LAB4

Name: Kushal Sourav B

Regno: 2347125

In [8]:
```
import numpy as np
import tensorflow as tf
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical
import tensorflow as tf
from sklearn.cluster import KMeans
import numpy as np


(train_images, train_labels), (test_images, test_labels) = tf.keras.datasets.mni

train_images = train_images.astype('float32') / 255.0
test_images = test_images.astype('float32') / 255.0

train_images = train_images.reshape((train_images.shape[0], 28 * 28))
test_images = test_images.reshape((test_images.shape[0], 28 * 28))

train_labels = to_categorical(train_labels, 10)
test_labels = to_categorical(test_labels, 10)

X_train, X_test, y_train, y_test = train_test_split(train_images, train_labels,
```

In [21]:
```
class RBFLayer(tf.keras.layers.Layer):
    def __init__(self, units, gamma, **kwargs):
        super(RBFLayer, self).__init__(**kwargs)
        self.units = units
        self.gamma = tf.constant(gamma, dtype=tf.float32)

    def build(self, input_shape):

        self.centers = self.add_weight(name='centers',
                                       shape=(self.units, input_shape[-1]),
                                       initializer='random_uniform',
                                       trainable=False)

    def call(self, inputs):

        diff = tf.expand_dims(inputs, axis=1) - self.centers
        l2 = tf.reduce_sum(tf.square(diff), axis=-1)
        return tf.exp(-self.gamma * l2)
```

In [22]:
```
def build_rbf_network(input_shape, num_classes, num_rbf_units, gamma):
    inputs = tf.keras.Input(shape=input_shape)


    rbf_layer = RBFLayer(num_rbf_units, gamma, name='rbf_layer')(inputs)

    outputs = tf.keras.layers.Dense(num_classes, activation='softmax')(rbf_layer

    model = tf.keras.Model(inputs=inputs, outputs=outputs)
```

```
    return model

input_shape = (784,)
num_classes = 10
num_rbf_units = 100
gamma = 0.1


model = build_rbf_network(input_shape, num_classes, num_rbf_units, gamma)


model.summary()


kmeans = KMeans(n_clusters=num_rbf_units)
kmeans.fit(X_train)


rbf_layer = model.get_layer('rbf_layer')

rbf_layer.centers.assign(kmeans.cluster_centers_)
```

Model: "functional_11"

Layer (type)	Output Shape	Param #
input_layer_6 (InputLayer)	(None, 784)	0
rbf_layer (RBFLayer)	(None, 100)	78,400
dense_6 (Dense)	(None, 10)	1,010

Total params: 79,410 (310.20 KB)

Trainable params: 1,010 (3.95 KB)

Non-trainable params: 78,400 (306.25 KB)

Out[22]:
```
<tf.Tensor: shape=(100, 784), dtype=float32, numpy=
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]], dtype=float32)>
```

In [19]:
```
model.compile(optimizer=tf.keras.optimizers.SGD(learning_rate=0.01),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

history = model.fit(X_train, y_train, epochs=100, validation_data=(X_test, y_tes
```

```
Epoch 1/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.0680 - loss: 2.3011 - va
l_accuracy: 0.1238 - val_loss: 2.2832
Epoch 2/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.1309 - loss: 2.2778 - va
l_accuracy: 0.1243 - val_loss: 2.2615
Epoch 3/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.1330 - loss: 2.2569 - va
l_accuracy: 0.1524 - val_loss: 2.2408
Epoch 4/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.1850 - loss: 2.2338 - va
l_accuracy: 0.2633 - val_loss: 2.2215
Epoch 5/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 9s 6ms/step - accuracy: 0.3021 - loss: 2.2149 - va
l_accuracy: 0.3144 - val_loss: 2.2033
Epoch 6/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.3281 - loss: 2.2003 - va
l_accuracy: 0.3374 - val_loss: 2.1864
Epoch 7/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.3464 - loss: 2.1797 - va
l_accuracy: 0.3617 - val_loss: 2.1708
Epoch 8/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.3680 - loss: 2.1613 - va
l_accuracy: 0.3782 - val_loss: 2.1563
Epoch 9/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 9s 6ms/step - accuracy: 0.3729 - loss: 2.1505 - va
l_accuracy: 0.3971 - val_loss: 2.1429
Epoch 10/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.3929 - loss: 2.1343 - va
l_accuracy: 0.4027 - val_loss: 2.1305
Epoch 11/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.4036 - loss: 2.1226 - va
l_accuracy: 0.4227 - val_loss: 2.1189
Epoch 12/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.4287 - loss: 2.1142 - va
l_accuracy: 0.4248 - val_loss: 2.1080
Epoch 13/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.4319 - loss: 2.1039 - va
l_accuracy: 0.4232 - val_loss: 2.0979
Epoch 14/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.4217 - loss: 2.0947 - va
l_accuracy: 0.4573 - val_loss: 2.0883
Epoch 15/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.4649 - loss: 2.0835 - va
l_accuracy: 0.4517 - val_loss: 2.0794
Epoch 16/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.4468 - loss: 2.0781 - va
l_accuracy: 0.4733 - val_loss: 2.0709
Epoch 17/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.4720 - loss: 2.0687 - va
l_accuracy: 0.4852 - val_loss: 2.0629
Epoch 18/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.4864 - loss: 2.0583 - va
l_accuracy: 0.5508 - val_loss: 2.0553
Epoch 19/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.5312 - loss: 2.0482 - va
l_accuracy: 0.5102 - val_loss: 2.0479
Epoch 20/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.5091 - loss: 2.0441 - va
l_accuracy: 0.5051 - val_loss: 2.0410
```

```
Epoch 21/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.5085 - loss: 2.0365 - va
l_accuracy: 0.5181 - val_loss: 2.0343
Epoch 22/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.4998 - loss: 2.0286 - va
l_accuracy: 0.5850 - val_loss: 2.0278
Epoch 23/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.5695 - loss: 2.0221 - va
l_accuracy: 0.5436 - val_loss: 2.0217
Epoch 24/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.5368 - loss: 2.0164 - va
l_accuracy: 0.5348 - val_loss: 2.0157
Epoch 25/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.5409 - loss: 2.0102 - va
l_accuracy: 0.5612 - val_loss: 2.0099
Epoch 26/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 9s 6ms/step - accuracy: 0.5333 - loss: 2.0067 - va
l_accuracy: 0.6088 - val_loss: 2.0043
Epoch 27/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 9s 6ms/step - accuracy: 0.5745 - loss: 2.0008 - va
l_accuracy: 0.6196 - val_loss: 1.9989
Epoch 28/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6080 - loss: 1.9920 - va
l_accuracy: 0.5722 - val_loss: 1.9935
Epoch 29/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 9s 6ms/step - accuracy: 0.5793 - loss: 1.9888 - va
l_accuracy: 0.5703 - val_loss: 1.9884
Epoch 30/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.5882 - loss: 1.9782 - va
l_accuracy: 0.5967 - val_loss: 1.9834
Epoch 31/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.5761 - loss: 1.9820 - va
l_accuracy: 0.6023 - val_loss: 1.9784
Epoch 32/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.6170 - loss: 1.9752 - va
l_accuracy: 0.5829 - val_loss: 1.9736
Epoch 33/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.5992 - loss: 1.9675 - va
l_accuracy: 0.5852 - val_loss: 1.9689
Epoch 34/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.5931 - loss: 1.9647 - va
l_accuracy: 0.6324 - val_loss: 1.9643
Epoch 35/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.6269 - loss: 1.9646 - va
l_accuracy: 0.6345 - val_loss: 1.9598
Epoch 36/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.6265 - loss: 1.9524 - va
l_accuracy: 0.6281 - val_loss: 1.9554
Epoch 37/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.6262 - loss: 1.9485 - va
l_accuracy: 0.6413 - val_loss: 1.9511
Epoch 38/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.6182 - loss: 1.9482 - va
l_accuracy: 0.6560 - val_loss: 1.9468
Epoch 39/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6544 - loss: 1.9424 - va
l_accuracy: 0.6635 - val_loss: 1.9426
Epoch 40/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6407 - loss: 1.9354 - va
l_accuracy: 0.6538 - val_loss: 1.9384
```

```
Epoch 41/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 6s 4ms/step - accuracy: 0.6203 - loss: 1.9323 - va
l_accuracy: 0.6633 - val_loss: 1.9344
Epoch 42/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6669 - loss: 1.9345 - va
l_accuracy: 0.6562 - val_loss: 1.9304
Epoch 43/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6471 - loss: 1.9260 - va
l_accuracy: 0.6555 - val_loss: 1.9265
Epoch 44/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6572 - loss: 1.9236 - va
l_accuracy: 0.6749 - val_loss: 1.9226
Epoch 45/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6482 - loss: 1.9255 - va
l_accuracy: 0.6728 - val_loss: 1.9188
Epoch 46/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6758 - loss: 1.9167 - va
l_accuracy: 0.6798 - val_loss: 1.9150
Epoch 47/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6825 - loss: 1.9113 - va
l_accuracy: 0.6594 - val_loss: 1.9113
Epoch 48/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6614 - loss: 1.9105 - va
l_accuracy: 0.6798 - val_loss: 1.9076
Epoch 49/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 9s 6ms/step - accuracy: 0.6723 - loss: 1.9091 - va
l_accuracy: 0.6688 - val_loss: 1.9040
Epoch 50/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.6744 - loss: 1.9003 - va
l_accuracy: 0.6630 - val_loss: 1.9004
Epoch 51/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6606 - loss: 1.8967 - va
l_accuracy: 0.6844 - val_loss: 1.8968
Epoch 52/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6888 - loss: 1.8944 - va
l_accuracy: 0.6834 - val_loss: 1.8933
Epoch 53/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6875 - loss: 1.8887 - va
l_accuracy: 0.6755 - val_loss: 1.8899
Epoch 54/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6871 - loss: 1.8888 - va
l_accuracy: 0.6839 - val_loss: 1.8865
Epoch 55/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.6738 - loss: 1.8816 - va
l_accuracy: 0.6933 - val_loss: 1.8831
Epoch 56/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6924 - loss: 1.8827 - va
l_accuracy: 0.6856 - val_loss: 1.8797
Epoch 57/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.6820 - loss: 1.8768 - va
l_accuracy: 0.6874 - val_loss: 1.8764
Epoch 58/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.6915 - loss: 1.8677 - va
l_accuracy: 0.6898 - val_loss: 1.8731
Epoch 59/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6943 - loss: 1.8740 - va
l_accuracy: 0.6953 - val_loss: 1.8699
Epoch 60/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6955 - loss: 1.8630 - va
l_accuracy: 0.6923 - val_loss: 1.8667
```

```
Epoch 61/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6892 - loss: 1.8646 - va
l_accuracy: 0.7013 - val_loss: 1.8635
Epoch 62/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6977 - loss: 1.8659 - va
l_accuracy: 0.6913 - val_loss: 1.8603
Epoch 63/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6965 - loss: 1.8558 - va
l_accuracy: 0.7001 - val_loss: 1.8572
Epoch 64/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7010 - loss: 1.8501 - va
l_accuracy: 0.6933 - val_loss: 1.8541
Epoch 65/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6948 - loss: 1.8475 - va
l_accuracy: 0.6989 - val_loss: 1.8510
Epoch 66/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7040 - loss: 1.8535 - va
l_accuracy: 0.6970 - val_loss: 1.8480
Epoch 67/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7020 - loss: 1.8430 - va
l_accuracy: 0.6938 - val_loss: 1.8450
Epoch 68/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.7020 - loss: 1.8374 - va
l_accuracy: 0.6879 - val_loss: 1.8420
Epoch 69/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.6893 - loss: 1.8364 - va
l_accuracy: 0.7026 - val_loss: 1.8390
Epoch 70/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7035 - loss: 1.8368 - va
l_accuracy: 0.7007 - val_loss: 1.8360
Epoch 71/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.7008 - loss: 1.8339 - va
l_accuracy: 0.7069 - val_loss: 1.8331
Epoch 72/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7088 - loss: 1.8327 - va
l_accuracy: 0.6982 - val_loss: 1.8302
Epoch 73/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7021 - loss: 1.8277 - va
l_accuracy: 0.7056 - val_loss: 1.8274
Epoch 74/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7087 - loss: 1.8227 - va
l_accuracy: 0.7020 - val_loss: 1.8246
Epoch 75/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7031 - loss: 1.8188 - va
l_accuracy: 0.7078 - val_loss: 1.8217
Epoch 76/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7069 - loss: 1.8206 - va
l_accuracy: 0.7003 - val_loss: 1.8189
Epoch 77/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7082 - loss: 1.8181 - va
l_accuracy: 0.7042 - val_loss: 1.8161
Epoch 78/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7039 - loss: 1.8153 - va
l_accuracy: 0.7032 - val_loss: 1.8134
Epoch 79/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7086 - loss: 1.8138 - va
l_accuracy: 0.7081 - val_loss: 1.8106
Epoch 80/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7177 - loss: 1.8094 - va
l_accuracy: 0.7071 - val_loss: 1.8078
```

```
Epoch 81/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7070 - loss: 1.8024 - va
l_accuracy: 0.7143 - val_loss: 1.8052
Epoch 82/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7118 - loss: 1.8003 - va
l_accuracy: 0.7086 - val_loss: 1.8025
Epoch 83/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7122 - loss: 1.8007 - va
l_accuracy: 0.7053 - val_loss: 1.7998
Epoch 84/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 9s 6ms/step - accuracy: 0.7113 - loss: 1.7970 - va
l_accuracy: 0.7022 - val_loss: 1.7972
Epoch 85/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.7061 - loss: 1.7919 - va
l_accuracy: 0.7143 - val_loss: 1.7945
Epoch 86/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7105 - loss: 1.7902 - va
l_accuracy: 0.7150 - val_loss: 1.7919
Epoch 87/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 9s 6ms/step - accuracy: 0.7186 - loss: 1.7885 - va
l_accuracy: 0.7092 - val_loss: 1.7894
Epoch 88/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 6ms/step - accuracy: 0.7105 - loss: 1.7907 - va
l_accuracy: 0.7113 - val_loss: 1.7868
Epoch 89/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7128 - loss: 1.7843 - va
l_accuracy: 0.7113 - val_loss: 1.7842
Epoch 90/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.7172 - loss: 1.7828 - va
l_accuracy: 0.7085 - val_loss: 1.7817
Epoch 91/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7149 - loss: 1.7814 - va
l_accuracy: 0.7103 - val_loss: 1.7792
Epoch 92/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7072 - loss: 1.7793 - va
l_accuracy: 0.7156 - val_loss: 1.7767
Epoch 93/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7182 - loss: 1.7803 - va
l_accuracy: 0.7125 - val_loss: 1.7742
Epoch 94/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7203 - loss: 1.7710 - va
l_accuracy: 0.7170 - val_loss: 1.7717
Epoch 95/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7216 - loss: 1.7684 - va
l_accuracy: 0.7132 - val_loss: 1.7693
Epoch 96/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7232 - loss: 1.7647 - va
l_accuracy: 0.7129 - val_loss: 1.7668
Epoch 97/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7212 - loss: 1.7649 - va
l_accuracy: 0.7100 - val_loss: 1.7645
Epoch 98/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7128 - loss: 1.7626 - va
l_accuracy: 0.7178 - val_loss: 1.7620
Epoch 99/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 7s 5ms/step - accuracy: 0.7233 - loss: 1.7583 - va
l_accuracy: 0.7114 - val_loss: 1.7596
Epoch 100/100
1500/1500 ━━━━━━━━━━━━━━━━━━━━ 8s 5ms/step - accuracy: 0.7191 - loss: 1.7566 - va
l_accuracy: 0.7216 - val_loss: 1.7572
```

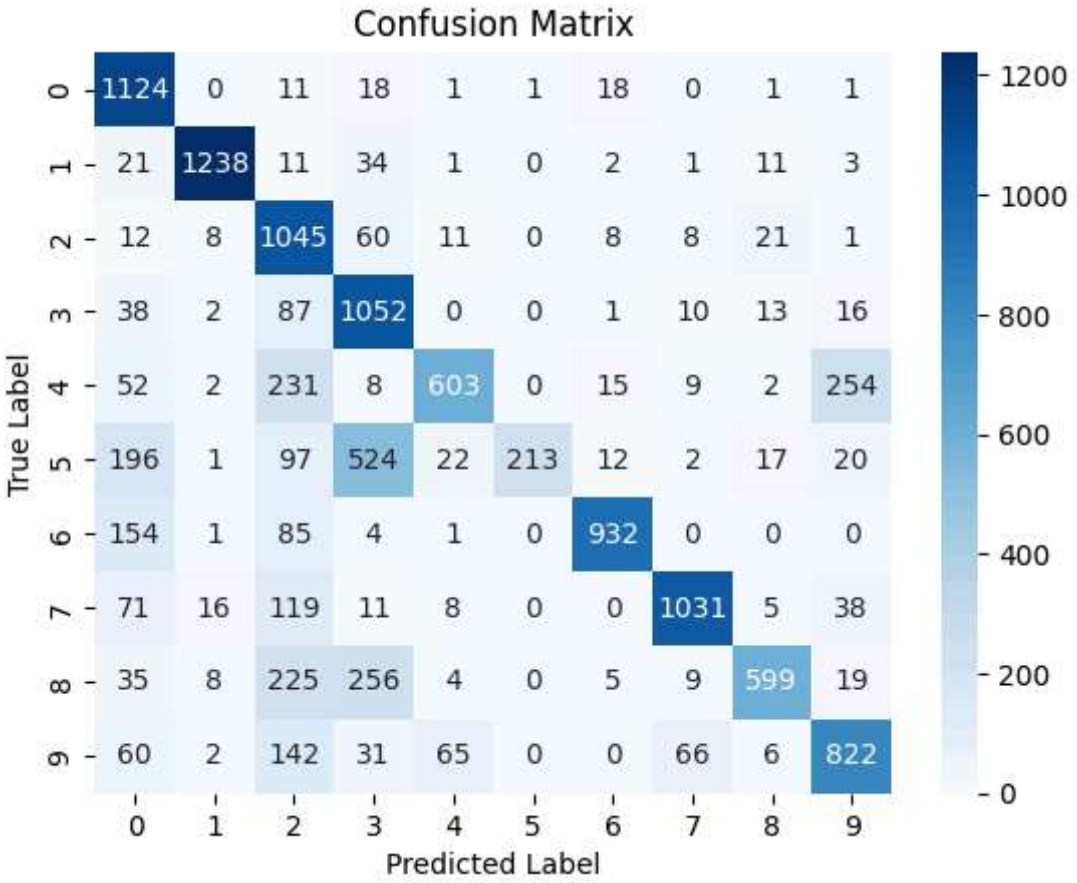
In [20]:
```
from sklearn.metrics import accuracy_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt


test_preds = model.predict(X_test)
test_preds_labels = np.argmax(test_preds, axis=1)
true_labels = np.argmax(y_test, axis=1)

accuracy = accuracy_score(true_labels, test_preds_labels)
print(f"Test Accuracy: {accuracy * 100:.2f}%")

conf_matrix = confusion_matrix(true_labels, test_preds_labels)
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.ylabel('True Label')
plt.xlabel('Predicted Label')
plt.show()
```

```
375/375 ━━━━━━━━━━━━━━━━━━━━ 2s 4ms/step
Test Accuracy: 72.16%
```



Analysis

Strengths:

- RBF networks can model complex, non-linear decision boundaries effectively.
- They are typically faster in training due to the use of K-means for determining the RBF centers.

Limitations:

- RBF networks tend to perform poorly if the number of RBF units is either too high or too low.
- Scaling to large datasets can be computationally expensive due to the clustering step. Choosing an optimal number of RBF units and the gamma parameter requires experimentation.

Effect of RBF Units:

- Increasing the number of RBF units usually improves performance by allowing the network to capture finer details of the data.
- However, too many units can lead to overfitting, where the model performs well on training data but poorly on unseen data.