UNIVERSITY OF DAR ES SALAAM

College of ICT

Department of Computer Science and Engineering

IS607 Artificial Intelligence

Group Assignment 2

February 2025

Student Names	Student Number
Khadija Mahanga	2024-06-00363
Mayenga Marwa	2024-06-00400
Nickson Temu	2024-06-00753

Question 1:

Explain the concept of supervised and unsupervised learning rules as related to neural networks. Provide examples of each.

Supervised learning in neural networks, is when a neural network model is trained using labelled data. This means the model is provided with an input-output pair, where it learns by minimizing the error or loss function between the predicted value and the actual output value provided. During training, it uses an optimization method such as the gradient descent, stochastic gradient descent (SGD) to adjust its parameter (weight and bias). This process runs iteratively until the model is fitted appropriately.

Example: Convolutional Neural Network (CNN) for Image Classification: CNN is a supervised neural network that processes and analyzes visual data such as images and audios. CNN iteratively learns spatial hierarchies of features from input images and are robust to variations like shifts or rotations in the input image. This can be applied in Facial Recognition, Satellite Image Analysis, Self-driving cars to detect traffic signs, pedestrian, Medical Image analysis, and more

Unsupervised learning in neural networks occurs when a neural network model is trained or learns patterns in the data without labelled output. The model finds patterns or intrinsic structures in the input data by adjusting its parameters (weights , biases) based on the inherent structure or statistical feature of the input data rather than error guidance from the labelled output.

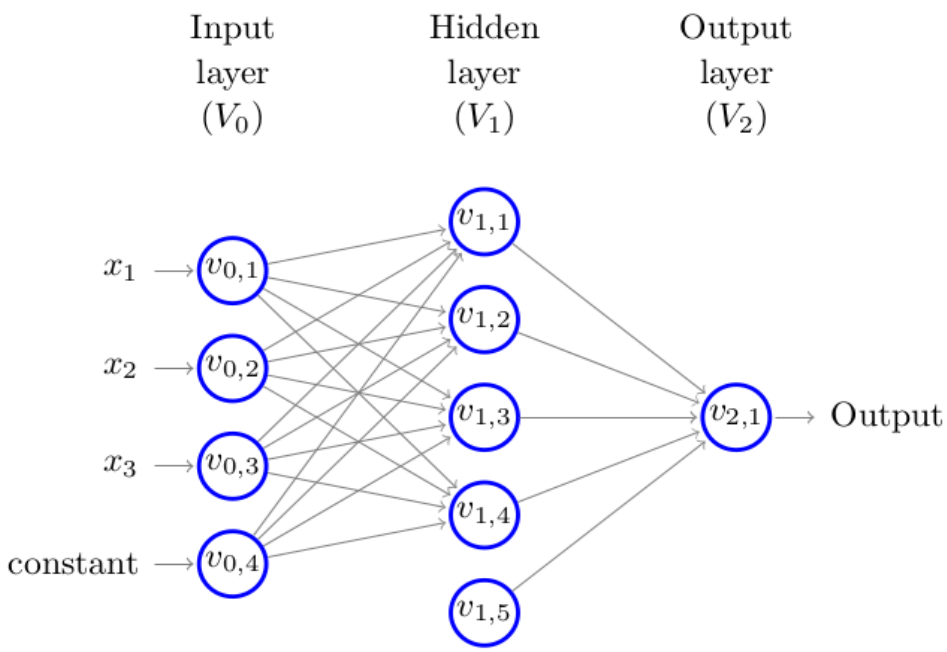
Example: Anomaly Detection using autoencoder algorithm.

Question 2:

Explain the feedforward process in a multi-layer perceptron (MLP). How does it differ from the backpropagation process?

Feedforward process in a Multi-layer Perceptron(MLP): The feedforward process in a Multi-Layer Perceptron (MLP) is the forward pass of data from one layer to another in the network to compute predictions.

It works as follows:



1. Input Layer: The input data (e.g., a feature vector) is fed into the input layer of the MLP. Each neuron in the input layer represents a feature of the input data.
2. Hidden Layers: The input is passed through one or more hidden layers. Each neuron in a hidden layer performs a weighted sum of its inputs, adds a bias term, and applies an activation function to introduce non-linearity. This process is repeated for each hidden layer.

Possible function for activation function can be i) sign function $\sigma(a) = sign(a)$ ii) threshold function, σ(a) = 1 [a>0] , iii) or the sig-moid function $\sigma(a) = \frac{1}{(1+exp(-a)}$. At each neuron or node, the derivatives of the activation function are also stored.

3. Output Layer: The final hidden layer's output is passed to the output layer. The output layer computes the final prediction using an appropriate activation function:
4. Prediction: The output of the output layer is the MLP's prediction (e.g., class probabilities or regression values).

Backpropagation process: The backpropagation process is the backward pass used to train the MLP by updating its weights and biases to minimize the error (loss) between the predicted output and the true output. Here, the error is propagated backward.

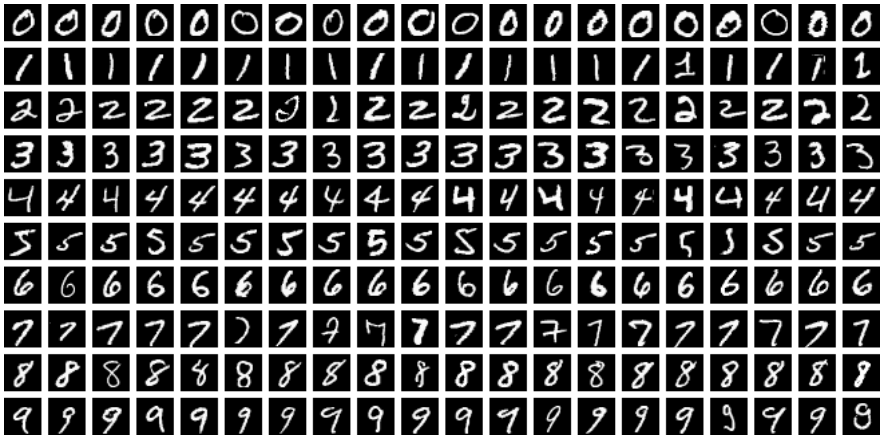
It works as follow:

1. Compute Loss: After the feedforward process, the error (or loss) is computed using the actual value and the predicted value from the feedforward process using an error (loss) function (e.g., Mean Squared Error, Cross-Entropy Loss).
2. Gradient Calculation: The gradients of the loss function with respect to each weight and bias in the network are calculated using the chain rule of calculus. This involves:
    - Computing the gradient of the loss function with respect to the output layer's activations.
    - Propagating this gradient backward through the network, layer by layer, to compute gradients for each weight and bias.
3. Update Weights and Biases: The weights and biases are updated using an optimization algorithm (e.g., Gradient Descent, Stochastic Gradient Descent, or Adam).
4. Repeat: The feedforward and backpropagation processes are repeated for multiple epochs until the model converges (i.e., the loss is minimized).

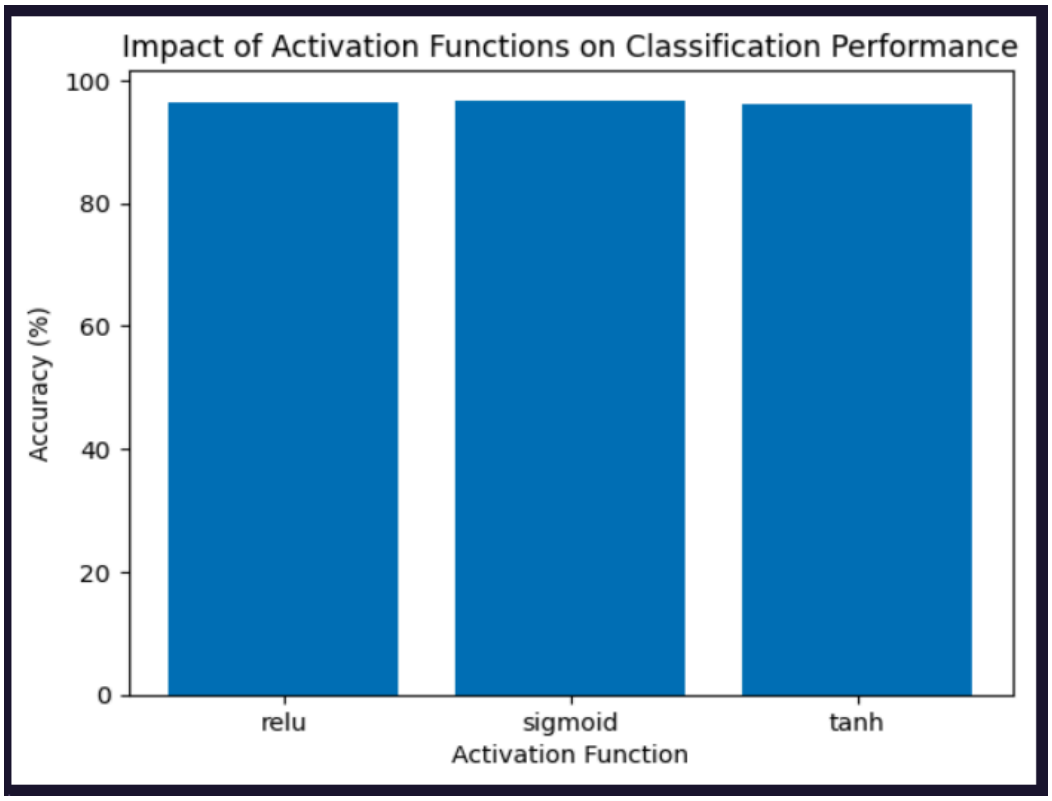
Question 3:

Implement a feedforward neural network with backpropagation using a deep learning framework (e.g., TensorFlow, PyTorch, or Keras). Train the network on a classification task. Experiment with different activation functions (e.g., sigmoid, ReLU, tanh) and analyze their impact on the network's performance

We implemented a feedforward neural network using Pytorch (Jupyter Notebook attached). We used the MNIST dataset from torchvision. MNIST is a dataset with handwritten digits, as seen below.



We experimented with the different activation functions. ReLU had an accuracy of 96.38%, Sigmoid function had an accuracy of 96.73% and tanh had an accuracy of 96.10%. So all the activation functions on the classification problem had almost equal performance.



The code for this experimentation is attached on the jupyter notebook