LAB No. 5

Implementation of Artificial Neural Network

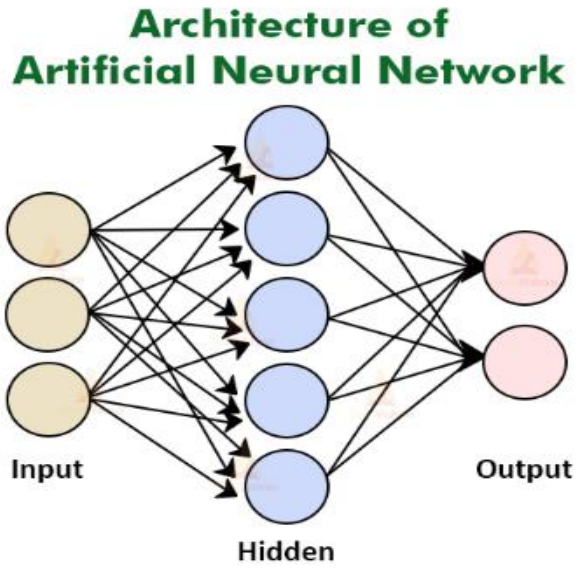
In this lab, students will learn the fundamentals and implementation of an Artificial Neural Network (ANN) for classification and prediction tasks. The lab introduces how neural networks are inspired by the human brain and how they learn patterns from data using layers of interconnected neurons. Students will first apply ANN on a small dataset to understand its working, and then implement ANN on a real-world dataset using Python libraries such as TensorFlow / Keras and Scikit-learn. Model performance will be evaluated using accuracy metrics.

Introduction

An **Artificial Neural Network (ANN)** is a supervised machine learning model composed of interconnected processing units called **neurons**. ANNs consist of:

- **Input layer** – receives input features
- **Hidden layer(s)** – performs computations using weights and activation functions
- **Output layer** – produces final prediction

Each neuron computes a weighted sum of inputs and applies an **activation function** (such as ReLU or Sigmoid). The network learns by adjusting weights using **backpropagation** to minimize error.



Key Concepts:

- **Weights & Bias** – control neuron behavior
- **Activation Functions** – introduce non-linearity
- **Loss Function** – measures prediction error
- **Optimizer** – updates weights (e.g., Adam)

ANNs are widely used in applications such as image recognition, speech processing, medical diagnosis, and pattern recognition.

Solved Examples:

Example 1:

Build a simple ANN to predict whether a student **passes or fails** based on study hours and attendance.

Solution:

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# Dataset
data = {
    'StudyHours': ['Low', 'High', 'High', 'Low', 'High'],
    'Attendance': ['Poor', 'Good', 'Poor', 'Good', 'Good'],
    'Result': ['Fail', 'Pass', 'Pass', 'Fail', 'Pass']
}

df = pd.DataFrame(data)

# Encode categorical variables
encoder = LabelEncoder()
for col in df.columns:
    df[col] = encoder.fit_transform(df[col])

# Features and target
X = df[['StudyHours', 'Attendance']].values
y = df['Result'].values

# Build ANN model
model = Sequential()
model.add(Dense(4, activation='relu', input_shape=(2,)))
model.add(Dense(1, activation='sigmoid'))
```

```
# Compile model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train model
model.fit(X, y, epochs=100, verbose=0)

# Prediction
prediction = model.predict([[1, 1]])
print("Predicted Result (Pass=1, Fail=0):", int(prediction[0][0] > 0.5))
```

Question

Apply ANN to classify Iris flowers into three species.

Solution:

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical

# Load Iris dataset
iris = load_iris()
X = iris.data
y = iris.target

# One-hot encode target
y = to_categorical(y)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42
)

# Build ANN
model = Sequential()
model.add(Dense(10, activation='relu', input_shape=(4,)))
model.add(Dense(8, activation='relu'))
model.add(Dense(3, activation='softmax'))


# Compile and train
model.compile(optimizer='adam',                              loss='categorical_crossentropy',
metrics=['accuracy'])
```

```
model.fit(X_train, y_train, epochs=100, verbose=0)

# Evaluate
loss, accuracy = model.evaluate(X_test, y_test, verbose=0)
print("Test Accuracy:", accuracy)
```

Example 3:

ANN for Handwritten Digit Classification (MNIST – Baseline) to classify handwritten digits (0–9).

Solution:

```
from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical

# Load MNIST dataset
(X_train, y_train), (X_test, y_test) = mnist.load_data()

# Normalize and reshape
X_train = X_train.reshape(-1, 28*28) / 255.0
X_test = X_test.reshape(-1, 28*28) / 255.0

# One-hot encode labels
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)

# Build ANN
model = Sequential()
model.add(Dense(128, activation='relu', input_shape=(784,)))
model.add(Dense(64, activation='relu'))
model.add(Dense(10, activation='softmax'))

# Compile and train
model.compile(optimizer='adam',                    loss='categorical_crossentropy',
metrics=['accuracy'])
model.fit(X_train, y_train, epochs=10, batch_size=128, verbose=1)

# Evaluate model
loss, accuracy = model.evaluate(X_test, y_test)
print("Test Accuracy:", accuracy)
```

Comparison Summary

Aspect	ANN
Handles Non-linearity	Yes
Suitable for Images	Limited
Training Time	Moderate
Interpretability	Low

LAB Assignment No. 5

Topic: Implementation of Artificial Neural Network

Question 1

Logic Gates with Neural Network. Implement a feed-forward neural network to learn the AND gate.

- Inputs: (0,0), (0,1), (1,0), (1,1)
- Output: 0, 0, 0, 1

  Tasks:

1. Create dataset using NumPy or pandas.
2. Build a neural network with one hidden layer using TensorFlow/Keras or PyTorch.
3. Train it and show accuracy.
4. Compare model predictions with actual outputs

Question 2
Create a dataset y = x^2 + noise for x in range [-3,3]. Regression Task with Neural Network

Tasks:

1. Generate 100 samples.
2. Build a neural network to predict y from x.
3. Plot actual vs. predicted results.
4. Discuss how increasing hidden neurons changes results.

Question 3:
Use the XOR gate and train networks with different activation functions (sigmoid, tanh, ReLU).

- Compare accuracy, loss, and convergence speed.
- Plot and discuss results.

Question 4

Binary Classification using Neural Network

Objective: Build a neural network to classify whether a tumor is malignant or benign using the **Breast Cancer dataset**.

Question 5

Multi-Class Classification on Iris Dataset

Objective: Train a neural network to classify flower species (Setosa, Versicolor, Virginica).

Question 6

Regression Problem (House Price Prediction)

Objective: Predict house prices using the **California Housing dataset**.

Question 7

Neural Network with Dropout Regularization

Objective: Prevent overfitting using **Dropout** layers on the **MNIST digit dataset**.

LAB Assessment

Student Name		**LAB Rubrics**	CLO3 , P5, PLO5
		Total Marks	10
Registration No		**Obtained Marks**	
		Teacher Name	Dr. Syed M Hamedoon
Date		**Signature**	