# LAB Assignment No. 9

## Convolution Neural Network

## Open Ended LAB

**Build a Convolutional Neural Network (CNN) using Python and TensorFlow/Keras to classify images of Cats and Dogs.**

**Instructions:**

1. Collect or download a dataset containing:
   - **500 images of Cats**
   - **500 images of Dogs**
2. Organize the dataset as:

```
dataset/
          cats/
           dogs/
```

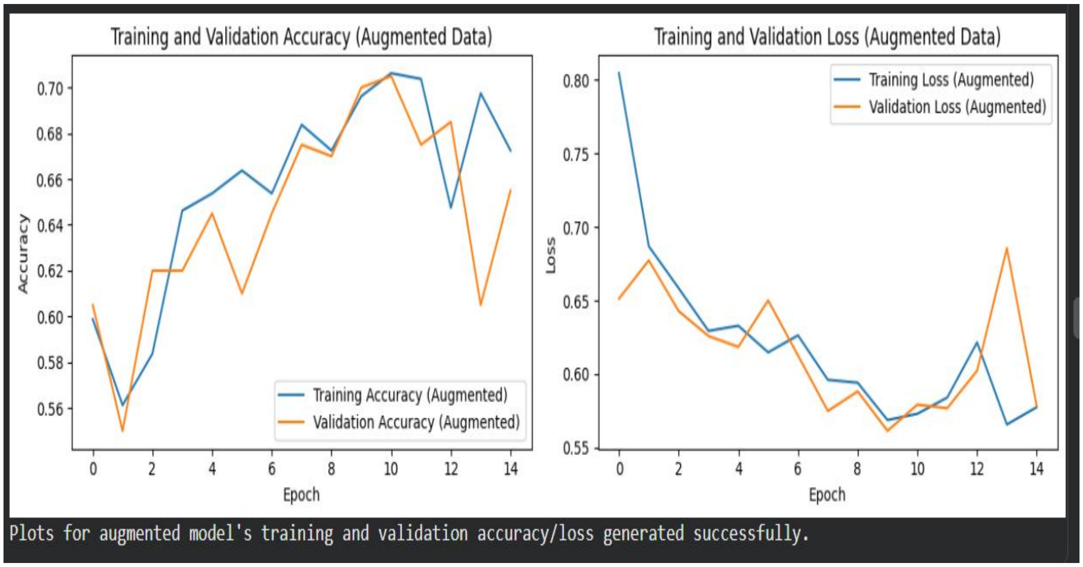
3. Write Python code to:
   - Load and preprocess images (resize to 150×150)
   - Split into training (80%) and testing (20%)
   - Build a CNN model
   - Train the model for 10–20 epochs
   - Plot training & validation accuracy and loss
   - Evaluate model performance using a confusion matrix
   - Predict whether a new input image is Cat or Dog
4. At the end of the program, show the prediction result for a test image:
   - "This image is a CAT"
   - or
     "This image is a DOG"
5. Submit your Python code, dataset, graphs, and output screenshots.

Code:

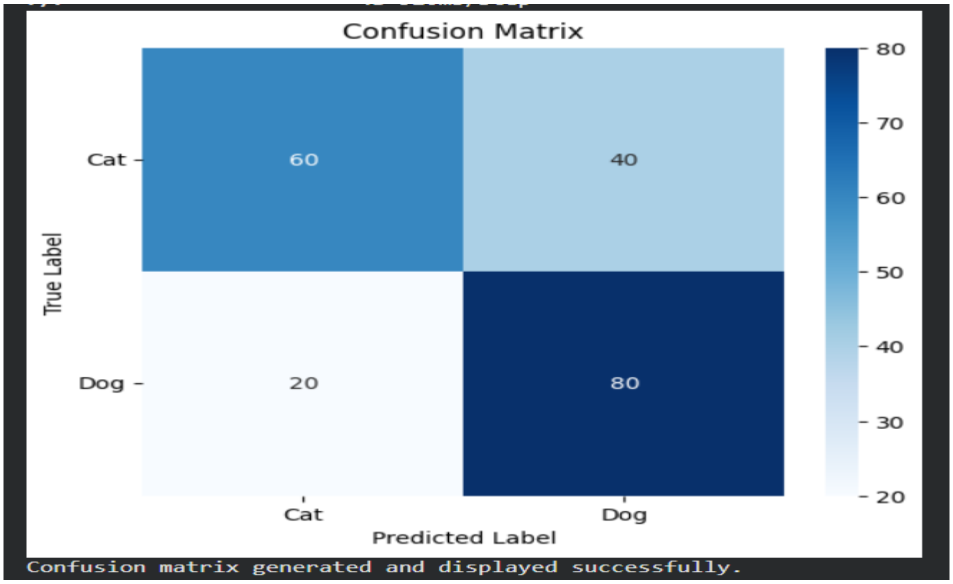
```
import numpy as np
import matplotlib.pyplot as plt
import kagglehub
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.utils import to_categorical
path = kagglehub.dataset_download("karakaggle/kaggle-cat-vs-dog-dataset")
print("Path to dataset files:", path)
DATASET_PATH = None
for root, dirs, files in os.walk(path):
  if "PetImages" in dirs:
    DATASET_PATH = os.path.join(root, "PetImages")  break
if DATASET_PATH is None:
  raise Exception("PetImages folder not found!")
print("Final Dataset Path:", DATASET_PATH)
print("Classes:", os.listdir(DATASET_PATH))
IMG_SIZE = 150
EPOCHS = 15
data = []
labels = []
categories = ["Cat", "Dog"]
for label, category in enumerate(categories):
  folder_path = os.path.join(DATASET_PATH, category)
  for img_name in os.listdir(folder_path):
    img_path = os.path.join(folder_path, img_name)
```

```
 img = cv2.imread(img_path)
      if img is None:    continue
      img = cv2.resize(img, (IMG_SIZE, IMG_SIZE))
      img = img / 255.0
      data.append(img)
      labels.append(label)
data = np.array(data)
labels = to_categorical(labels, num_classes=2)
X_train, X_test, y_train, y_test = train_test_split(
   data, labels, test_size=0.2, random_state=42)
model=Sequential([Conv2D(32, (3,3), activation='relu',input_shape=(IMG_SIZE, IMG_SIZE, 3)),
   MaxPooling2D(2,2),
   Conv2D(64, (3,3), activation='relu'),
   MaxPooling2D(2,2),
   Conv2D(128, (3,3), activation='relu'),
   MaxPooling2D(2,2),
   Flatten(),
   Dense(128, activation='relu'),
   Dropout(0.5),
   Dense(2, activation='softmax')])
model.compile(
   optimizer='adam',
   loss='categorical_crossentropy',
   metrics=['accuracy'])
model.summary()
history = model.fit(
   X_train, y_train,
   validation_data=(X_test, y_test),
plt.figure(figsize=(12,5))

plt.subplot(1,2,1)

plt.plot(history.history['accuracy'], label='Train Accuracy')

plt.plot(history.history['val_accuracy'], label='Validation Accuracy')

plt.legend()

plt.title("Accuracy")

plt.subplot(1,2,2)

plt.plot(history.history['loss'], label='Train Loss')

plt.plot(history.history['val_loss'], label='Validation Loss')

plt.legend()

plt.title("Loss") plt.show()

y_pred = model.predict(X_test)

y_pred_classes = np.argmax(y_pred, axis=1)

y_true = np.argmax(y_test, axis=1)

cm = confusion_matrix(y_true, y_pred_classes)

disp = ConfusionMatrixDisplay(cm, display_labels=["Cat", "Dog"])

disp.plot(cmap=plt.cm.Blues) plt.show()

def predict_image(img_path):

   img = cv2.imread(img_path)

   img = cv2.resize(img, (IMG_SIZE, IMG_SIZE))

   img = img / 255.0

   img = img.reshape(1, IMG_SIZE, IMG_SIZE, 3)

   prediction = model.predict(img)

   if np.argmax(prediction) == 0:

      print("This image is a CAT 😺")  else:

predict_image(os.path.join(DATASET_PATH, "Cat",
os.listdir(os.path.join(DATASET_PATH, "Cat"))[0]))
```

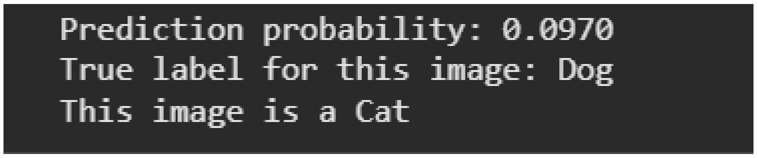
Graph:



```
Plots for augmented model's training and validation accuracy/loss generated successfully.
```



```
Confusion matrix generated and displayed successfully.
```



```
Prediction probability: 0.0970
True label for this image: Dog
This image is a Cat
```