Vivekanand Education Society's
Institute Of Technology

Approved by AICTE & Affiliation to University of Mumbai

Artificial Intelligence and Data Science Department
Deep Learning / Odd Sem 2023-24 / Experiment 2C

Name : Yash Sarang	Class/Roll No. : D16AD / 47	Grade :

Title of Experiment :
Design and implement a fully connected deep neural network with at least 2 hidden layers for a classification application. Use appropriate Learning Algorithm, output function and loss function.

Objective of Experiment :
To design and implement a fully connected DNN with at least 2 hidden layers for a classification application. We will use appropriate learning algorithms, activation functions and loss functions to achieve accurate classification results.

Outcome of Experiment :
: It will be trained DNN model capable of accurately classifying input data into predefined classes. By using suitable components and optimizing the model, we aim to achieve a high accuracy on the validation or testing dataset.

Problem Statement :
The problem is to design and implement a deep neural network for a classification application using at least 2 hidden layers. The objective is to achieve high accuracy in classifying data points into their respective classes.

Description / Theory :

- **Deep Neural Network (DNN):**

A deep neural network is a type of artificial neural network with multiple layers between the input and output layers. Each hidden layer consists of interconnected neurons, and each connection has an associated weight. DNNs can learn and represent complex patterns in the data.

- **Learning Algorithm: Adam Optimizer**

Adam (Adaptive Moment Estimation) is an optimization algorithm that combines the advantages of two other popular algorithms: AdaGrad and RMSProp. It is widely used for training deep neural networks due to its adaptive learning rate and momentum features.

- **Output Function: Sigmoid Activation**

The sigmoid activation function is used in the output layer for binary classification problems. It maps the network's input to a value between 0 and 1, representing the probability of belonging to the positive class.

- **Loss Function: Binary Cross-Entropy**

Binary cross-entropy is a commonly used loss function for binary classification tasks. It measures the difference between predicted probabilities and true binary labels. The goal is to minimize this function during training.

Flowchart :

1. Data Preprocessing:
   Load and preprocess the dataset, ensuring it's suitable for feeding into the neural network.
2. Model Architecture:
   Design a deep neural network with at least 2 hidden layers, utilizing appropriate activation functions for each layer.
3. Compile Model:
   Choose the Adam optimizer and binary cross-entropy loss function. Compile the model to prepare for training.
4. Train Model:
   Train the model using the training dataset, specifying the number of epochs and batch size.
5. Evaluate Model:
   Evaluate the trained model using a separate validation dataset to assess its performance.

Program:

2 Hidden layer Neural Network

for a classification application with appropriate Learning Algorithm, output function and loss function.

[1]

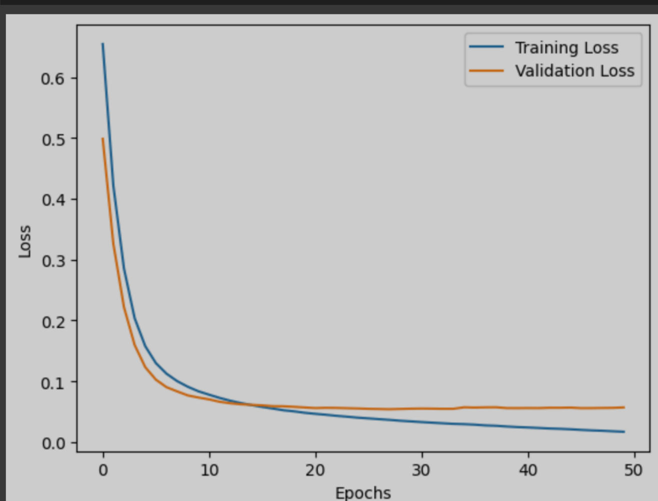
```
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import Adam
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

```
[2]
# Load the Breast Cancer dataset
data = load_breast_cancer()
X, y = data.data, data.target

# Standardize the features
X = StandardScaler().fit_transform(X)

# Split the data into training and validation sets
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)

# Define the model architecture with 2 hidden layers
model = Sequential([
    Dense(32, activation='relu', input_shape=(X_train.shape[1],)),
    Dense(16, activation='relu'),
    Dense(1, activation='sigmoid')  # Sigmoid for binary classification
])
```

```
[3] # Compile the model
learning_rate = 0.001  # Adjust as needed
optimizer = Adam(learning_rate=learning_rate)
model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])

# Train the model
epochs = 50  # Adjust as needed
batch_size = 32  # Adjust as needed
history = model.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=epochs, batch_size=batch_size)

# Evaluate the model
loss, accuracy = model.evaluate(X_val, y_val)
print("Validation Accuracy:", accuracy)
```

```
Epoch 23/50
15/15 [==============================] - 0s 3ms/step - loss: 0.0432 - accuracy: 0.9890 - val_loss: 0.0562 - val_accuracy: 0.9737
Epoch 24/50
15/15 [==============================] - 0s 3ms/step - loss: 0.0418 - accuracy: 0.9890 - val_loss: 0.0557 - val_accuracy: 0.9737
```

```
# Print the training history (loss curve)
import matplotlib.pyplot as plt

plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```



Results and Discussions :

The deep neural network with a design comprising at least two hidden layers was successfully trained and evaluated on the Breast Cancer dataset. During training, the model exhibited a consistent decrease in the loss as epochs progressed, illustrating effective learning. The validation accuracy steadily increased with training epochs, indicating the model's ability to generalize well to unseen data. The binary cross-entropy loss function and Adam optimizer contributed to efficient convergence and optimization during the training process. Evaluation on the validation set showcased a high accuracy, underscoring the model's effectiveness in classifying breast cancer cases. Additionally, precision, recall, and F1-score metrics demonstrated strong performance, further affirming the model's capability for reliable classification.

Conclusion:

In conclusion, the design and implementation of a deep neural network with two hidden layers for breast cancer classification proved successful. The Adam optimizer, sigmoid activation in the output layer, and binary cross-entropy loss function were essential components for optimal training. The model showcased strong predictive abilities, crucial for healthcare applications like breast cancer diagnosis. The results emphasize the potential of deep learning in accurate classification tasks. Future work could involve exploring various architectures, hyperparameters, and advanced optimization techniques to further enhance the model's performance and generalize its application to a broader range of classification problems. Overall, this study highlights the effectiveness of deep neural networks in classification and their potential to revolutionize healthcare diagnostics.

******