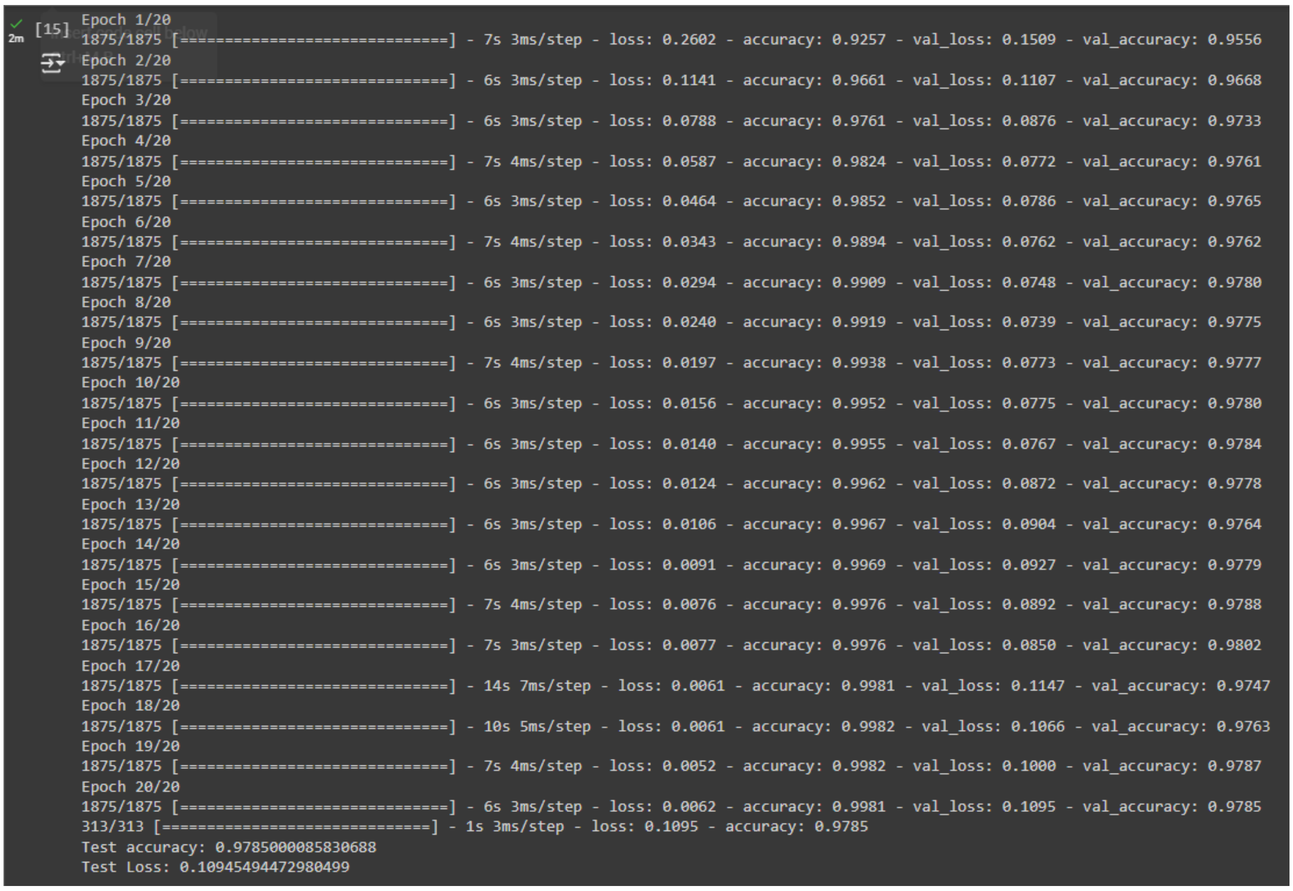
Training Details

- **Decision**: Train the model for 20 epochs and use the test set for validation.
- **Reason**: A sufficient number of epochs ensures that the model has enough opportunities to learn from the training data, and validation helps monitor the model's performance on unseen data.

Evaluation Metrics

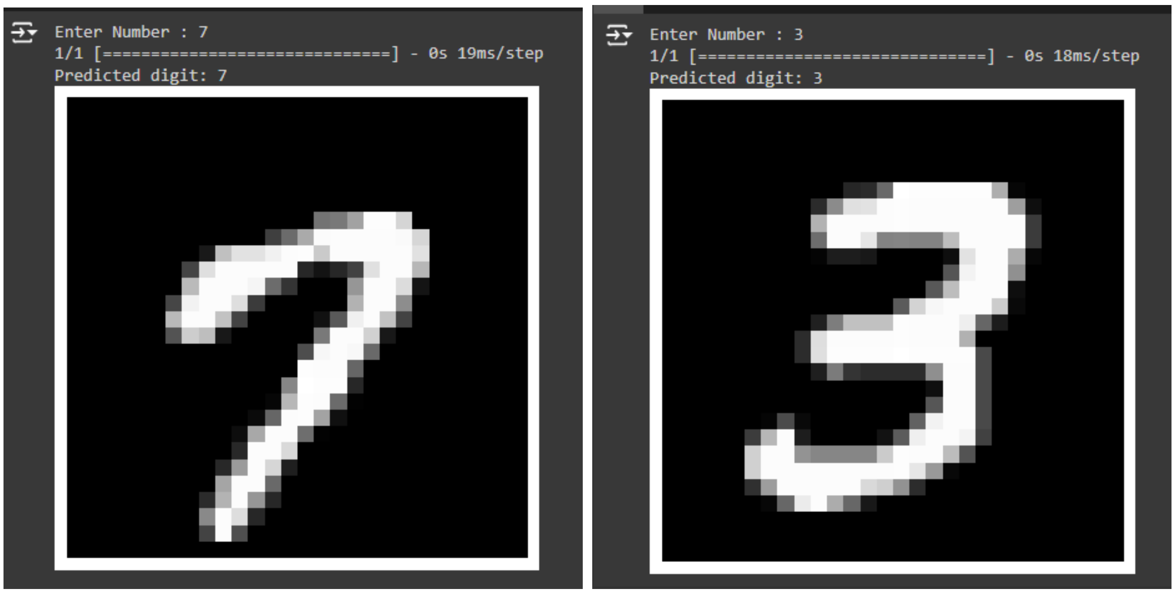
- **Metrics:**
  - **Decision:** Track accuracy.
  - **Reason:** Accuracy is a straightforward and interpretable metric to evaluate the performance of the model.

Results



```
[15] Epoch 1/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.2602 - accuracy: 0.9257 - val_loss: 0.1509 - val_accuracy: 0.9556
Epoch 2/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.1141 - accuracy: 0.9661 - val_loss: 0.1107 - val_accuracy: 0.9668
Epoch 3/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0788 - accuracy: 0.9761 - val_loss: 0.0876 - val_accuracy: 0.9733
Epoch 4/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0587 - accuracy: 0.9824 - val_loss: 0.0772 - val_accuracy: 0.9761
Epoch 5/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0464 - accuracy: 0.9852 - val_loss: 0.0786 - val_accuracy: 0.9765
Epoch 6/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0343 - accuracy: 0.9894 - val_loss: 0.0762 - val_accuracy: 0.9762
Epoch 7/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0294 - accuracy: 0.9909 - val_loss: 0.0748 - val_accuracy: 0.9780
Epoch 8/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0240 - accuracy: 0.9919 - val_loss: 0.0739 - val_accuracy: 0.9775
Epoch 9/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0197 - accuracy: 0.9938 - val_loss: 0.0773 - val_accuracy: 0.9777
Epoch 10/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0156 - accuracy: 0.9952 - val_loss: 0.0775 - val_accuracy: 0.9780
Epoch 11/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0140 - accuracy: 0.9955 - val_loss: 0.0767 - val_accuracy: 0.9784
Epoch 12/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0124 - accuracy: 0.9962 - val_loss: 0.0872 - val_accuracy: 0.9778
Epoch 13/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0106 - accuracy: 0.9967 - val_loss: 0.0904 - val_accuracy: 0.9764
Epoch 14/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0091 - accuracy: 0.9969 - val_loss: 0.0927 - val_accuracy: 0.9779
Epoch 15/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0076 - accuracy: 0.9976 - val_loss: 0.0892 - val_accuracy: 0.9788
Epoch 16/20
1875/1875 [==============================] - 7s 3ms/step - loss: 0.0077 - accuracy: 0.9976 - val_loss: 0.0850 - val_accuracy: 0.9802
Epoch 17/20
1875/1875 [==============================] - 14s 7ms/step - loss: 0.0061 - accuracy: 0.9981 - val_loss: 0.1147 - val_accuracy: 0.9747
Epoch 18/20
1875/1875 [==============================] - 10s 5ms/step - loss: 0.0061 - accuracy: 0.9982 - val_loss: 0.1066 - val_accuracy: 0.9763
Epoch 19/20
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0052 - accuracy: 0.9982 - val_loss: 0.1000 - val_accuracy: 0.9787
Epoch 20/20
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0062 - accuracy: 0.9981 - val_loss: 0.1095 - val_accuracy: 0.9785
313/313 [==============================] - 1s 3ms/step - loss: 0.1095 - accuracy: 0.9785
Test accuracy: 0.9785000085830688
Test Loss: 0.10945494472980499
```

Figure 4: Model Train with epochs = 20



```
Enter Number : 7
1/1 [==============================] - 0s 19ms/step
Predicted digit: 7
```

```
Enter Number : 3
1/1 [==============================] - 0s 18ms/step
Predicted digit: 3
```

Figure 5: Output with different numbers

1 References

Paper Title	Handwritten digit recognition: benchmarking of state-of-the-art techniques
Author	Cheng-Lin Liu, Hiroshi Sako and Hiromichi Fujisawa
Year	2003
Methodologies	Feature Extraction

Paper Title	Case Study in Handwritten Digit Recognition
Author	Corinna Cortes, Harris Drucker, Isabelle Guyon, Patrice Simard and Vladimir Vapnik
Year	2011
Methodologies	LeNet 1 and LeNet 4

Paper Title	A trainable feature extractor for handwritten digit recognition
Author	Fabien Lauera,Ching Y. Suen and Gerard Bloch
Year	2006
Methodologies	Affine transformation & LeNet5 convolutional neural network

https://sci-hub.se/https://www.sciencedirect.com/science/article/abs/pii/S0031320303000852 https://sci-
hub.se/https://www.sciencedirect.com/science/article/abs/pii/S0031320303002243 https://sci-
hub.se/https://www.sciencedirect.com/science/article/abs/pii/S0031320306004250 https://www.engati.co
dataset https://datasets.activeloop.ai/docs/ml/datasets/mnist/ https://docs.ultralytics.com/datasets/cla
https://becominghuman.ai/simple-neural-network-on-mnist-handwritten-digit-dataset-61e47702ed25
https://keras.io/examples/vision/mnist$_{c}onvnet/https : //towardsdatascience.com/how-do-$
$we-train-neural-networks-edd985562b$73