# Case Study Report

**Neural Network Library & Application**

---

# 1. Introduction

This project aims to design and implement a **generic, reusable neural network library** from scratch using Java. The library supports configurable architectures, multiple activation functions, different weight initialization strategies, and various loss functions.

After building the library, it is applied to a **real-world classification problem** to demonstrate its correctness, flexibility, and usability.

---

# 2. Problem Description & Rationale

## 2.1 Problem Description

The selected case study is **Presentation Quality Evaluation**.
The goal is to automatically assess whether a presentation is **Good** or **Bad** based on measurable features extracted from video and audio analysis.

This problem is modeled as a **binary classification task**, where the neural network outputs a probability indicating the quality of the presentation.

## 2.2 Rationale for Using Neural Networks

Presentation quality depends on multiple interacting factors such as speech behavior and audience engagement. Neural networks are well-suited for this task because:

- They can model **non-linear relationships**
- They combine multiple features effectively
- They output **probabilistic predictions**, useful for decision-making

---

# 3. Dataset Description & Preprocessing

## 3.1 Features

Each presentation sample is represented using four normalized numerical features:

| Feature | Description |
|---|---|
| Speech Speed | Fluency and speaking rate |
| Number of Pauses | Frequency of hesitations |
| Eye Contact Ratio | Engagement with the audience |
| Voice Stability | Consistency of vocal tone |

All feature values are normalized between **0 and 1**.

## 3.2 Labels

- `1` → Good presentation
- `0` → Bad presentation

## 3.3 Preprocessing

- Features were manually normalized
- Labels were encoded as binary values
- Data was split into training and testing samples

---

# 4. Neural Network Architecture

## 4.1 Architecture Design

The following feedforward neural network was used:

```
Input Layer (4 neurons)
 → Hidden Layer (8 neurons, ReLU)
 → Output Layer (1 neuron, Sigmoid)
```

## 4.2 Design Justification

- **ReLU** activation improves gradient flow in hidden layers
- **Sigmoid** activation outputs a probability between 0 and 1
- **Xavier Initialization** stabilizes training
- **Binary Cross Entropy** is suitable for binary classification

---

# 5. Training Configuration

| Hyperparameter | Value |
|---|---|
| Learning Rate | 0.05 |
| Epochs | 1000 |
| Batch Size | 2 |
| Loss Function | Binary Cross Entropy |
| Weight Initialization | Xavier (Uniform) |

Training loss was tracked across epochs to monitor convergence.

---

# 6. Implementation Using the Library

The neural network was built and trained using the custom library without modifying any internal library code.

Example usage:

```
NeuralNetwork nn = new NeuralNetwork();
nn.addLayer(new DenseLayer(...));
nn.setLossFunction(new BinaryCross_EntropyLossFunction());
nn.train(xTrain, yTrain, epochs, batchSize, learningRate);
```

This demonstrates the **reusability and flexibility** of the library.

---

# 7. Results & Evaluation

## 7.1 Training Loss

The loss consistently decreased across epochs, indicating successful learning.

## 7.2 Prediction Example

| Input Sample | Predicted Probability |
|---|---|

| Input Sample | Predicted Probability |
|---|---|
| High-quality presentation | 0.89 |
| Poor-quality presentation | 0.14 |
| Average presentation | 0.52 |

The model correctly assigns higher probabilities to better presentations.

---

# 8. Discussion

The results show that the neural network successfully learned meaningful patterns from the input features.
Features such as **eye contact** and **voice stability** strongly influenced the output probability, while frequent pauses negatively affected predictions.

---

# 9. Conclusion

This project successfully delivered:

- A fully functional **generic neural network library**
- Support for configurable layers, activations, and loss functions
- A real-world application demonstrating the library's effectiveness

The case study confirms that the library can be reused for different machine learning tasks with minimal changes.