CIFAR 10 dataset using ANN algorithm

In [1]:
```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
```

loading the dataset:

In [2]:
```
cifar=tf.keras.datasets.cifar10
(training_images,training_labels),(testing_images,testing_labels)=cifar.load_data()
```

Assigning names of labels:

In [3]:
```
names=['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','truck']
```

Checking the shapes of Images & Labels:

In [4]:
```
training_images.shape
```

Out[4]:
```
(50000, 32, 32, 3)
```

In [5]:
```
training_labels.shape
```

Out[5]:
```
(50000, 1)
```

Reshaping training and testing images to a single 4D list

In [6]:
```
training_images=training_images.reshape(50000,32,32,3)
testing_images=testing_images.reshape(10000,32,32,3)
```

Dividing image pixel by 255 so that pixel comes in range 0 to 1:

In [7]:
```
training_images=training_images/255.0
testing_images=testing_images/255.0
```

Build the neural network

In [8]:
```
model=tf.keras.models.Sequential([
  tf.keras.layers.Conv2D(64, (3,3), activation='relu', input_shape=(32, 32, 3)),
  tf.keras.layers.MaxPooling2D(2, 2),
  tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
  tf.keras.layers.MaxPooling2D(2,2),
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dense(10, activation='softmax')
])
```

Compiling the model:

In [9]:
```
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

Training the the model:

In [10]:
```
model.fit(training_images,training_labels,epochs=8)
```

```
Epoch 1/8
1563/1563 [==============================] - 62s 38ms/step - loss: 1.4143 - accuracy: 0.4906
Epoch 2/8
1563/1563 [==============================] - 54s 35ms/step - loss: 1.0503 - accuracy: 0.6318
Epoch 3/8
1563/1563 [==============================] - 51s 33ms/step - loss: 0.9154 - accuracy: 0.6819
Epoch 4/8
1563/1563 [==============================] - 55s 35ms/step - loss: 0.8167 - accuracy: 0.7163
Epoch 5/8
1563/1563 [==============================] - 52s 33ms/step - loss: 0.7330 - accuracy: 0.7431
Epoch 6/8
1563/1563 [==============================] - 57s 36ms/step - loss: 0.6655 - accuracy: 0.7684
Epoch 7/8
1563/1563 [==============================] - 59s 38ms/step - loss: 0.5990 - accuracy: 0.7912
Epoch 8/8
1563/1563 [==============================] - 55s 35ms/step - loss: 0.5389 - accuracy: 0.8111
```

Out[10]:
```
<keras.src.callbacks.History at 0x187b5a4bf40>
```

Evaluating the model:

In [11]:
```
test_loss,test_acc=model.evaluate(testing_images,testing_labels)
```

```
313/313 [==============================] - 3s 10ms/step - loss: 0.9285 - accuracy: 0.7095
```

In [12]:
```
print("Test accuracy:",test_acc)
```

```
Test accuracy: 0.7095000147819519
```

Data prediction:

In [13]:
```
predictions=model.predict(testing_images)
```

```
313/313 [==============================] - 3s 10ms/step
```

Printing data predicted for 3rd index:

In [14]:
```
print(names[np.argmax(predictions[3])])
```

```
airplane
```

Actual data at index 3:

In [15]:
```
print(names[testing_labels[3][0]])
```

```
airplane
```