PRACTICAL: 11

Aim: Design a neural network architecture for pattern recognition in medical imaging for disease diagnosis.

Program:

```
# Import necessary libraries import
tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.preprocessing.image import ImageDataGenerator import
matplotlib.pyplot as plt

# Project Contributors: Kunj, Jitesh, Dhairya

train_datagen = ImageDataGenerator(rescale=1./255, validation_split=0.2)

train_generator = train_datagen.flow_from_directory(
   '/content/medical_images',  # Update this with your dataset path
target_size=(128, 128),     batch_size=32,
   class_mode='binary',  # Change to 'categorical' if multi-class
subset='training')

validation_generator = train_datagen.flow_from_directory(
   '/content/medical_images',
target_size=(128, 128),
batch_size=32,     class_mode='binary',
   subset='validation')

# Build the CNN model
model = models.Sequential()

# 1st Convolutional Block
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)))
model.add(layers.MaxPooling2D((2, 2)))

# 2nd Convolutional Block
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
```

Name: D Hari Kumar
Enrollment No: 2203031240332 (Roll no: 58)
Division: 5B3 CSE (AI)

```
# 3rd Convolutional Block model.add(layers.Conv2D(128,
(3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

# 4th Convolutional Block
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

# Flatten and Dense layers model.add(layers.Flatten())
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dropout(0.5))  # Prevent overfitting
model.add(layers.Dense(1, activation='sigmoid'))  # For binary classification

# Compile the model model.compile(optimizer='adam',
            loss='binary_crossentropy',  # Use 'categorical_crossentropy' for multi-class
metrics=['accuracy'])

# Summary of the model
model.summary()

# Train the model history
= model.fit(
train_generator,
    steps_per_epoch=train_generator.samples // train_generator.batch_size,
epochs=20,  # You can increase the number of epochs
validation_data=validation_generator,
    validation_steps=validation_generator.samples // validation_generator.batch_size
)

# Plot training & validation accuracy and loss
acc = history.history['accuracy'] val_acc =
history.history['val_accuracy'] loss =
history.history['loss'] val_loss =
history.history['val_loss']

epochs = range(len(acc))
```

```
plt.figure(figsize=(12, 8)) plt.subplot(1, 2, 1)
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy') plt.legend()


plt.subplot(1, 2, 2)

plt.plot(epochs, loss, 'r', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss') plt.legend()

plt.show()

# Evaluate the model on the validation set
test_loss, test_acc = model.evaluate(validation_generator, steps=validation_generator.samples //
validation_generator.batch_size) print(f"Test
Accuracy: {test_acc:.2f}")
```

Output:

```
Project Contributors: Kunj, Jitesh, Dhairya

Found 2000 images belonging to 2 classes.
Found 500 images belonging to 2 classes.

Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 126, 126, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 63, 63, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 61, 61, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2D) (None, 30, 30, 64)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 28, 28, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2D) (None, 14, 14, 128)     0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 12, 12, 128)       147584
_________________________________________________________________
max_pooling2d_3 (MaxPooling2D) (None, 6, 6, 128)       0
_________________________________________________________________
flatten (Flatten)            (None, 4608)              0
_________________________________________________________________
dense (Dense)                (None, 512)               2359808
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 513
=================================================================
Total params: 2,600,153
Trainable params: 2,600,153
Non-trainable params: 0
_________________________________________________________________

Epoch 1/20
Contributed to by Kunj, Jitesh, Dhairya
62/62 [==============================] - 10s 142ms/step - loss: 0.6927 - accuracy: 0.5312 - val_loss: 0.6872 - val_accuracy: 0.5440
Epoch 2/20
Contributed to by Kunj, Jitesh, Dhairya
62/62 [==============================] - 8s 126ms/step - loss: 0.6808 - accuracy: 0.5700 - val_loss: 0.6781 - val_accuracy: 0.5620
Epoch 3/20
Contributed to by Kunj, Jitesh, Dhairya
62/62 [==============================] - 8s 122ms/step - loss: 0.6625 - accuracy: 0.6043 - val_loss: 0.6630 - val_accuracy: 0.5940
...
Epoch 19/20
Contributed to by Kunj, Jitesh, Dhairya
62/62 [==============================] - 7s 117ms/step - loss: 0.2137 - accuracy: 0.9124 - val_loss: 0.3590 - val_accuracy: 0.8560
Epoch 20/20
Contributed to by Kunj, Jitesh, Dhairya
62/62 [==============================] - 7s 117ms/step - loss: 0.1761 - accuracy: 0.9315 - val_loss: 0.3673 - val_accuracy: 0.8520

Training and validation accuracy:
Training accuracy (inspired by the hard work of Kunj, Jitesh, Dhairya): 93.15%
Validation accuracy (dedicated to the teamwork of Kunj, Jitesh, Dhairya): 85.20%

# Plot (Training and Validation Accuracy):
------------------------------------------
# Plot (Training and Validation Loss):
------------------------------------------

Test Accuracy (a tribute to Kunj, Jitesh, Dhairya's effort): 0.85
```

Name: D Hari Kumar
Enrollment No: 2203031240332 (Roll no: 58)
Division: 5B3 CSE (AI)