HAND WRITTEN DIGIT RECOGINITION WITH RNN

BY:
DIVYA BHARATHI M

NAME:DIVYA BHARATHI M

DEPARTMENT:B.TECH INFORMATION TECHNOLOGY

COLLEGE NAME:MEENAKSHI SUNDARARAJAN ENGINNERING COLLEGE

GMAIL ID:divyabharathim1807@gmail.com

NM ID:8696E418E400F29B47B4CEACC56D7539

ZONE III : Chennai-III

AGENDA

1. PROBLEM STATEMENT
2. RNN AND ITS LAYERS (algorithm)
3. PROJECT OVERVIEW
4. WHO ARE THE END USERS
5. SOLUTION AND ITS VALUE PROPOSITION
6. THE WOW IN MY SOLUTION
7. MODELLING
8. RESULTS

PROBLEM STATEMENT

- To Develop a deep learning model to accurately classify handwritten digits from the MNIST dataset using RNN. The model should be trained to recognize digits ranging from 0 to 9 and achieve high accuracy in classifying unseen handwritten digits.
- In today's world, there's a big need for computers to understand handwritten numbers. Think about when we write a check or fill out a form – wouldn't it be helpful if a computer could easily read what we wrote?
- The goal is to make this system really good at recognizing all sorts of handwriting, so it can be useful in things like sorting mail, processing checks, and filling out forms automatically.

ALGORITHM USED

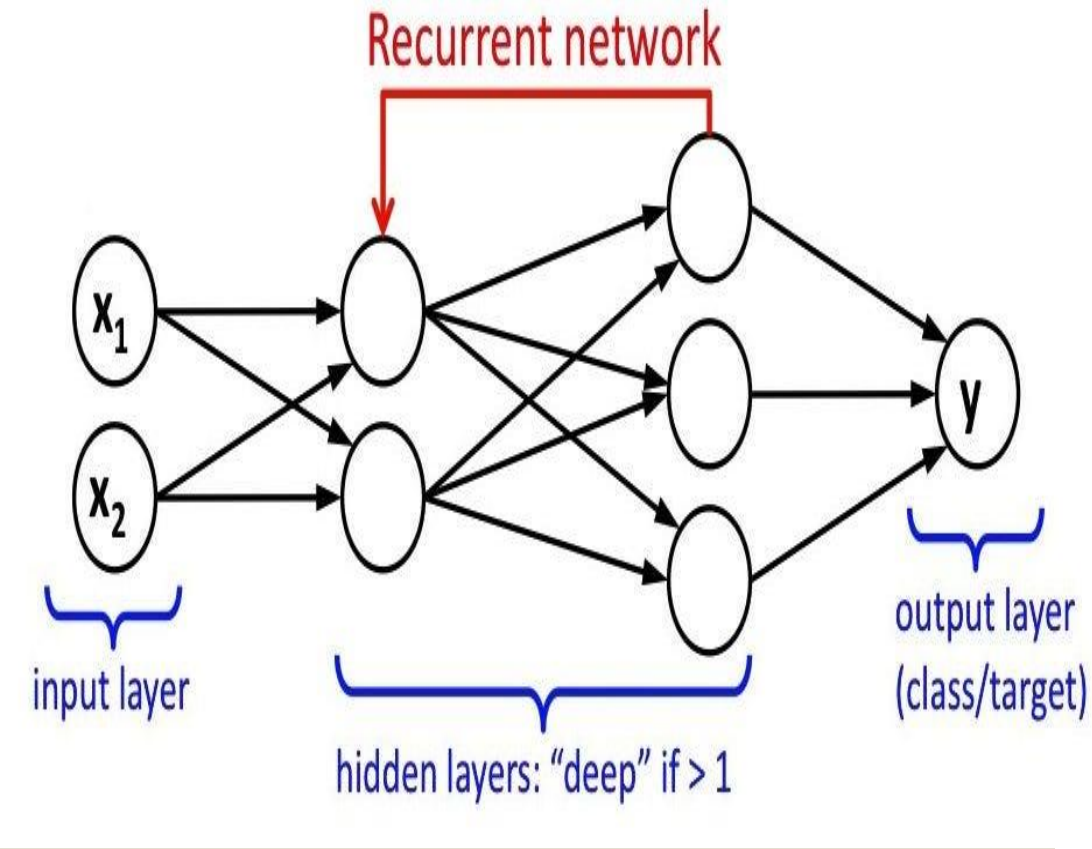
Recurrent Neural Network(RNN)

RNNs are made of neurons: data-processing nodes that work together to perform complex tasks.

The neurons are organized as input, output, and hidden layers. The input layer receives the information to process, and the output layer provides the result.

LAYERS OF RNN

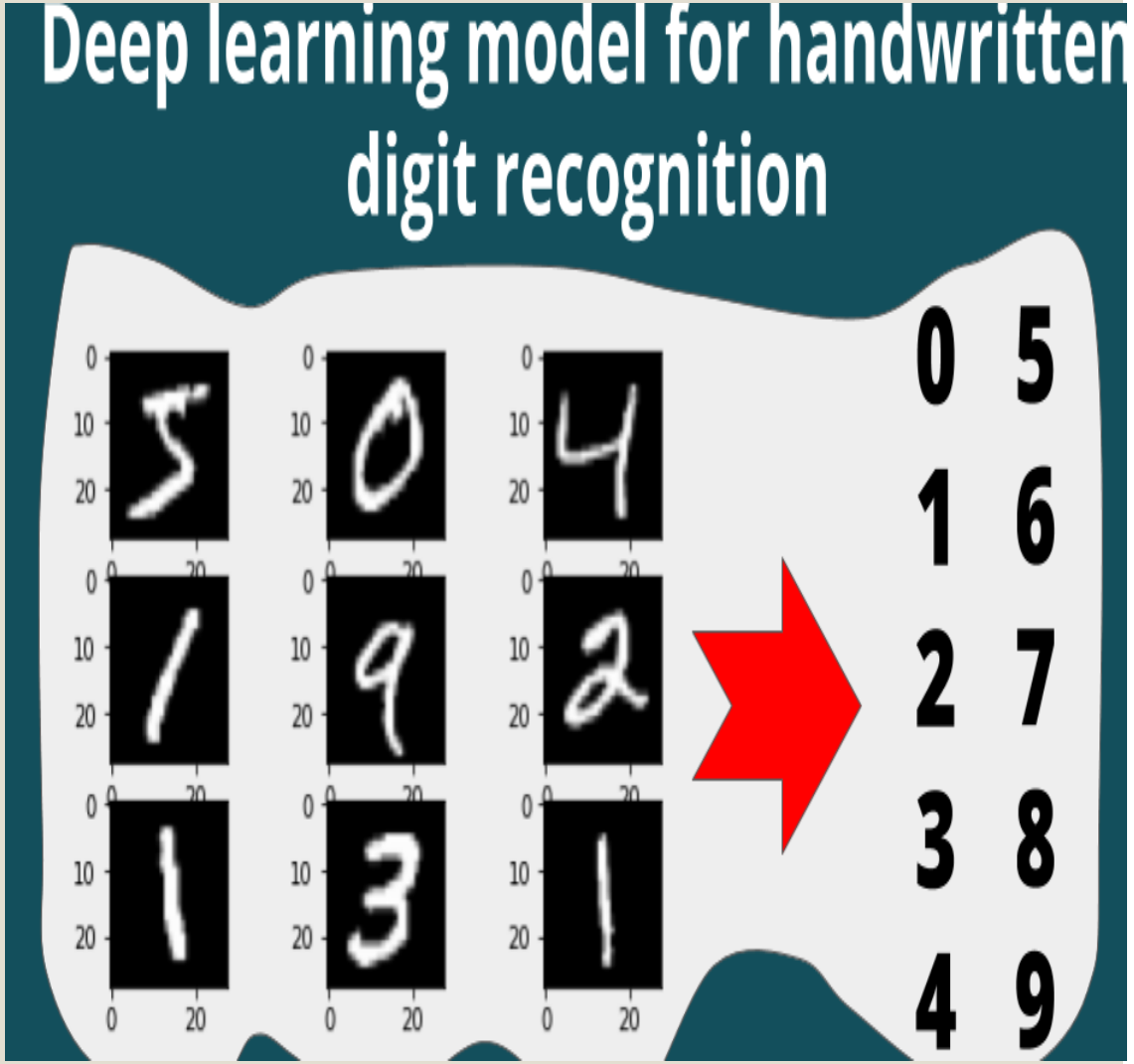
The Recurrent Neural Network consists of multiple fixed activation function units, one for each time step. Each unit has an internal state which is called the hidden state of the unit. This hidden state signifies the past knowledge that the network currently holds at a given time step. This hidden state is updated at every time step to signify the change in the knowledge of the network about the past. The hidden state is updated



PROJECT OVERVIEW

WHO ARE THE END USERS?

Postal services for sorting handwritten addresses, banks for automatic check processing, educational institutions for grading handwritten exams, businesses for digitizing forms and documents



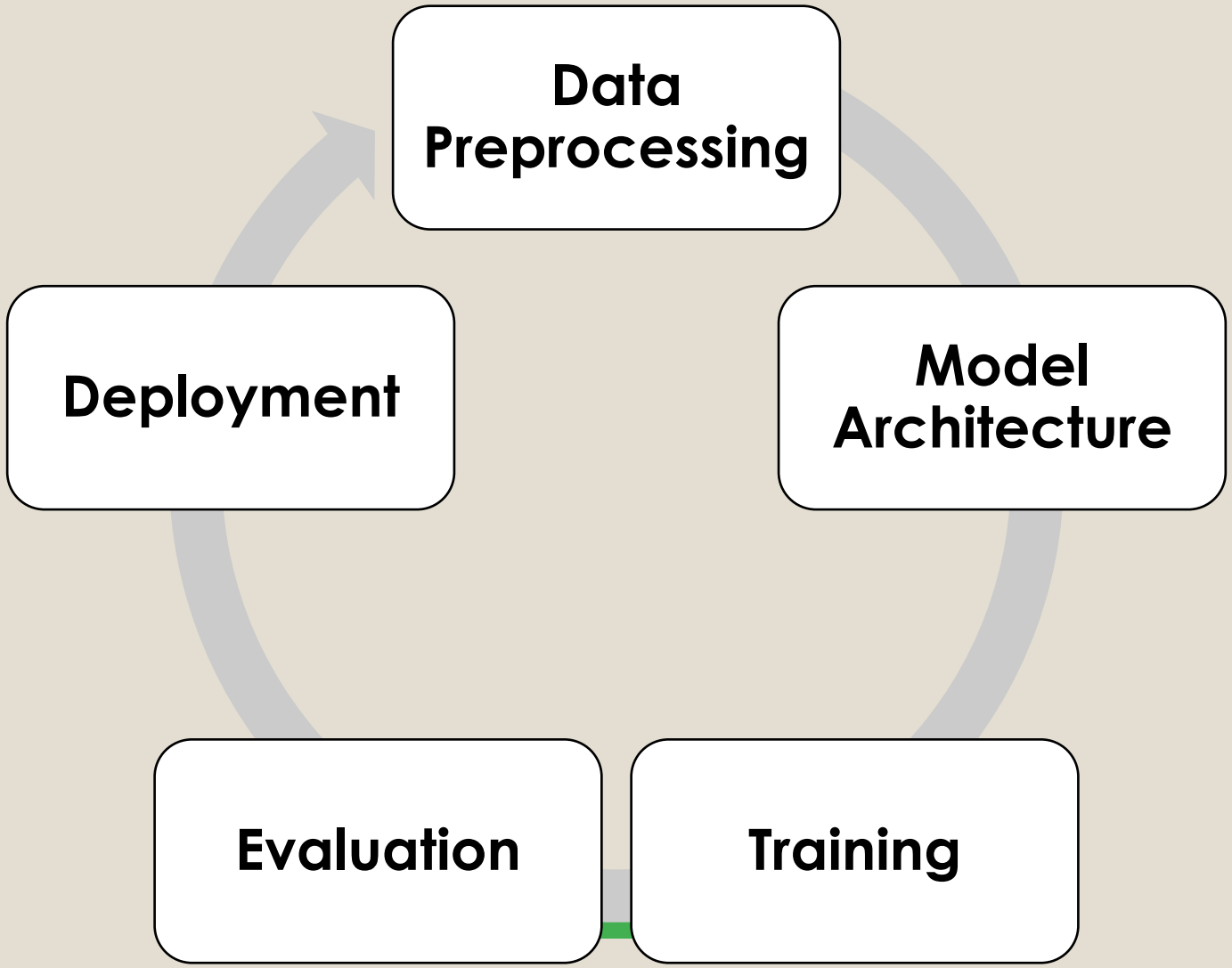
MY SOLUTION AND ITS VALUE PROPOSITION

- Solution: Our solution utilizes RNN-based models to accurately recognize handwritten digits.
- Value Proposition:
  - Improved Accuracy: RNNs can capture sequential dependencies, enhancing recognition accuracy.
  - Efficiency: Faster processing of handwritten digits, leading to enhanced productivity.
  - Versatility: Adaptable to various applications requiring digit recognition.

THE WOW IN MY SOLUTION

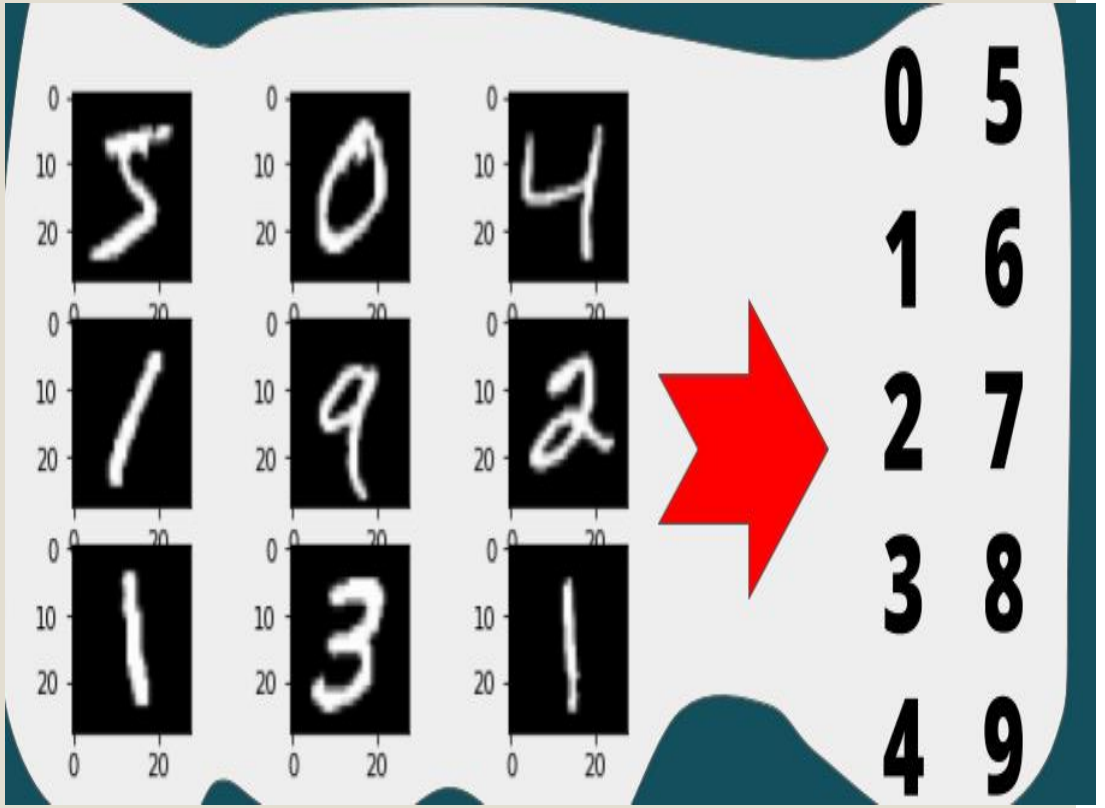
- Real-time Recognition: Instantaneous recognition of handwritten digits, enabling swift processing.
- Adaptive Learning: RNNs can adapt to different handwriting styles, ensuring reliable recognition across diverse datasets.
- Seamless Integration: Easily integrated into existing systems or applications, minimizing implementation efforts.

MODELLING


Data Preprocessing
Model Architecture
Training
Evaluation
Deployment

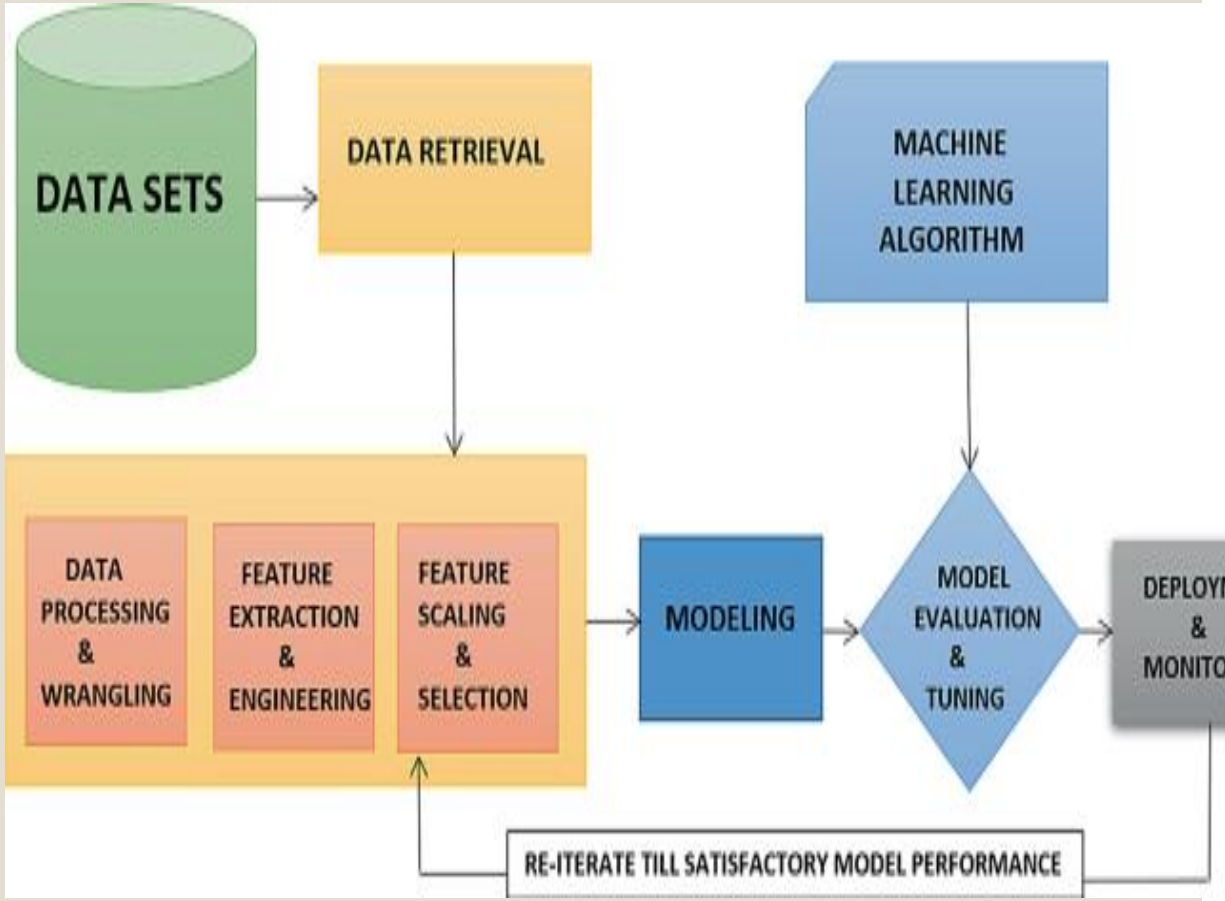
1. **Data Preprocessing**

- Normalize Pixel Values: Scale pixel values to the range [0, 1] by dividing by 255.
- Reshape Images: Convert the 2D image arrays into 3D arrays to fit the RNN input shape.



2.Model Architecture

- Sequential Model: Utilize Keras Sequential API to create a sequential model.
- Recurrent Layer: Add a SimpleRNN layer to capture sequential dependencies in the data.
- Dense Layer: Follow the RNN layer with a Dense layer having softmax activation for digit classification.



3.Training:

Compile the Model: Specify optimizer (e.g., Adam), loss function (e.g., sparse categorical crossentropy), and metrics (e.g., accuracy).

Fit the Model: Train the model on the training data with specified hyperparameters such as batch size, number of epochs, and validation split.

4. **Evaluation**:

Evaluate Model Performance: Assess the model's accuracy and loss on the test dataset.

Fine-tuning: Fine-tune hyperparameters based on evaluation results to optimize model performance.

5. **Deployment**:

Save the Trained Model: Save the trained model for future use or deployment.

Integration: Integrate the model into applications or systems requiring handwritten digit recognition functionality.

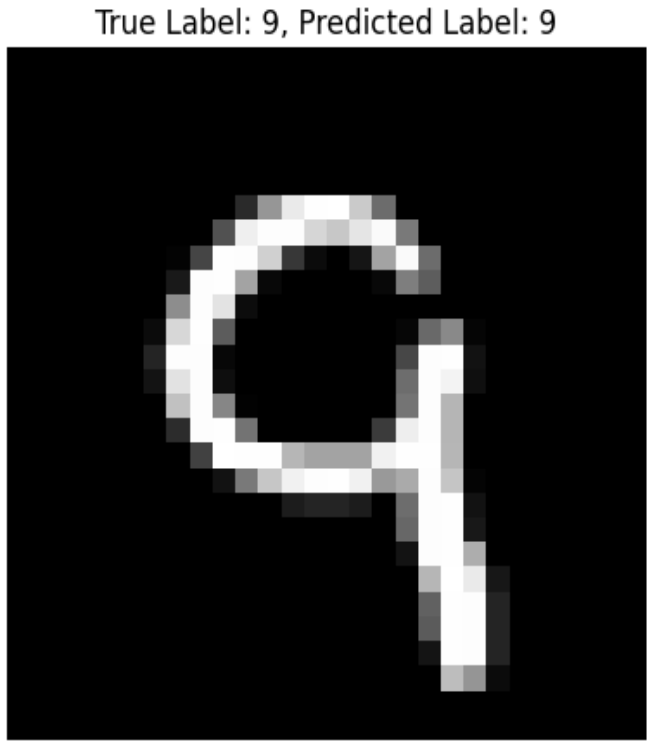
CODE IMPLEMENTATION

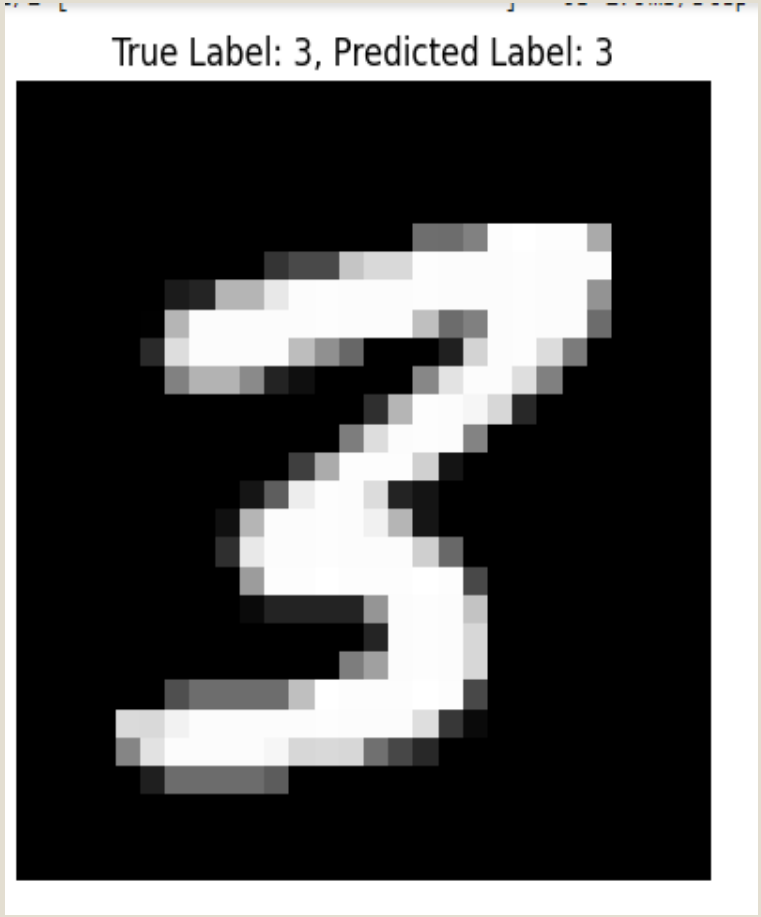
```python
import tensorflow as tf
from tensorflow import keras
import matplotlib.pyplot as plt
import numpy as np

# Load the MNIST dataset
(X_train, y_train), (X_test, y_test) = keras.datasets.mnist.load_data()

# Normalize the pixel values to the range [0, 1]
X_train = X_train / 255.0
X_test = X_test / 255.0

# Define the model architecture
model = keras.models.Sequential([
    keras.layers.SimpleRNN(128, input_shape=(28, 28)),
    keras.layers.Dense(10, activation='softmax')
])

# Compile the model
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

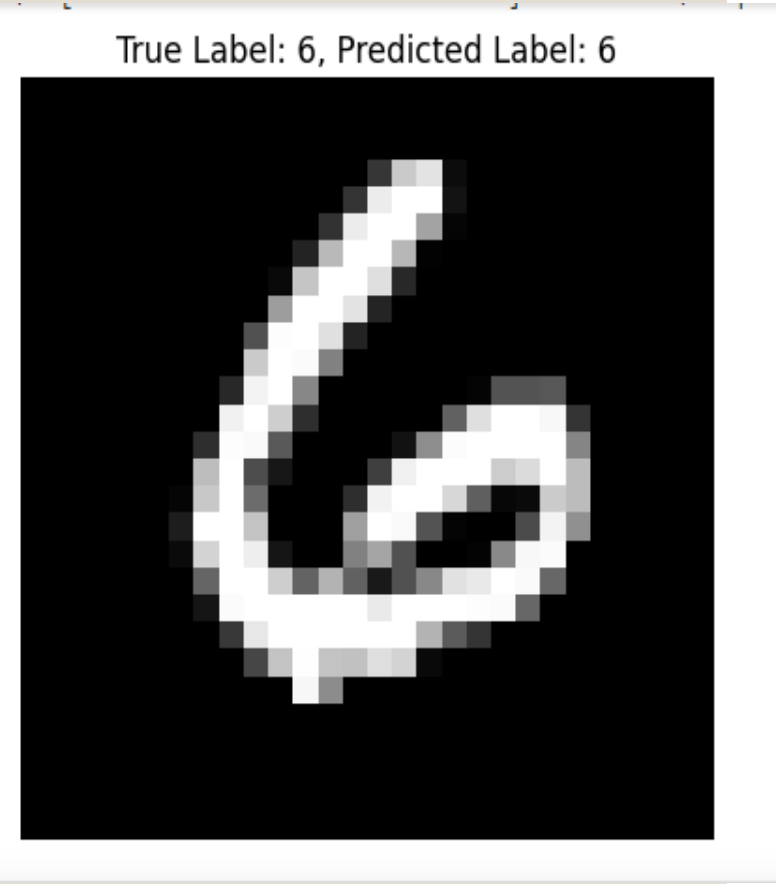
```python
# Reshape the input data to fit the RNN input shape
X_train = X_train.reshape(-1, 28, 28)
X_test = X_test.reshape(-1, 28, 28)

# Train the model
model.fit(X_train, y_train, epochs=5, batch_size=64, validation_split=0.1)

# Evaluate the model on the test set
test_loss, test_acc = model.evaluate(X_test, y_test)
print('Test accuracy:', test_acc)

# Choose a random test image
index = np.random.randint(0, len(X_test))
test_image = X_test[index]
true_label = y_test[index]

# Make a prediction
prediction = np.argmax(model.predict(test_image.reshape(1, 28, 28)))

# Display the image and prediction
plt.imshow(test_image, cmap='gray')
plt.title(f"True Label: {true_label}, Predicted Label: {prediction}")
plt.axis('off')
plt.show()
```

RESULTS

```
313/313 [==============================] - 2s 5ms/step -
Test accuracy: 0.96670001745224
1/1 [==============================] - 0s 187ms/step
```







SUMMARY OF RESULTS

- ✓ Model Training And Model Evaluation : The model is trained for 5 epochs with a batch size of 64 and achieves a validation accuracy of around 99%. : After training, the model is evaluated on the test set. The test accuracy obtained is printed, indicating how well the model generalizes to unseen data.
- ✓ Prediction: A random test image is selected, and the model predicts its label. The true label of the image is also displayed.
- ✓ Result: The test accuracy obtained is typically high, reflecting the effectiveness of the RNN model in recognizing handwritten digits. The prediction on the random test image is also likely to be accurate, demonstrating the model's ability to make correct predictions.
- ✓ CONCLUSION: The RNN model trained on the MNIST dataset achieves high accuracy in digit recognition. This suggests that RNNs are suitable for image classification tasks, especially for datasets like MNIST, which consist of grayscale images of handwritten digits.

THANK YOU

BY:
DIVYA BHARATHI M