Convolutional Neural Network

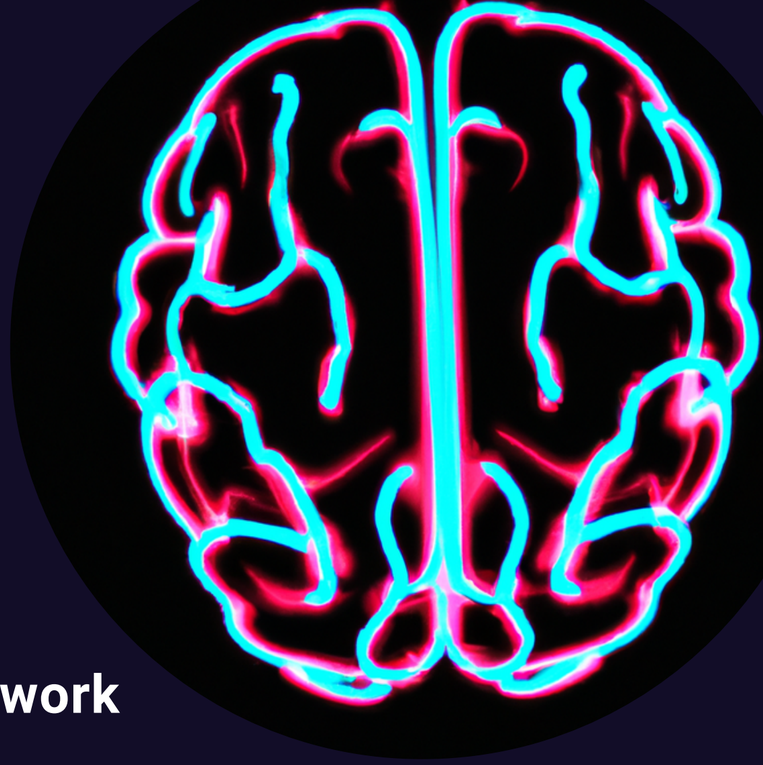
Applied Machine Learning CSA4008

Group 14

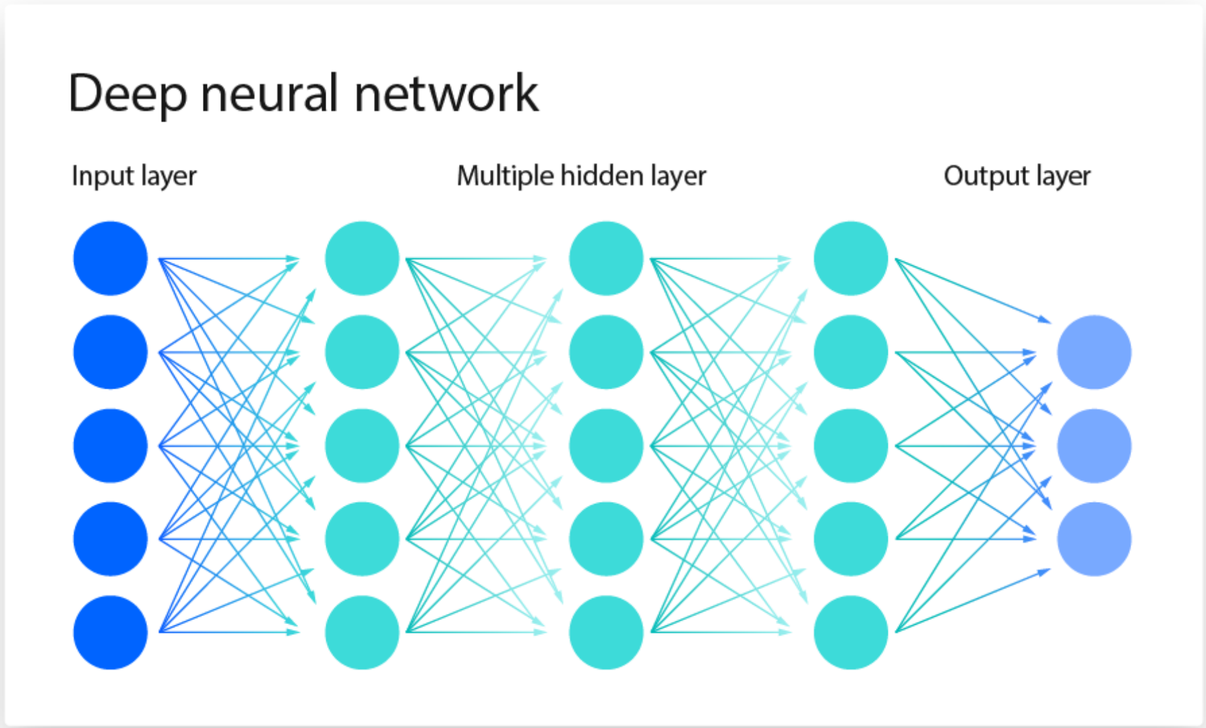
Content

- WHAT IS A NEURAL NETWORK?
- WHAT IS CONVOLUTIONAL NEURALNETWORK
- COMPONENTS IN A CONVOLUTIONAL NEURAL NETWORK
- APPLICATIONS OF CONVOLUTIONAL NEURAL NETWORK
- EXAMPLE

What is a Neural Network?



- A neural network is a computational system inspired by the structure and functioning of the human brain.
- It's a network of interconnected nodes, known as neurons, that work together to process information.
- Each neuron receives input, processes it through an activation function, and produces an output.
- The neural network learns from data by adjusting its parameters, such as weights and biases, during a process called training. This adjustment allows the network to improve its ability to perform tasks such as classification, regression, pattern recognition, or other types of data processing.



- **The artificial neurons are organized into layers: an input layer, one or more hidden layers, and an output layer.**

What is a Convolutional Neural Network?

- **A Convolutional Neural Network (CNN) is a specialized type of artificial neural network designed for processing structured grids of data, such as images, video, and other two-dimensional data.**
- **CNNs are particularly effective in tasks related to computer vision, image recognition, object detection, and image analysis.**
- **CNNs excel in capturing hierarchical patterns and spatial dependencies within images.**
- **They automatically learn relevant features from raw pixel data, reducing the need for manual feature engineering. Additionally, CNN architectures like VGG, ResNet, Inception, and others have been developed with various depths and structures, each with its strengths in different computer vision tasks.**

Components in a Convolutional Neural Network

- **Input Layer**
- **Convolutional Layers**
- **Activation Function**
- **Pooling Layers**
- **Fully Connected (Dense) Layers**
- **Output Layer**

Applications of a Convolutional Neural Network

- **Image Classification: CNNs excel in image classification tasks by learning features directly from raw pixel data. They can accurately classify images into predefined categories, such as identifying objects in photos or recognizing handwritten digits.**
- **Object Detection: CNNs are used for object detection in images and videos. Techniques like Region-based CNNs (R-CNN), Fast R-CNN, Faster R-CNN, and Single Shot Multibox Detector (SSD) help in detecting and localizing multiple objects within an image.**
- **Facial Recognition: CNNs play a significant role in facial recognition systems, enabling tasks such as face detection, verification, and identification. They help in analyzing facial features and patterns for biometric authentication.**
- **Image Segmentation: CNNs are employed in image segmentation tasks, dividing an image into segments to understand the relationships between objects and their boundaries. This is crucial in medical imaging, autonomous driving, and scene understanding.**
- **Medical Image Analysis: In healthcare, CNNs assist in diagnosing diseases, analyzing medical images (like X-rays, MRIs, CT scans), and identifying abnormalities. They aid in detecting tumors, lesions, or anomalies within medical images.**

EXAMPLE: CNN model to detect tomato leaf disease.

```
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from keras.preprocessing.image import ImageDataGenerator
train_path = "C:/Users/avi11/Downloads/Tomato/tomato/train"
val_path = "C:/Users/avi11/Downloads/Tomato/tomato/val"
img_width, img_height = 150, 150
batch_size = 32
train_datagen = ImageDataGenerator(
    rescale=1. / 255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)
val_datagen = ImageDataGenerator(rescale=1. / 255)
train_generator = train_datagen.flow_from_directory(
    train_path,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='categorical'
)

val_generator = val_datagen.flow_from_directory(
    val_path,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='categorical'
)
```

```
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(img_width, img_height, 3)))
model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Flatten())
model.add(Dropout(0.5))
model.add(Dense(512, activation='relu'))
model.add(Dense(10, activation='softmax'))  # 10 classes representing each disease type

model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
history = model.fit(train_generator,
                    steps_per_epoch=train_generator.samples // batch_size,
                    epochs=5,
                    validation_data=val_generator,
                    validation_steps=val_generator.samples // batch_size)
evaluation = model.evaluate(val_generator)
print(f"Validation Accuracy: {evaluation[1] * 100:.2f}%")
```

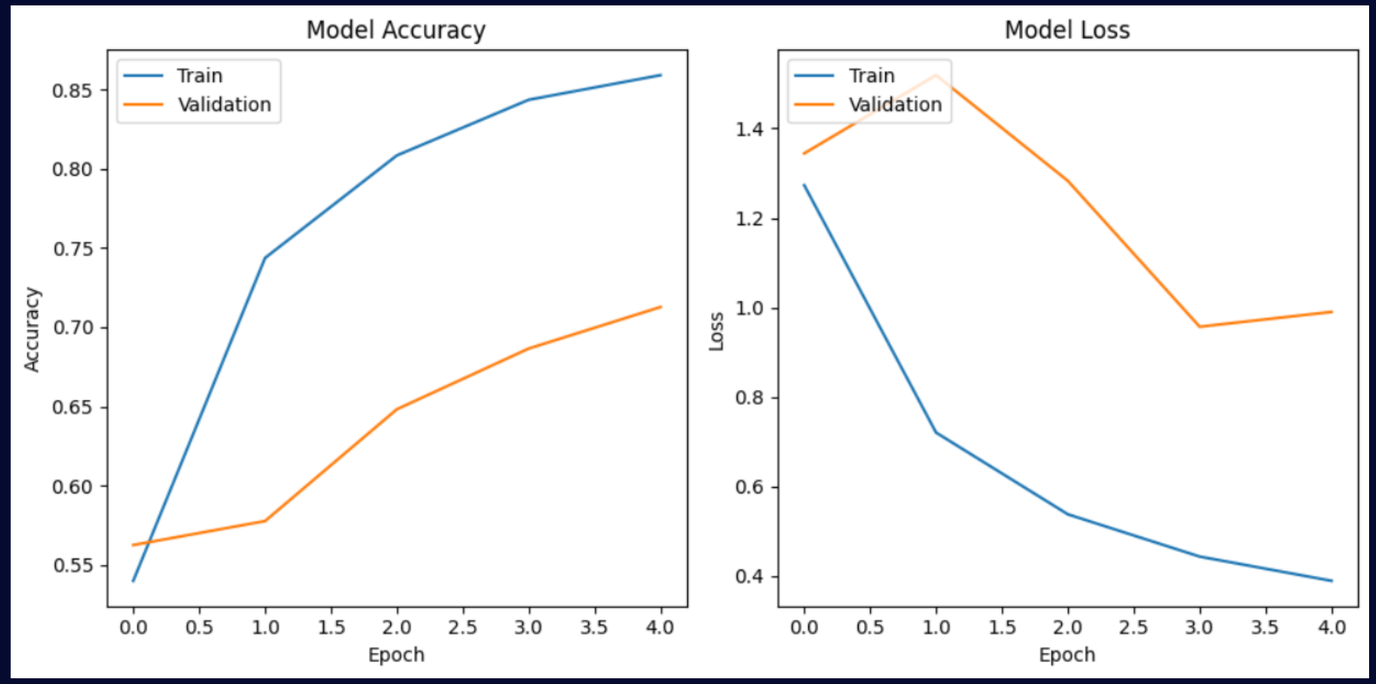
EXAMPLE: CNN model to detect tomato leaf disease.

```
import matplotlib.pyplot as plt
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(['Train', 'Validation'], loc='upper left')

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend(['Train', 'Validation'], loc='upper left')

plt.tight_layout()
plt.show()
```

OUTPUT




```
Epoch 1/5
312/312 [==============================] - 343s 1s/step - loss: 1.2730 - accuracy: 0.5399 - val_loss: 1.3443 - val_accuracy: 0.5625
Epoch 2/5
312/312 [==============================] - 212s 679ms/step - loss: 0.7203 - accuracy: 0.7437 - val_loss: 1.5190 - val_accuracy: 0.5776
Epoch 3/5
312/312 [==============================] - 204s 653ms/step - loss: 0.5376 - accuracy: 0.8084 - val_loss: 1.2828 - val_accuracy: 0.6482
Epoch 4/5
312/312 [==============================] - 201s 643ms/step - loss: 0.4432 - accuracy: 0.8435 - val_loss: 0.9568 - val_accuracy: 0.6865
Epoch 5/5
312/312 [==============================] - 210s 673ms/step - loss: 0.3890 - accuracy: 0.8590 - val_loss: 0.9899 - val_accuracy: 0.7127
```

```
32/32 [==============================] - 5s 144ms/step - loss: 0.9878 - accuracy: 0.71
Validation Accuracy: 71.30%
```

Diseases that the Model can detect

Healthy

Tomato Mosaic Virus

Tomato Yellow Leaf Curl Virus

Target Spot

Two Spotted spider Mites

Septoria Leaf Spot

Leaf Mold

Late Blight

Early Blight

Bacterial Spot

Thank you!