```
#IMPORT LIBRARIES
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers, models
import matplotlib.pyplot as plt


#LOAD DATASET
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()

#Normalize Pixel values
x_train, x_test=x_train / 255.0, x_test / 255.0


#Define Model function
def create_model():
    model = models.Sequential([
        layers.Flatten(input_shape=(28, 28)),
        layers.Dense(128, activation='relu'),
        layers.Dropout(0.2),
        layers.Dense(10, activation='softmax')
    ])
    return model
```

Create models

```
adam_model=create_model()
nadam_model=create_model()
sgd_model=create_model()
adagrad_model=create_model()
adamax_model=create_model()


#Compile model function
def compile_model(model,optimizer):
    model.compile(optimizer,
                loss='sparse_categorical_crossentropy',
                metrics=['accuracy'])
```

Compile and train for different optimizers

ADAM

```
compile_model(adam_model,'adam')


adam_history = adam_model.fit(x_train, y_train, epochs=10, validation_data=(x_test, y_test))
```

```
Epoch 1/10
1875/1875 [==============================] - 4s 2ms/step - loss: 0.3022 - accuracy: 0.9123 - val_loss: 0.1458 - val_accuracy: 0.9575
Epoch 2/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1432 - accuracy: 0.9579 - val_loss: 0.0998 - val_accuracy: 0.9682
Epoch 3/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1057 - accuracy: 0.9682 - val_loss: 0.0881 - val_accuracy: 0.9725
Epoch 4/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0863 - accuracy: 0.9740 - val_loss: 0.0833 - val_accuracy: 0.9748
Epoch 5/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0738 - accuracy: 0.9764 - val_loss: 0.0675 - val_accuracy: 0.9793
Epoch 6/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0637 - accuracy: 0.9797 - val_loss: 0.0642 - val_accuracy: 0.9804
Epoch 7/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0588 - accuracy: 0.9815 - val_loss: 0.0673 - val_accuracy: 0.9805
Epoch 8/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0525 - accuracy: 0.9830 - val_loss: 0.0699 - val_accuracy: 0.9783
Epoch 9/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0480 - accuracy: 0.9841 - val_loss: 0.0731 - val_accuracy: 0.9787
Epoch 10/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0433 - accuracy: 0.9854 - val_loss: 0.0730 - val_accuracy: 0.9805
```

NADAM

```
compile_model(nadam_model,'nadam')


nadam_history = nadam_model.fit(x_train, y_train, epochs=10, validation_data=(x_test, y_test))
```

```
Epoch 1/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.2949 - accuracy: 0.9141 - val_loss: 0.1336 - val_accuracy: 0.9621
Epoch 2/10
1875/1875 [==============================] - 3s 2ms/step - loss: 0.1406 - accuracy: 0.9575 - val_loss: 0.1004 - val_accuracy: 0.9691
Epoch 3/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1067 - accuracy: 0.9680 - val_loss: 0.0876 - val_accuracy: 0.9732
Epoch 4/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0862 - accuracy: 0.9729 - val_loss: 0.0816 - val_accuracy: 0.9743
Epoch 5/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0726 - accuracy: 0.9775 - val_loss: 0.0737 - val_accuracy: 0.9786
Epoch 6/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0639 - accuracy: 0.9796 - val_loss: 0.0727 - val_accuracy: 0.9791
Epoch 7/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0569 - accuracy: 0.9816 - val_loss: 0.0717 - val_accuracy: 0.9793
Epoch 8/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0523 - accuracy: 0.9825 - val_loss: 0.0702 - val_accuracy: 0.9796
Epoch 9/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0489 - accuracy: 0.9837 - val_loss: 0.0746 - val_accuracy: 0.9776
Epoch 10/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0435 - accuracy: 0.9853 - val_loss: 0.0718 - val_accuracy: 0.9793
```

SGD

```
compile_model(sgd_model,'SGD')
```

```
sgd_history = sgd_model.fit(x_train, y_train, epochs=10, validation_data=(x_test, y_test))
```

```
Epoch 1/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.7104 - accuracy: 0.8053 - val_loss: 0.3614 - val_accuracy: 0.9048
Epoch 2/10
1875/1875 [==============================] - 2s 1ms/step - loss: 0.3786 - accuracy: 0.8923 - val_loss: 0.2862 - val_accuracy: 0.9206
Epoch 3/10
1875/1875 [==============================] - 3s 2ms/step - loss: 0.3147 - accuracy: 0.9114 - val_loss: 0.2512 - val_accuracy: 0.9282
Epoch 4/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.2786 - accuracy: 0.9198 - val_loss: 0.2238 - val_accuracy: 0.9374
Epoch 5/10
1875/1875 [==============================] - 2s 1ms/step - loss: 0.2530 - accuracy: 0.9284 - val_loss: 0.2045 - val_accuracy: 0.9417
Epoch 6/10
1875/1875 [==============================] - 2s 1ms/step - loss: 0.2335 - accuracy: 0.9336 - val_loss: 0.1883 - val_accuracy: 0.9476
Epoch 7/10
1875/1875 [==============================] - 2s 1ms/step - loss: 0.2161 - accuracy: 0.9383 - val_loss: 0.1774 - val_accuracy: 0.9501
Epoch 8/10
1875/1875 [==============================] - 2s 1ms/step - loss: 0.2042 - accuracy: 0.9421 - val_loss: 0.1674 - val_accuracy: 0.9523
Epoch 9/10
1875/1875 [==============================] - 2s 1ms/step - loss: 0.1929 - accuracy: 0.9456 - val_loss: 0.1571 - val_accuracy: 0.9546
Epoch 10/10
1875/1875 [==============================] - 2s 1ms/step - loss: 0.1836 - accuracy: 0.9479 - val_loss: 0.1507 - val_accuracy: 0.9565
```

ADAMAX

```
compile_model(adamax_model,'adamax')
```

```
adamax_history = adamax_model.fit(x_train, y_train, epochs=10, validation_data=(x_test, y_test))
```

```
Epoch 1/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.4268 - accuracy: 0.8808 - val_loss: 0.2300 - val_accuracy: 0.9346
Epoch 2/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.2400 - accuracy: 0.9310 - val_loss: 0.1774 - val_accuracy: 0.9506
Epoch 3/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1915 - accuracy: 0.9452 - val_loss: 0.1456 - val_accuracy: 0.9575
Epoch 4/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1640 - accuracy: 0.9531 - val_loss: 0.1278 - val_accuracy: 0.9625
Epoch 5/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1446 - accuracy: 0.9594 - val_loss: 0.1168 - val_accuracy: 0.9666
Epoch 6/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1300 - accuracy: 0.9632 - val_loss: 0.1071 - val_accuracy: 0.9697
Epoch 7/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1206 - accuracy: 0.9653 - val_loss: 0.1001 - val_accuracy: 0.9723
Epoch 8/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1107 - accuracy: 0.9683 - val_loss: 0.0955 - val_accuracy: 0.9718
Epoch 9/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.1027 - accuracy: 0.9701 - val_loss: 0.0908 - val_accuracy: 0.9729
Epoch 10/10
1875/1875 [==============================] - 3s 1ms/step - loss: 0.0979 - accuracy: 0.9714 - val_loss: 0.0853 - val_accuracy: 0.9745
```

```
#Evaluate model function
def evaluate_model(model):
    test_loss, test_accuracy = model.evaluate(x_test, y_test, verbose=2)
    print(f"Test accuracy: {test_accuracy*100:.2f}%")
    return test_accuracy
```

Evaluate each model

ADAM

```
adam_accuracy=evaluate_model(adam_model)
```

```
313/313 - 0s - loss: 0.0730 - accuracy: 0.9805 - 248ms/epoch - 791us/step
Test accuracy: 98.05%
```

NADAM

```
nadam_accuracy=evaluate_model(nadam_model)
```

```
313/313 - 0s - loss: 0.0718 - accuracy: 0.9793 - 253ms/epoch - 809us/step
Test accuracy: 97.93%
```

SGD

```
sgd_accuracy=evaluate_model(sgd_model)
```

```
313/313 - 0s - loss: 0.1507 - accuracy: 0.9565 - 245ms/epoch - 781us/step
Test accuracy: 95.65%
```

ADAMAX

```
adamax_accuracy=evaluate_model(adamax_model)
```

```
313/313 - 0s - loss: 0.0853 - accuracy: 0.9745 - 251ms/epoch - 803us/step
Test accuracy: 97.45%
```

Visualization

```
#Training Accuracy
plt.figure(figsize=(10, 4))
plt.subplot(1, 1, 1)
plt.plot(adam_history.history['accuracy'], linestyle='solid', label='Adam Training Accuracy')
plt.plot(nadam_history.history['accuracy'], linestyle='dashed', label='Nadam Training Accuracy')
plt.plot(sgd_history.history['accuracy'], linestyle='dotted', label='SGD Training Accuracy')
plt.plot(adamax_history.history['accuracy'], linestyle='dashdot', label='Adamax Training Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.title('Training Accuracy of Different optimizers')
plt.legend()

#Validation Accuracy
plt.figure(figsize=(10, 4))
plt.subplot(1, , 1)
plt.plot(adam_history.history['val_accuracy'], linestyle='solid', label='Adam Validation Accuracy')
plt.plot(nadam_history.history['val_accuracy'], linestyle='dashed', label='Nadam Validation Accuracy')
plt.plot(sgd_history.history['val_accuracy'], linestyle='dotted', label='SGD Validation Accuracy')
plt.plot(adamax_history.history['val_accuracy'], linestyle='dashdot', label='Adamax Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.title('Validation Accuracy of Different optimizers')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x1210427b150>
```