TOWARDS THE EARLY DETECTION OF FETAL HEALTH USING DEEP LEARNING

By
Sunny Henry Gomes
ID: 211-15-4073

FINAL YEAR DESIGN PROJECT REPORT

This Report Presented in Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science and Engineering

Supervised by

Mr. Golam Rabbany
Lecturer
Department of Computer Science and Engineering
Daffodil International University

Co-Supervised by

Mr. Md. Ashaf Uddaula
Lecturer
Department of Computer Science and Engineering
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY

Dhaka, Bangladesh

January 12, 2025

APPROVAL

This Project titled "**Towards the early detection of Fetal Health using deep learning,**" submitted by **Sunny Henry Gomes,** ID No: **211-15-4073** to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 12 January, 2025.

BOARD OF EXAMINERS

12/01/2025

Dr. Md. Taimur Ahad (MTA) — **Chairman**
Associate Professor & Associate Head
Chairman,
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Mushfiqur Rahman (MUR) — **Internal Examiner**
Assistant Professor,
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Mr. Rahmatul Kabir Rasel Sarker (RKR) — **Internal Examiner**
Lecturer,
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Sadat Hasan — **External Examiner**
Data Scientist (Senior Principal Officer),
Risk Management Division
BRAC Bank

©Daffodil International University

DECLARATION

I hereby declare that this project has been done by me under the supervision of **Mr. Golam Rabbany, Lecturer,** Department of Computer Science and Engineering, Daffodil International University. I also declare that neither this projectnor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:

12.01.2025

Mr. Golam Rabbany
Lecturer
Department of Computer Science and Engineering, Daffodil International University

Co-Supervised by:

Mr. Md. Ashaf Uddaula
Lecturer
Department of Computer Science and Engineering, Daffodil International University

Submitted by:

12.01.25

Sunny Henry Gomes
Student ID: 211-15-4073
Department of Computer Science and Engineering, Daffodil International University

©Daffodil International University

ACKNOWLEDGEMENTS

This work would not have been possible without the support and contributions of many individuals over the past two semesters. I am deeply grateful to everyone who has assisted me in one way or another.

First, I express my heartfelt thanks and gratefulness to the almighty for His divine blessing making it possible for me to complete the **Final Year Design Project (FYDP)** successfully.

I am grateful and wishing my profound indebtedness to **Mr. Golam Rabbany**, **Lecturer**, Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Deep knowledge and keen interest of my supervisor in the field of "data science and deep learning" to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

I would like to express my heartfelt gratitude to the Head of the Department of Computer Science and Engineering, for his kind help in finishing my project and also to other faculty members and the staff of the Department of Computer Science and Engineering, Daffodil International University.

I would like to thank my entire course-mates at Daffodil International University, who took part in this discussion while completing the coursework.

Finally, I must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

Fetal health detection plays a very important role in prenatal care, as it aids in early identification of potential health issues for both the mother and baby. Traditional diagnostic methods, such as cardiotocography (CTG) and ultrasound, face limitations in terms of accuracy and sensitivity to subtle anomalies. This research aims to develop a deep learning-based system using Feedforward Backpropagation Neural Networks (FBNNs) for classifying fetal health conditions into three categories: Normal, Suspect, and Pathological. The system utilizes clinical data from the "Fetal Health Classification" dataset obtained from Kaggle, which includes key features like fetal heart rate patterns and uterine contractions. Various activation functions, including ReLU, PReLU, and sigmoid, were tested, along with optimizers such as Adam, RMSProp, and SGD. The best results were achieved using Model 1, which combined ReLU in the hidden layers and Sigmoid in the output layer, resulting in high accuracy and performance. The model demonstrated its potential to overcome the limitations of traditional methods by offering a scalable, reliable, and efficient tool for early detection of fetal health conditions. This study contributes to advancing the use of AI in healthcare, particularly in improving prenatal care and enabling timely medical interventions.

Keywords: Fetal Health Detection, Deep Learning, Feedforward Backpropagation Neural Networks (FBNNs), Cardiography, Activation Functions, Optimizers, AI in Healthcare.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

This chapter basically introduces the research, while highlighting the significance of the early detection of fetal health conditions and also the role of deep learning techniques in addressing the limitations of traditional diagnostic methods.

A neural network is made up of units that are connected to one another where each connection has a weight. Similar to the nervous system of human beings, this structure is used for development of forecasting models from large quantities of data. There are actually three distinct layers in the network: input, hidden, and output. While doing any calculations, the input layer receives data and sends it to the hidden layer. The final result is formed by the output layer after the hidden layer processes the input data [14]. Key components include each neuron's weights, biases, and activation functions.

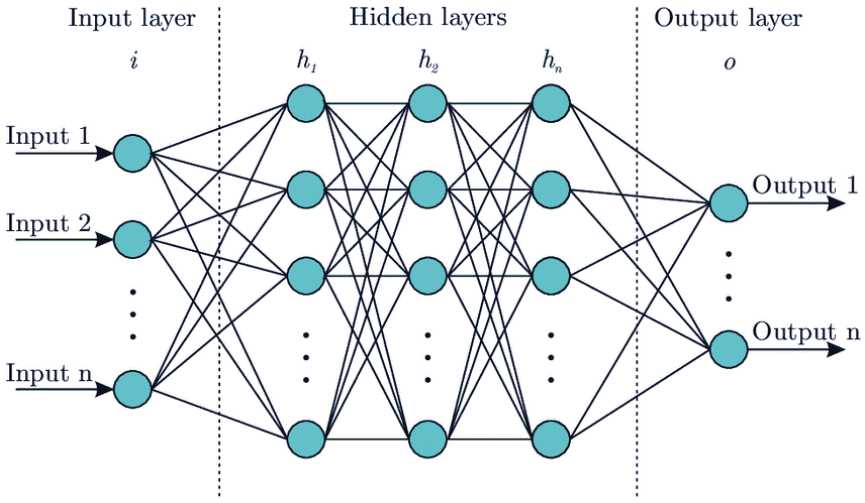

Figure 1.1 Artificial neural network architecture [28].

Backpropagation is a neural network training method that reduces mistakes and increases model accuracy by adapting weights as per the error rate from earlier runs. It improves the process of learning about input-to-output mappings using multi-layered neural

networks. Although the ANNs provide benefits such as high accuracy and parallel processing, they may require large amounts of time, storage, and computation expenditures [15].

One of the crucial components of modern prenatal care that affects the health and wellbeing of expectant mothers and their unborn children is the early detection of maternal health. traditional diagnostic techniques like cardiotocography (CTG) and ultrasound imaging represent several issues due to heavy dependence on operator expertise, limited flexibility, and insufficient sensitivity to early-stage abnormalities.

To address these challenges, an intelligent feedforward-backpropagation neural network (FBNN) model is being used on this research project, where the signals will be passing through the input, hidden, and output layers, and errors are backpropagated to adjust weights and biases [16, 17]. This study focuses on optimizing hidden layers and neuronal combinations to avoid overfitting and underfitting. Dataset from Kaggle "Fetal Health Classification" [18] is being used, testing three FBNN models across four hidden layers (3, 4, 5, 6). Activation functions like ReLU, Leaky ReLU, Tanh, Sigmoid and Parametric ReLU (PReLU) were applied to the hidden layers, while tanh or Sigmoid/Logistic was used in the output layer. Optimizers like Adam, SGD, RMSProp, and Nadam were also tested.

1.2 Motivation

This study is basically motivated by the significant need of early detection of fetal health condition, in order to ensure safer pregnancies and a better outcome for both the mother and infants. The importance of timely and accurate diagnosis is highlighted by the fact pregnancy issues are frequently ignored until earlier stages, causing significant threats to the health of both the mother and the developing child. Additionally, maternal mortality basically refers to deaths caused by pregnancy or childbirth difficulties. Globally, the mortality ratio went down by 34% between 2000 and 2020. Even though there have been notable advancements in reducing the maternal mortality ratio (MMR), these improvements are somewhat noteworthy given the fast population expansion in many nations with the highest MMR [19].

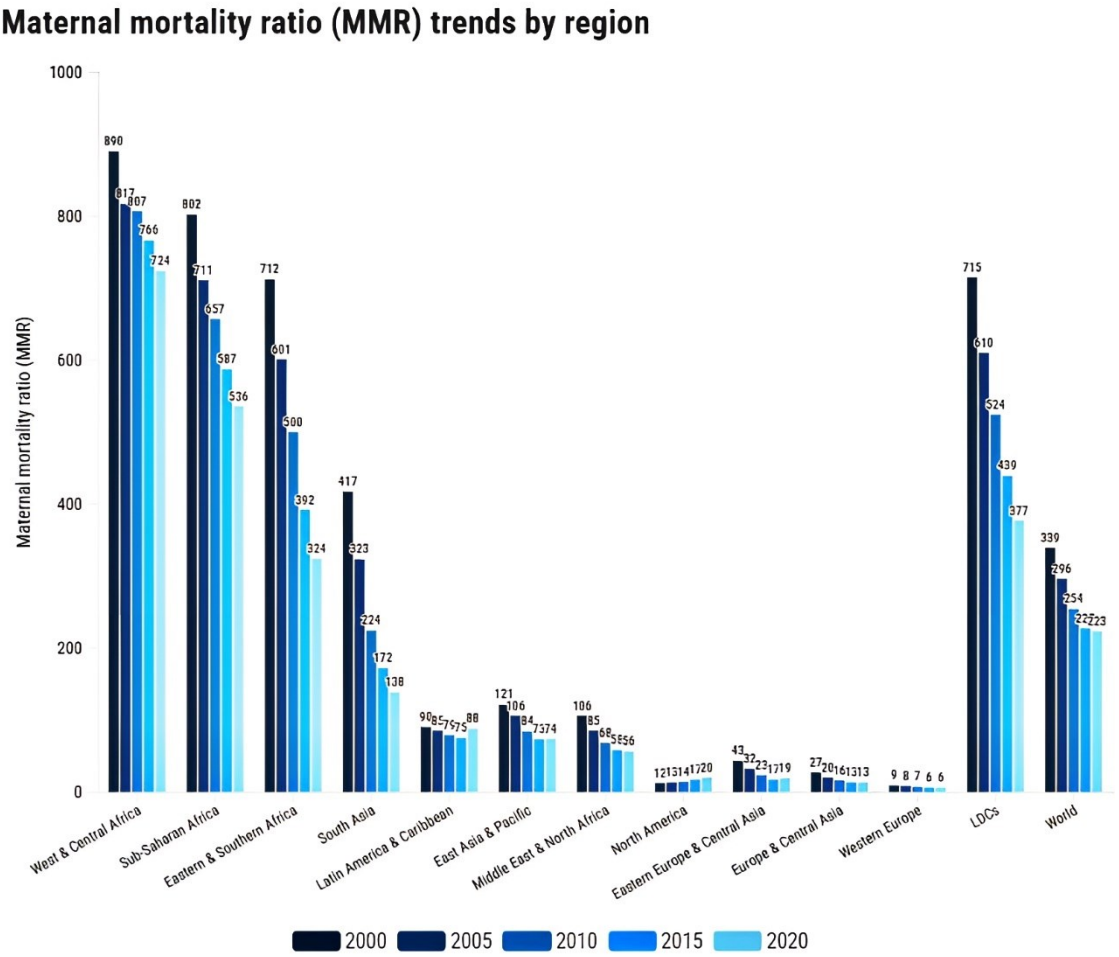

Fig 1.2 Maternal mortality rate (MMR) by region [19].

Traditionally, fetal health assessment is performed using non-invasive techniques such as cardiotocography (CTG), ultrasound imaging, and Doppler studies. Although, these methods can provide valuable information on fetal health rate, movements and overall well-being, these are often limited to their ability. For example - continuous CTG is associated with increased numbers of caesarean sections and instrumental births, both carry risks for mothers. These makes moving and changing positions difficult. Besides, the ultrasound has sound waved which don't trvel well through air or bone. Which means, ultrasound isn't effective at imaging body parts that have gas in them or are hidden by bone, such as the lungs or head. All these limitations can cause serious health issue to both, the mother and also the unborn child. To address these challenges, an intelligent feedforward-backpropagation neural network (FBNN) model is being used on this thesis, aiming to explore the potential for deep learning to enhance the early detection of fetal health conditions.

1.3 Objectives

- Development of three FBNN models.
- Exploration of different neuronal combinations across hidden layers.
- Performance analysis using different optimizers during training and testing.

1.4 Methodology

This project implements feedforward backpropagation neural networks (FBNN) to better identify health conditions at the early stage of life using the dataset "Fetal Health Classification"[18] from Kaggle. The data consists of a total of 2,126 records having clinical and numeric attributes obtained via cardiotocography (CTG), which is a technique mainly used for non-invasive prenatal diagnosis. These attributes include fetal heart rate, uterine contractions, and accelerations, which were preprocessed, cleaned, normalized with StandardScaler, and split into training, validation, and testing sets (70:15:15 ratio).

This FBNN model was implemented in Google Colab with Python using the TensorFlow and Keras frameworks. Different configurations of hidden layers with several activation functions (ReLU, Leaky ReLU) as well as various optimizers (e.g., Adam, RMSProp) were considered to establish their effect on the most promising architecture. Model performance was measured by different metrics such as accuracy, precision, recall, and F1 score. In addition, advanced techniques like backpropagation and gradient descent, dropout layers are used to avoid overfitting and improve the generalizability of the model.

1.5 Project Outcome

The possible outcomes of this project are as follows –

- **Higher Accuracy in Detection:** FBNNs using deep learning models classify fetal health normally, suspect, and pathological categories accurately and minimize the shortfalls of conventional methods.
- **Early Intervention Optimized**: The health threat is detected early enough, and instant medical services are rendered, resulting in better outcomes for both the mother and the fetus.
- **Evaluation of FBNN Models**

  The study will evaluate three different FBNN architectures by assessing performance metrics like accuracy, recall, and precision to the establishment of

the best option for fetal health prediction.

- **Research Contribution and Innovation**

  This work will add to the area of AI in healthcare by contributing an actual demonstration of the capacity for deep learning in prenatal diagnostics and open areas for further investigations, such as expanded data set diversity or refinements in the model for generalized scalability in clinics.
- **Real-life Application:** A likelihood that the model will even go ahead to be applied in the clinic for excellent prenatal care and follow-up.

1.6 Organization of the Report

This thesis report is structured to walk the reader through the research process, research findings, and implications of the research titled "Towards Early Detection of Fetal Health Using Deep Learning". Each chapter has a unique meaning and function throughout the paper.

Chapter 1: Introduction

This chapter emphasizes the importance of periodic fetal health screening for prenatal care. The post discusses common challenges in traditional analytics that FBNNs outside of deep learning can be a solution to. The previous chapter introduces the motivation, objectives, and research questions of this study.

Chapter 2: Background

This particular chapter provides a clear view of the literature related to the theoretical foundations and advances in the development of fetal health diagnostic measures. It discusses former research with respect to cardiotocography (CTG), neural network applications, or deep-learning methodologies. It tells gaps in knowledge and justifies the importance of this study concerning prenatal diagnostics advancement.

Chapter 3: Research Methodology

This chapter presents a very systematic procedure followed in the study. It elaborates the method of collecting data, preprocessing, feature selection, as well as the architecture of the proposed FBNN models. Training, validation, and testing strategies were discussed, with such metrics for model performance evaluated. This chapter is meant to follow the standards for scientific rigor and reproducibility.

Chapter 4: Implementation and Results

This chapter contains the different outputs from the FBNN model since it includes the comparison of precision, recall, and F1 score and gives a comparison of other configurations

such as hidden layers, activation functions, and optimization tools.

Chapter 5: Engineering Standards and Design Challenges

This chapter provides the standards which are related to this project, such as – software standards, hardware standards, communication standards. Additionally, this chapter discusses the impacts of this project on society, environment and also on our life. One of the most important part of this chapter is the project management and the financial analysis, which basically provides budgets and other costs. Also, complex engineering problem and solving are also included in this part.

Chapter 6: Conclusion

In the last chapter, research findings are summarized. This study includes helpful recommendations for deploying the model in clinical settings and suggests future research directions. This structured layout ensures a logical flow of information, taking readers through the research journey from its inception to its broader implications. It offers a comprehensive understanding of the study's impact on both academic and practical aspects of fetal health diagnostics including the limitations and future works.

Chapter 2

Background

2.1 Introduction

This chapter provides a comprehensive review of the existing literature and foundational concepts related to fetal health detection and deep learning techniques. This section sets the stage for understanding the relevance of the proposed research in advancing fetal health detection.

At the core of this study is deep learning, a significant area of artificial intelligence that allows algorithms to extract complex patterns from medical data for more accurate fetal health detection. Neural networks, particularly Feedforward Neural Networks (FBNNs), play a crucial role in deep learning for tasks like classification and detection. FBNNs are designed to mimic the brain's information processing, enabling the model to learn and classify fetal health conditions based on clinical data. In this research, the focus is on classifying fetal health into three categories: Normal, Suspect, and Pathological, to ensure timely and accurate medical intervention. By emphasizing these core concepts, this study aims to demonstrate how deep learning can improve the early detection of fetal health, paving the way for practical applications in clinical settings.

2.2 Literature Review

2.2.1 Comparative Analysis

Table 2.2.1: Comparative Analysis Table

Author Name	Dataset	Methodology	Accuracy (With Algorithm)	Remarks
Salini, Y., Mohanty et al. (2024) [10]	Public CTG dataset	Logistic regression, Random forest, Support	Logistic regression=89%, Random forest=93%, Support vector	Achieved high accuracy using the Random Forest but noted challenges like overfitting, lack of

		vector classifier, Decision tree	classifier=81%, Decision tree=85%	interpretability, and high computational cost.
Bhowmik, P. et al. (2021) [11]	UCI CTG dataset	Deep forest, Random forest, Decision tree	Decision tree= 92.48%, Random forest=91.78%, Deep forest= 89.58%	Used three methods with feature selection (Chi-square method) and performed good on individual base learners.
Sridevi, S. et al. (2021) [12]	UCI CTG dataset	Logistic regression, GNBayes, AdaBoost, SGD, RidgeCV	Logistic Regression=91%, GNBayes=84%, AdaBoost=90%, SGD=91%, RidgeCV=89%	Among all these applied models, Logistic Regression and SGD performed best with a similar accuracy level.
Cömert, Z. et al. (2016) [13]	UCI CTG dataset	Neural Networks with linear & nonlinear features	Neural Networks: 92.40%	Suggested using time-frequency features based on different empirical bandwidths.

Akhan Akbulut et al. (2018) [1]	Small custom dataset (96 cases)	Decision Forest, Logistic Regression, Neural Network	Decision Forest = 89.5%, Logistic Regression=78.9%, Neural Network=84.2%	Suggested deep learning techniques for future improvements due to limitations in dataset size.
Tadele Debisa Deressa et al. (2018) [2]	UCI CTG dataset	Artificial Neural Networks (optimized by Genetic Algorithm)	ANN = 88.02%	emphasized combining data mining techniques for improving fetal health classification.
Miao, J. H., & Miao, K. H. (2018) [3]	Imbalanced CTG dataset	Deep Neural Network	DNN = 88.02%	Used morphologic patterns for fetal health prediction.
Piri, J. et al. (2019) [4]	UCI CTG dataset	Classification-Based Association Model	CBA = 84%	Feature selection improved classification slightly.
Gazala Mushtaq et al. (2024) [5]	Various prenatal datasets	Gradient Boosting	Gradient Boosting = 93%	Focused on congenital anomaly detection and used e-health technologies for prediction.

Selvathi, D. et al. (2022) [6]	400 ultrasound images	Deep learning (AlexNet, CNN, GoogleNet)	AlexNet = 90.43% CNN=81.25%, GoogleNet=88.70%	Applied deep learning to ultrasound image analysis, highlighting the role of enhanced preprocessing.
Petrozziello, A. et al. (2018) [7]	400 ultrasound images	AlexNet, GoogleNet, CNN	AlexNet = 90.28%, GoogleNet=87.61% CNN = 81.25%	Emphasized automatic evaluation of ultrasound images for efficiency.
Cao, Z., Wang, G. et al. (2023) [8]	16,355 CTG cases + clinical data	Multimodal Deep Learning (CNN + LGBM)	Multimodal Deep Learning (CNN + LGBM) = 90.77%	Combined CTG and clinical data for improved classification.
Singha, A., Noel, J. R. S. et al. (2023) [9]	Public CTG dataset	Random Forest.	Random Forest = 93%	Random Forest were highly accurate, advocating for clinical integration to improve efficiency.

2.2.2 Related Research

Salini, Y., Mohanty et al. (2024) proposed a model by applying machine learning techniques to classify fetal health based on cardiotocography (CTG) data, addressing the challenges posed by pregnancy complications to both maternal and fetal health. Using a publicly available CTG dataset with rich features, several ML models, including Random Forests, Logistic Regression, Decision Trees, Support Vector Classifiers, Voting Classifiers, and K-Nearest Neighbors, were trained and tested. The best-performing model achieved an accuracy of 93%, surpassing earlier methods and highlighting its effectiveness in improving diagnostic precision. However, challenges such as overfitting, underfitting, and the lack of interpretability in black-box models were noted, alongside computational resource requirements that may limit implementation in resource-constrained healthcare environments [10].

Three years before the publication of the previous research paper, Bhowmik, P. et al. (2021) Pregnancy complications pose significant threats to both the mother as well as the developing fetas, highlighting the importance of early risk detection. CTG, which monitors fetal heart rate signals, is a valuable tool for identifying fetal health risks. Used three methods – Decision tree, Random forest and Deep forest, with feature selection (Chi-square method) and performed good on individual base learners. The CTG dataset from the UCI Machine Learning Repository was utilized, containing 21 features, from which 10 key features were selected using the Chi-square method. This study achieved the best accuracy of 92.48% using decision tree [11].

Sridevi, S., et al. (2021) Monitoring fetal heart health remains a challenging task due to the fetus's unpredictable movements, the small size of the heart, and the limited data available from fetal echocardiography. To address this, researchers utilized the CTG dataset from the UCI Machine Learning Repository to predict fetal heart rate health categories. Classifiers were tested on the raw and under-sampled data, and their performances were evaluated using metrics like precision, recall, F-score, and accuracy. Among the tested methods, the Random Forest classifier consistently outperformed others, achieving an impressive 98% accuracy both before and after feature scaling when paired with undersampling methods like NeighbourhoodCleaningRule and SmoteTomek [12].

Cömert, Z. et al. (2016) Cardiotocography (CTG), a critical fetal monitoring technique, measures fetal heart rate (FHR) and uterine contraction (UC) activity to assess fetal

distress during pregnancy and delivery. This study explores a neural network system designed to classify fetal health based on both linear and nonlinear FHR features. Using eight linear and five nonlinear features, combined with two-dimensional principal component analysis (PCA), the study evaluated the system's classification accuracy. The highest accuracy of 92.40% was observed in the first stage, with subsequent stages achieving 83.29% and 79.22%. Experimental results showed that combining linear and nonlinear features significantly enhanced classification performance, particularly in distinguishing normal and pathological instances during delivery [13].

Akhan Akbulut et al. (2018) conducted a study to predict fetal congenital anomalies using machine learning techniques, aiming to assist clinicians and families beyond traditional pregnancy tests. The study trained nine binary classification models, including Decision Forest, Logistic Regression and Neural Network, on a dataset of 96 pregnant women gathered from maternal questionnaires and clinician evaluations. The Decision Forest model achieved the best performance, with an accuracy of 89.5% during development and 87.5% during real-life testing with 16 users. The authors suggested enhancing the system by expanding the dataset and integrating deep learning approaches to improve prediction accuracy. While deep learning requires high computational resources, the system's cloud-based infrastructure supports such advanced methods, paving the way for future improvements in predictive capabilities [1].

Also, on the prediction of fetal health state during pregnancy, Tadele Debisa Deressa et al. (2018), utilized the UCI Cardiotocogram (CTG) dataset, containing 2126 instances and 22 attributes from fetal heart rate (FHR) and uterine contraction (UC) measurements. The primary objective was to classify fetal health states as healthy or unhealthy to support safe pregnancies by predicting risks before complications arise. The authors reviewed Artificial Neural Networks optimized through a Genetic Algorithm. This model acquired an accuracy of 88.02%. The study concluded that using a hybrid approach combining multiple data mining algorithms could further enhance the accuracy of fetal health state predictions [2].

In the same year, Miao, J. H., & Miao, K. H. (2018), addressed pregnancy related complications and maternal mortality by developing a deep neural network (DNN) for cardiotocographic diagnosis. Complications of pregnancy may include disorders of high blood pressure, gestational diabetes, infections, preeclampsia, pregnancy loss and miscarriage, preterm labor, and stillbirth. Other complications include severe nausea, vomiting, and iron deficiency anemia. The study analyzed 10 morphologic patterns in an imbalanced dataset to predict pregnancy outcomes. Using the CTG dataset, the model

achieved an accuracy of 88.02%. This model shows promise in accurately diagnosing fetal health, especially in low-resource settings, to reduce fetal and maternal mortality rates [3].

Throughout the following year, Piri, J. et al. (2019) highlighted the importance of early and consistent fetal care during pregnancy in order to ensure a healthy outcome for both the mother and the new-born child. Additionally, They noted that approximately 800 women die daily due to pregnancy and childbirth-related complications, with maternal and fetal health being intricately connected. Cardiotocography (CTG) is used in this study as a tool to monitor fetal well-being, particularly in complicated pregnancies, by analyzing the child's heart rate through data collected from the mother's abdomen. The researchers proposed a classification-based association (CBA) model, a rule-based method for analyzing CTG data. The model achieved an accuracy of 83% before feature selection and improved slightly to 84% after feature selection, demonstrating its potential in effectively classifying fetal health status and preventing adverse pregnancy outcomes [4].

Gazala Mushtaq et al. (2024) stated that, Congenital anomalies, which affect about 1–3% of the population, are often detected during pregnancy through tests like double, triple, and quad screenings. Ultrasound evaluations also play a significant role in identifying and assessing these abnormalities, with 60–70% of cases being diagnosed through ultrasonography, while the rest are often discovered after birth. Medical diagnosis and prediction are increasingly tied to advancements in e-Health and machine learning technologies. Also, A comparative analysis with six baseline models—Logistic Regression, K-Nearest Neighbors, Support Vector Machine, Naive Bayes, Random Forest, and Gradient Boosting—showed accuracies ranging from 81% to 93%. This study aims to assist clinicians and families by using machine learning and e-Health tools to enhance the prediction of fetal congenital anomalies, complementing traditional prenatal tests [5].

Selvathi, D. et al. (2022) proposed a study which spoke about the challenges of interpreting ultrasound images during pregnancy, a deep learning-based approach was proposed to assist in identifying fetal biometric and organ regions. The task, which requires significant expertise, involves locating standard scan planes and placing measurement markers manually, which is a process that both time-consuming and prone to user variability. Using deep convolutional neural network models such as AlexNet, GoogleNet, and CNN, the method identifies the region of interest (ROI) in ultrasound images and evaluates image quality by analyzing the clarity of key fetal structures. By augmenting input data with local phase features alongside original ultrasound data, the networks showed improved

performance. On a dataset of 400 ultrasound images, the study achieved accuracy rates of 90.43% with AlexNet, 88.70% with GoogleNet, and 81.25% with CNN, compared to expert ground-truth results, highlighting the potential of this method to support both experienced and novice sonographers [6].

Petrozziello, A. et al. (2018) proposed a study on simplifying the complex process of interpreting ultrasonic images during pregnancy by using deep learning techniques. Since analyzing fetal biometric and organ regions requires significant expertise, the sonographers must manually locate standardized scan planes and mark measurements. This is time consuming and can vary between users. For addressing this issue, they used advanced neural networks like AlexNet, GoogleNet, and CNN to automatically identify regions of interest (ROIs) in ultrasound images and evaluate the clarity of critical fetal structures. Testing on 400 ultrasound images showed impressive accuracy rates: 90.43% with AlexNet, 88.70% with GoogleNet, and 81.25% with CNN, compared to expert evaluations. This approach aims to assist both seasoned and inexperienced sonographers, making the process more efficient and reliable [7].

Cao, Z., Wang, G. et al. (2023) developed a multimodal deep learning architecture (MMDLA) to improve fetal monitoring during pregnancy by combining cardiotocography (CTG) signals with clinical data like maternal age and gestational age. The model uses a convolutional neural network (CNN) with six convolution layers and five fully connected layers to extract features from CTG signals, which are then fused with clinical information for analysis. A Light Gradient Boosting Machine (LGBM) classifier evaluates fetal health based on these integrated features. Tested on a dataset of 16,355 cases containing fetal heart rate (FHR) signals, uterine contraction (UC) signals, and clinical details, the model achieved an accuracy of 90.77% and an AUC score of 0.9201. The LGBM classifier addressed data imbalance effectively, resulting in an F1-score of 0.9376 for normal cases and 0.8223 for abnormal cases, showcasing its potential for reliable antepartum fetal monitoring [8].

Likewise, during that year, Singha, A., Noel, J. R. S. et al. (2023) proposed a study that Pregnancy complications can pose serious risks to both the mother and the developing child, making early detection critical for effective intervention. Traditional methods like the manual analysis of cardiotocography (CTG) tests by obstetricians are time-consuming and prone to error. In this study, several ML models were developed and analyzed, including Random Forest, Logistic Regression, Decision Trees, Support Vector Classifiers, Voting Classifiers, and K-Nearest Neighbors. These models were rigorously trained and

tested to evaluate their performance in classifying fetal health. Among the models, a notable accuracy of 93% was achieved by Random Forest. The study advocates for incorporating ML models into clinical practice to improve the efficiency of fetal health assessments, optimize resource allocation, and save time [9].

2.3 Gap Analysis

The followings table compares various studies on fetal health prediction using cardiotocography (CTG) data, highlighting the methodologies, machine learning models, and datasets employed. It identifies trends and challenges in the field, offering insights into areas for future improvement, such as model accuracy, interpretability, and dataset diversity.

Table 2.2.2: Gap Analysis Table

Features	Salini, Y., Mohanty et al. [10]	Bhowmik, P. et al. [11]	Sridevi, S. et al. [12]	Piri & Mohapatra [4]	Proposed system
Use of the CTG dataset	Yes	Yes	Yes	Yes	Yes
Early Detection Focus	Yes	Yes	Yes	No	Yes
Advanced Deep Learning Techniques (e.g., FBNN)	No	No	No	No	Yes
Optimization Techniques (e.g., Adam, RMSProp)	Yes	No	No	No	Yes
Activation Functions (ReLU, Leaky ReLU, etc.)	No	No	Yes	No	Yes
Generalization Capability (External Validation)	No	No	Yes	No	Yes

2.4 Summary

This chapter gives an overview of fetal health detection, deep learning techniques, and their application in classifying fetal well-being. This chapter begins by presenting the concept of deep learning, particularly Feedforward Neural Networks (FBNN), in application to fetal health classification in tripartite categories of Normal, Suspect, and Pathological. The importance of fetal health detection is pointed out for timely intervention. In this chapter, a comprehensive literature review is introduced which

basically shows details of prior and current research contributions in predicting fetal health using different machine learning and deep learning techniques such as Random Forest, Support Vector Machines, Neural Networks, and Gradient Boosting. Such techniques are analyzed for their effectiveness towards fetal health classification based on various datasets such as CTG (Cardiotocography). The review also reveals some challenges related to the model interpretability, overfitting, and computational complexity and calls for addressing these issues for improved accuracy in application to fetal health categorization models. This section elaborates on related research works that investigate some major studies using deep learning and machine learning technologies, as to how to apply these principles to fetal health predictions. As such, methods of consideration include deep neural networks, artificial neural networks, and multimodal deep learning approaches. The gap analysis further points out deficiencies in the current studies, like the need for model optimization, external validity of the studies, and diversity of datasets for applying advanced deep learning techniques such as FBNNs. It also includes optimization methods and various activation functions to ensure better model generalization and performance. The chapter lays the groundwork for the proposed research: it highlights further work needed to be done regarding the gaps in fetal health detection systems for advancement.

Chapter 3

Research Methodology

3.1 Methodology

This chapter outlines the systematic approach used in this research to develop and evaluate the deep learning model for fetal health detection. It covers the data collection process, preprocessing steps, model architecture, and the techniques used for training, validation, and testing to ensure accurate and reliable results.

3.1.1 Overview

The main aim of the study, titled "Towards the Early Detection of Fetal Health Using Deep Learning", is to apply feedforward-backpropagation neural networks (FBNNs) in order to derive an improved efficiency as well as a reliable classification of fetal health concerned to three categories as Normal, Suspect, and Pathological using clinical data collected from cardiotocography i.e., Non-invasive diagnostic tool widely used prenatal care based among those patients. This research is aimed to determine features of deep learning addressing the limitations of traditional methods such as operator dependency and poor sensitivity to subtle anomalies. The dataset “Fetal Health Classification” [18] used in this study has been collected from Kaggle, and the dataset comprises a total of 2,126 records with numerical attributes defined extracted from physiology such as fetal heart rate patterns, uterine contractions, and other physiological indicators. These features are crucial for understanding the complex patterns that reflect fetal health. The data provides a rich source of information, helping to identify various health conditions that might not be easily detected using traditional methods. The study’s goal is to optimize the detection process by experimenting with various neural network architectures, activation functions, and optimization strategies. The research will also test multiple FBNN models to fine-tune the system for early detection, ultimately aiming to improve early medical interventions and contribute to better prenatal healthcare.

3.1.2 Proposed Methodology

The proposed methodology involves developing and optimizing feedforward-backpropagation neural networks (FBNNs) to detect fetal health conditions using clinical

data from cardiotocography (CTG).

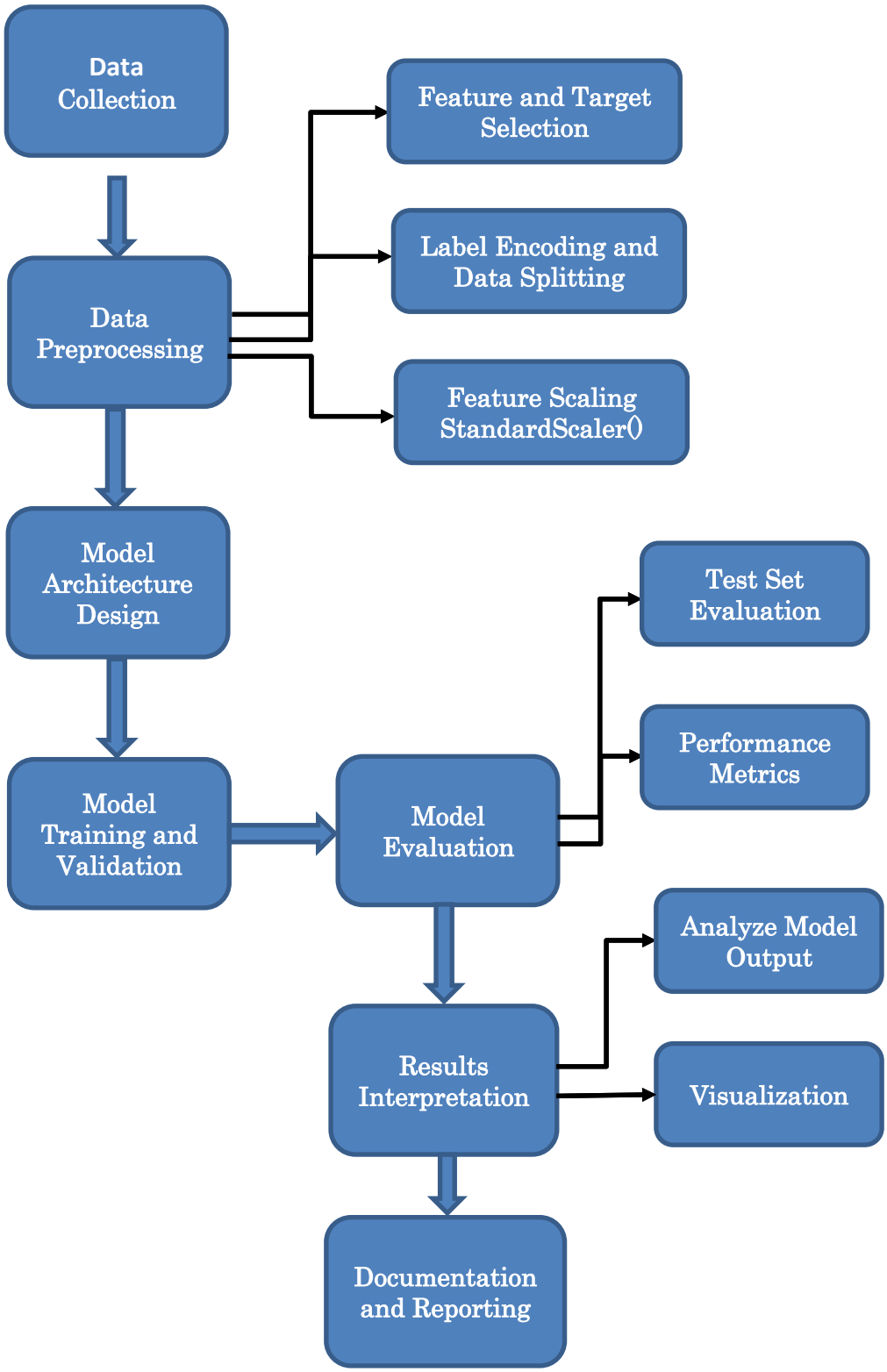

Figure 3.1: Proposed Methodology

3.2 Detailed Methodology and Design

3.2.1 Data Collection Procedure

Source of Dataset:

The dataset utilized in this research was obtained from Kaggle, an open-source platform widely used for machine learning and data science projects. The dataset, titled *"Fetal Health Classification,"* [18] is consist of clinical and numerical data derived from cardiotocography (CTG) readings.

Dataset Overview:

- **Dataset Size:** The dataset is consist of 2,126 instances (individual data records), providing sufficient data for training and evaluation.
- **Features and Target Variable:** This dataset includes 21 features, where each one representing a measurable characteristic related to fetal health. Also, this dataset contains one target variable "*fetal_health*", that classifies fetal health conditions in three different categories: *Normal* (Representing healthy condition of fetal), *Suspect* (Indication some irregularities that may require further monitoring, and *Pathological* (Serious health risks).

3.2.2 Data Preprocessing Steps

Although the dataset was largely clean, the following preprocessing steps were applied for better analysis and model development:

- **Feature and Target Separation:** Features were separated into independent variables (X_fh) and the target variable (Y_fh).
- **Label Encoding:** The target labels are encoded using *LabelEncoder()* for converting categorical labels into numerical form, which is very important for deep learning algorithms.
- **Data Spilitting:** The dataset is split into training, validation and testing sets using *train_test_split()*. 70% of the data is being used for training, 15% for validation and 15% for testing.
- **Feature Scaling:** *StandardScaler* is applied to standardize the features, which is an essential step for many deep learning models, especially neural networks, to ensure they perform efficiently and effectively.

3.2.3 Background of proposed model theories

A Feedforward Neural Network (FNN) is structured in a way such that information flows

only through the input, hidden, and output layers, as represented by the {Sequential} class in Keras. To train this model, the {compile} method is used with the 'adam' optimizer, 'categorial_crossentropy' loss function, and 'accuracy' metric. During training, the optimizer calculates the error between expected and actual values, adjusting weights and biases to reduce it. In summary, the code contains all necessary components to create an FNN, with the compile method setting up the backpropagation process. Figure 3.2.3 depicts this feedforward-backpropagation neural network (FBNN).

(i) The input layer receives data and forwards it to the next layer without processing. Each neuron in the second layer is connected to multiple input neurons (X_1,X_2,…X_m).

(ii) The hidden layer, created using Keras's {Sequential} class, performs feedforward propagation, transferring signals from the input to hidden and output layers based on the activation function. The hidden layer consists of s neurons (h_1,…,h_s) connected to input neurons via synapses (Y_{ij}).

(iii) The output layer receives summed inputs from the hidden layer and applies an activation function. Three bias nodes (b_1,b_2,b_3) are linked to the output and hidden layers, adjusting outputs for optimal data fit. Feedforward propagation moves data forward through layers, while backpropagation reduces error using gradient descent.

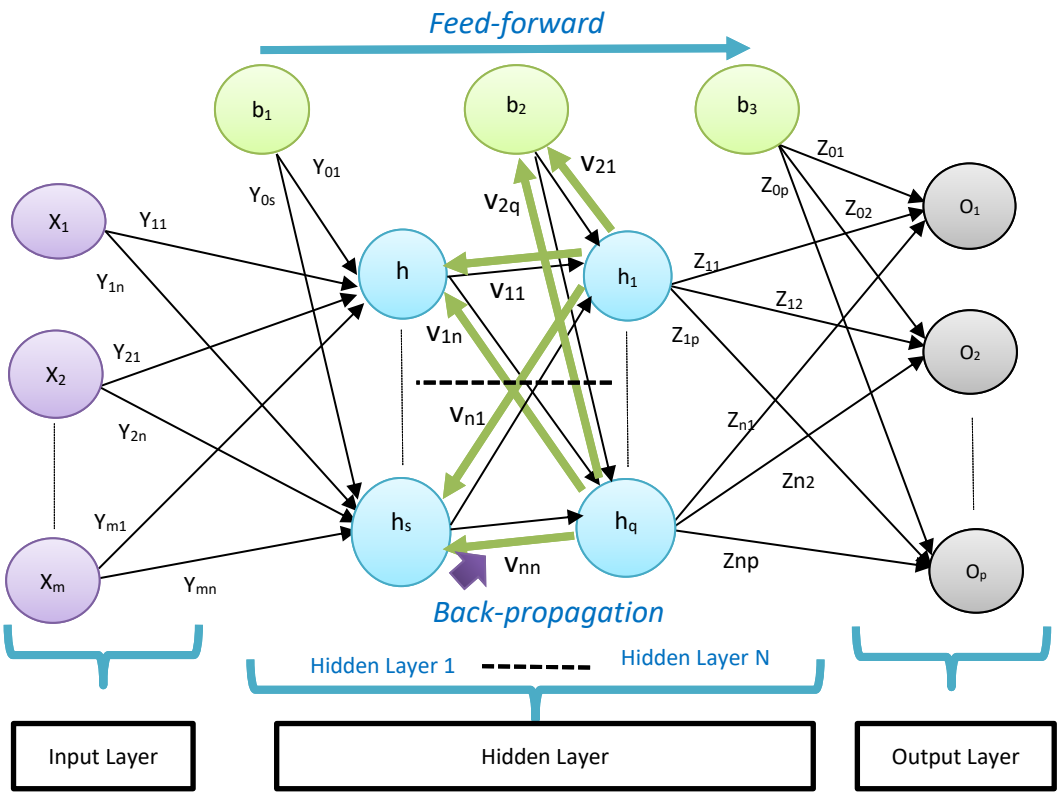

Figure 3.2.1 Diagram of the Feedforward-backpropagation neural network (FBNN) algorithm.

The following steps are required for the proposed model:

Step 1: Based on the target/output class, process the input data according to their values.
Step 2: Assign random weights to every neuron in the network's output and hidden layers.
Step 3: Provide an input unit xi, $1 \leq i \leq m$, to every input neuron that transfers the information to the intermediate layer.
Step 4: In the hidden layer, every hidden neuron h_j, $1 \leq j \leq s$, adds up its weighted input signals.

$$h_{inj} = Y_{0j} + m \sum_{i=1}^{m} xi \times Yij \quad (1)$$

Proceed to step five: calculate the output signal and distribute it to every unit within the hidden layer.

$$h_j = F(h_{inj}), \text{ where } F \text{ is a activation function in hidden layer} \quad (2)$$

Step 6: Send unit h_j, $1 \leq j \leq s$, to every input neuron so that it can be forwarded to the following hidden layer. In N hidden layers, each output neuron h_t, $1 \leq t \leq q$, adds up its weighted input signals.

$$h_{jnt} = V_{2t} + s \sum_{j=1}^{s} hj \times Vjt \quad (3)$$

Step 7: Calculate the output signal and distribute it to each of the N hidden layer units.

$$h_t = F(h_{jnt}), \text{ where } F \text{ is a activation function in N hidden} \quad (4)$$

Step 8. Input unit ht, $1 \leq t \leq q$ to each input neuron that passes it to the Output Layer Each output neuron O_k, $1 \leq k \leq p$ sums its weighted input signals in output layer through

$$O_{tnk} = Z_{0k} + s \sum_{t=1}^{q} ht \times Ztk \quad (5)$$

Step 9. Send the output signal to each unit in the output layer after computing,

$$O_k = F(O_{tnk}), \text{ where } F \text{ is activation function in Output layer} \quad (6)$$

Step 10. Every output unit is trained using the 'categorical_crossentropy' loss function of the optimizer after receiving a target pattern that matches the input training pattern. The optimizer calculates the error during training and reverse-engineers the network's weights and biases to minimize this error.

$$\beta_k = (E_k - O_k) \times F'(O_{tnk}) \quad (7)$$

Step 11. Calculate previous hidden layer weights updating term

$$\triangle Z_{tk} = \alpha \times \beta_k \times h_t \quad (8)$$

Where, α = learning rate.
Step 12: Determine the bias transformation term.

$$\triangle Z_{0k} = \alpha \times \beta_k \quad (9)$$

and returns β_k to the layer's units.

Step 13. Sum the bita inputs of each hidden neuron of h_t

$$\beta_{jnt} = \sum_{k=1}^{p} \beta k \times Ztk \quad (10)$$

Step 14. Compute minimizing error using optimizer and loss function

$$\beta_t = \beta_{jnt} \times F'(h_{jnt}) \quad (11)$$

Step 15. Calculate previous layer weight correction term

$$\triangle V_{jt} = \alpha \times \beta_t \times h_j \quad (12)$$

Where, α = learning rate.

Step 16. Calculate its bias correction term

$$\triangle V_{0t} = \alpha \times \beta_t \quad (13)$$

and returns β_t to the layer's units.

Step 17. Sum the bita inputs of each hidden neuron of h_j

$$\beta_{inj} = \sum_{k=1}^{p} \beta t \times Vjt \quad (14)$$

Step 18. Compute again minimizing error using optimizer and loss function

$$\beta_j = \beta_{inj} \times F'(h_{inj}) \quad (15)$$

Step 19. Calculate previous layer weight updating term

$$\triangle Y_{ij} = \alpha \times \beta_j \times x_i \quad (16)$$

Where, α = learning rate.

Step 20. Calculate its bias correction term

$$\triangle Y_{0j} = \alpha \times \beta_j \quad (17)$$

Step 21. upgrade each weight of the network through

$$Z_{tk}(new) = Z_{tk}(old) + \triangle Z_{tk} \quad (18)$$

$$V_{jt}(new) = V_{jt}(old) + \triangle V_{jt} \quad (19)$$

$$Y_{ij}(new) = Y_{ij}(old) + \triangle Y_{ij} \quad (20)$$

Step 22. To get the weights to converge, repeat steps 4–21.

3.2.4 Process of experiments

In order to develop a system (fig 3.2.3), the network was trained using different hidden layers and activation function like table below (table 3.2.4). The hidden layer used activation functions like tanh, sigmoid, ReLU, Parametric ReLU, while the output layer used tanh or sigmoid/logistic functions, with optimizers such as Adam, Stochastic Gradient Descent (SGD), RMSProp, or Nadam. The study tested three models to identify the best evaluation parameters: Model 1 (ReLU, sigmoid), Model 2 (ReLU, tanh),), and Model 3 (PReLU, sigmoid).

Model 1. Using Eq. (21) in hidden layers and Eq. (24) in Output Layer.

Model 2. Using Eq. (21) in hidden layers and Eq. (23) in Output Layer.

Model 3. Using Eq. (22) in hidden layers and Eq. (24) in Output Layer.

ReLU Function stands for Rectified Linear Unit [20].

$$F(x) = x^{+} = \max(0,x) = \begin{cases} 0 & if\ x \le 0 \\ x & if\ x > 0 \end{cases} \quad (21)$$

PReLU, or parametric rectified linear unit: To address the issue of the gradient becoming zero for the left half of the axis, ReLU is an additional variation of ReLU [21].

$$F(x) = \begin{cases} \alpha x & if\ x < 0 \\ x & if\ x \ge 0 \end{cases} \text{ with parameter } \alpha \quad (22)$$

The Tanh function, also known as the Hyperbolic Tangent, bears a striking resemblance to the sigmoid/logistic activation function. In fact, they even share the same S-shape, although their output ranges differ by -1 to 1. In Tanh, the output value will be closer to 1.0 the larger the input (more positive), and the closer to -1.0 the smaller the input (more negative) [22].

$$F(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad x \in R \quad (23)$$

The sigmoid activation function produces values between 0 and 1 when given any real value as input [23].

$$F(x) = \frac{1}{1+e^{-x}} \quad x \in R \quad (24)$$

Table 3.1: mathematical functions of models

Eq.	Activation function name	Function $F(x)$	Range
21	ReLU (Rectified linear unit)	$x^{+} = \max(0,x) = \begin{cases} 0 & if\ x \le 0 \\ x & if\ x > 0 \end{cases}$	$(0,\infty)$
22	PReLU	$F(x) = \begin{cases} \alpha x & if\ x < 0 \\ x & if\ x \ge 0 \end{cases}$ with parameter α	$(-\infty, \infty)$
23	Hyperbolic tangent (tanh)	$F(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$ $x \in R$	(-1,1)
24	Sigmoid	$F(x) = \frac{1}{1+e^{-x}}$ $x \in R$	(0,1)

The three models mentioned were tested on the trained system to assess accuracy at each layer. The mean squared error (MSE) Eq. (25) [24] was calculated, displaying the cumulative squared error between predicted and actual values. False positives (FP) and false negatives (FN) were also computed. The model also records the minimum CPU time for completion. Where N is the number of data points, O_i and O'_i represents observed and predicted values, respectively. The MSE, type I, type II and CPU times of the performance case is plotted to diagnose the system efficiently. Also, different performance metrics have used as a key-indicators including accuracy (AC) [25], precision [26], recall, ROC [26], F1-score [27], and the confusion matrix (CM) [27], which involves true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values.

$$\text{Mean Square Error (MSE)} = \frac{\sum_{i=1}^{n}(O_i - O'_i)^2}{N} \quad (25)$$

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+FP+FN+TN)} \quad (26)$$

$$\text{Precision} = \frac{TP}{(TP+FP)} \quad (27)$$

$$\text{Recall} = \frac{TP}{(TP+FN)} \quad (28)$$

$$\text{F1 score} = \frac{2\times(\text{Precision}\times\text{Recall})}{(\text{Precision}+\text{Recall})} \quad (29)$$

3.3 Data Flow Diagram

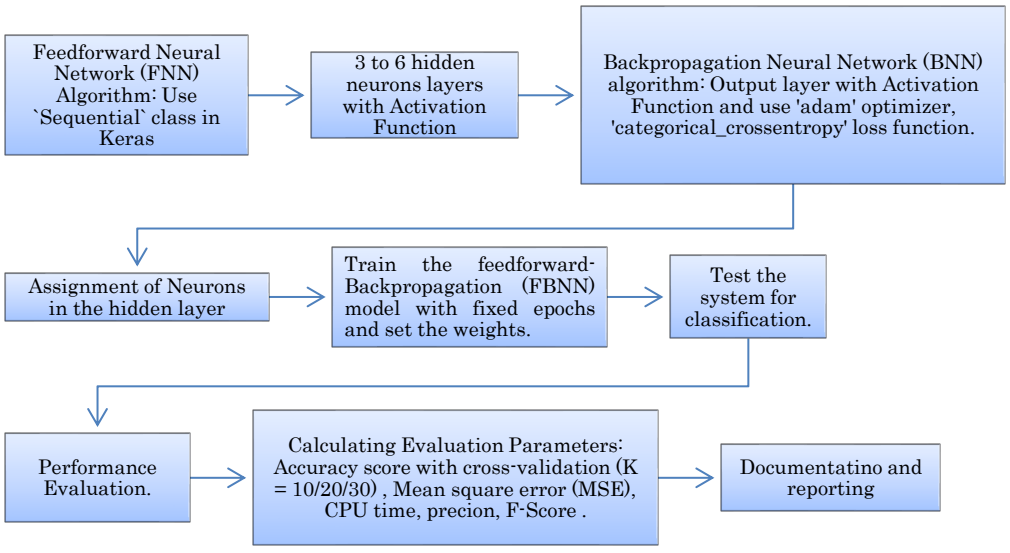

Figure 3.3 Data flow diagram

3.4 Summary

This proposed methodology uses Feedforward Neural Networks (FBNNs) to develop a system for identifying fetal health conditions from clinical (CTG) data. The first step involves preparing the dataset, that is, separating the independent features from the target variable, encoding the feature labels by using some labeling system in numerical form, normalizing using StandardScaler, and finally splitting the dataset into training, validation, and test datasets. These preparatory steps ensure clean and ready data for the training of the deep learning model. The FBNN model comprises input, hidden, and output layers using Keras's (Sequential) class for building it up. The input layer passes the data to the hidden layers, where each neuron processes it with weights and activation functions like ReLU, PReLU, tanh, or sigmoid. The output layer collects those signals to classify fetal health into Normal, Suspect, or Pathological.

The model is trained through backpropagation, which tunes the weights and biases according to errors found while predicting. The compile method establishes the optimizer (for example, Adam, SGD) and loss function (categorical_crossentropy) entitled for this process. The methodology explains conducting trials with three models all using different configurations of activation functions and optimizers to further enhance the system.

Chapter 4

Implementation and Results

This chapter details the practical execution of the proposed methodology, including the environment setup, testing, and evaluation of the models. It presents the results, comparative performance analysis, and a discussion of the findings to highlight the model's effectiveness.

4.1 Environment Setup

Hardware Configuration:

- Processor: AMD Ryzen 3 PRO 2300U (4 cores).
- Memory: 8 GB RAM.
- Graphics: Radeon Vega 6 integrated graphics for efficient model training and evaluation.

Software Configuration:

- Programming Language: Python.
- Libraries: TensorFlow, Keras, scikit-learn, Pandas, NumPy, Matplotlib, Seaborn, SciPy, imbalanced-learn, OpenCV, PyTorch, Plotly, SHAP, and LIME.
- Development Environment: Google Colab and Jupyter Notebook for conducting experiments and managing workflows.

Experimentation Techniques:

- Activation functions such as ReLU, Leaky ReLU, PReLU, tanh, and sigmoid were tested in different configurations.
- Optimizers including Adam, Stochastic Gradient Descent (SGD), RMSProp, and Nadam were used to refine model performance.

4.2 Results and Discussion

The study tested three different models with varying activation functions, loss functions, and configurations of hidden layers to evaluate their performance in detecting fetal health

conditions. The models used in the research are as follows:

- **Model 1:** Uses ReLU activation in hidden layers and Sigmoid activation in the output layer with the loss function *categorical_crossentropy*.
- **Model 2:** Uses ReLU activation in hidden layers and Tanh activation in the output layer with the loss function *categorical_crossentropy*.
- **Model 3**: Uses PReLU activation in hidden layers and Sigmoid activation in the output layer with the loss function *categorical_crossentropy*.

The following tables basically summarizes the results for each model based on the number of hidden layers (3,4,5,6).

Training of twelve combinations of neural network with Model 1

Table 4.2.1: Attainment of Evaluation Matrices of 3 Hidden Layers for Model 1

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(8, 9, 6)	56	0.922715	0.909091	0.877743	20.14066	0.092729	0.815127	0.843055	0.827607
(7, 8, 14)	78	0.94086	**0.918495**	0.865204	20.56036	**0.112627**	0.831383	0.85174	0.839804
(10, 8, 5)	58	0.930108	0.909091	0.880878	16.60412	0.065845	0.822483	0.859233	0.838196
(15, 12, 10)	4	0.852151	0.846395	0.846395	4.057266	0.124354	0.740321	0.605926	0.653141
(10, 10, 12)	67	0.9375	0.909091	0.887147	15.60683	0.094263	0.815624	0.838469	0.823908
(14, 15, 8)	13	0.900538	0.899687	0.871473	5.60611	0.086102	0.793392	0.841376	0.810062
(14, 7, 9)	12	0.890457	0.884013	0.887147	4.019419	0.109086	0.764162	0.804609	0.77492
(13, 9, 5)	41	0.913978	0.880878	0.880878	8.216318	0.103677	0.862907	0.728856	0.751793
(5, 8, 9)	54	0.916667	**0.918495**	0.884013	10.30692	**0.207232**	0.848187	0.860913	0.854369
(13, 7, 13)	29	0.915323	0.896552	0.887147	7.586107	0.063418	0.78549	0.849182	0.812407
(8, 10, 13)	2	0.796371	0.793103	0.811912	2.454556	0.136095	0.47566	0.391491	0.388753
(12, 10, 6)	74	0.954301	0.899687	0.890282	13.32482	0.072333	0.832418	0.809021	0.819579

Table 4.2.2: Attainment of Evaluation Matrices of 4 Hidden Layers for Model 1

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(13, 9, 9, 5)	66	0.938844	0.902821	0.874608	29.19811	0.081955	0.796877	0.835737	0.81256
(10, 13, 5, 15)	44	0.930108	**0.915361**	0.884013	18.7563	**0.24823**	0.824958	0.866552	0.841785
(11, 8, 12, 6)	28	0.912634	0.896552	0.880878	12.17787	0.280809	0.851766	0.78689	0.816179
(11, 13, 15, 11)	44	0.93078	0.909091	0.896552	15.70114	0.08915	0.838866	0.838469	0.838137
(8, 10, 15, 12)	7	0.844086	0.815047	0.833856	8.697613	0.113498	0.714683	0.578757	0.615847
(10, 8, 5, 12)	13	0.893145	0.871473	0.890282	12.28788	0.078297	0.722132	0.73902	0.72834
(9, 10, 12, 9)	75	0.936156	0.896552	0.890282	31.1363	0.110505	0.794253	0.805235	0.799586
(14, 10, 10, 11)	8	0.864919	0.840125	0.862069	8.415501	0.084628	0.828251	0.682418	0.64482
(8, 12, 8, 7)	81	0.946237	0.905956	0.884013	23.50004	0.126711	0.808268	0.809334	0.808448

(7, 13, 13, 7)	37	0.909274	0.899687	0.877743	16.15342	0.084477	0.803691	0.788257	0.794341
(15, 8, 11, 15)	46	0.946909	0.890282	0.868339	16.06299	0.066515	0.768667	0.816514	0.785963
(10, 11, 12, 11)	11	0.894489	0.887147	0.880878	4.764108	0.099185	0.814884	0.74318	0.761361

Table 4.2.3: Attainment of Evaluation Matrices of 5 Hidden Layers for Model 1

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	68	0.954973	0.902821	0.868339	13.40673	0.055727	0.796306	0.821978	0.804318
(15, 7, 11, 9, 7)	74	0.962366	0.905956	0.896552	14.07601	0.094737	0.820905	0.82793	0.821535
(7, 12, 9, 10, 12)	3	0.823925	0.862069	0.824451	3.354128	0.10258	0.53947	0.549961	0.54376
(14, 13, 10, 5, 7)	**80**	0.977151	**0.940439**	0.874608	14.43874	**0.039628**	0.850256	0.872643	0.860817
(14, 14, 13, 11, 15)	1	0.830645	0.799373	0.827586	3.952929	0.216007	0.615093	0.578932	0.580545
(6, 14, 15, 6, 11)	15	0.892473	0.884013	0.865204	4.973547	0.069102	0.745133	0.77684	0.75446
(5, 15, 12, 8, 15)	13	0.877688	0.880878	0.855799	5.403526	0.080954	0.742341	0.756877	0.749087
(6, 15, 7, 9, 14)	81	0.923387	0.871473	0.874608	14.67343	0.11529	0.733936	0.785134	0.755467
(8, 13, 7, 12, 15)	27	0.90121	0.874608	0.887147	7.902645	0.063688	0.785426	0.74472	0.75485
(6, 13, 11, 7, 8)	46	0.924731	0.924765	0.893417	10.88882	0.083377	0.850702	0.831289	0.840711
(12, 8, 15, 7, 14)	30	0.915323	0.887147	0.868339	6.897243	0.09042	0.775248	0.766614	0.770004
(6, 9, 12, 15, 11)	60	0.927419	0.871473	0.887147	12.66398	0.218304	0.760443	0.810484	0.783218

Table 4.2.4: Attainment of Evaluation Matrices of 6 Hidden Layers for Model 1

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(6, 7, 11, 7, 7, 7)	61	0.922715	0.880878	0.893417	11.92987	0.083196	0.78486	0.798405	0.791172
(5, 6, 13, 8, 13, 14)	3	0.778898	0.76489	0.789969	6.791885	0.121848	0.254963	0.333333	0.288928
(7, 7, 5, 13, 10, 11)	3	0.778898	0.76489	0.789969	4.532917	0.324209	0.254963	0.333333	0.288928
(9, 6, 12, 14, 14, 11)	88	0.935484	0.877743	0.877743	21.04739	0.1173	0.749286	0.781113	0.761323
(15, 8, 6, 5, 10, 8)	14	0.895833	0.890282	0.865204	5.71116	0.267818	0.760427	0.809508	0.779114
(6, 8, 7, 7, 14, 13)	77	0.936828	0.899687	0.880878	15.08378	0.20804	0.79364	0.811188	0.801807
(12, 15, 10, 11, 9, 5)	58	0.94086	0.905956	0.874608	12.93091	0.071928	0.798407	0.807166	0.796621
(11, 14, 11, 5, 9, 9)	48	0.941532	0.902821	0.905956	11.21864	0.294723	0.797151	0.819559	0.806857
(6, 15, 10, 15, 14, 8)	74	0.940188	0.899687	0.893417	14.54027	0.240041	0.790781	0.804183	0.79731
(14, 8, 5, 10, 9, 11)	81	0.965054	0.874608	0.874608	15.69393	0.071999	0.733402	0.784333	0.753048
(14, 5, 9, 15, 10, 14)	26	0.917339	**0.918495**	0.865204	6.420511	**0.084989**	0.825386	0.854159	0.83041
(15, 6, 15, 11, 14, 11)	15	0.903226	0.862069	0.884013	7.514815	0.06991	0.703119	0.760523	0.72118

And the followings are the diagrams for 'Accuracy' and 'CPU times on Model 1

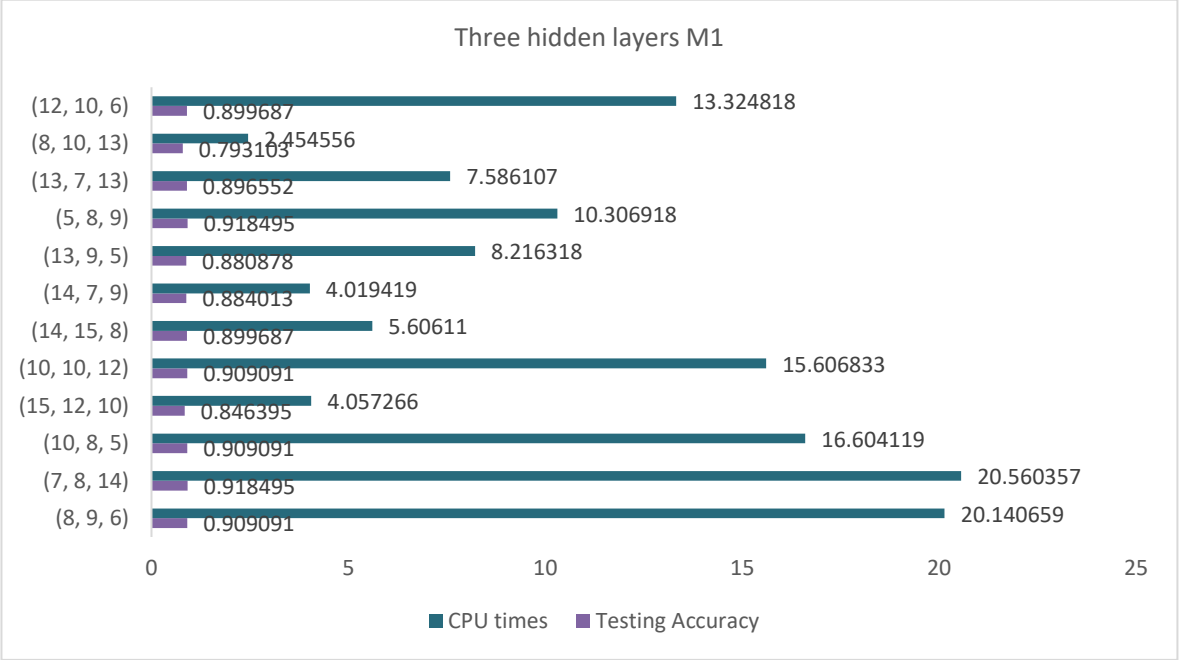

Figure 4.2.1 Accuracy and CPU times of 3 hidden layer for Model 1

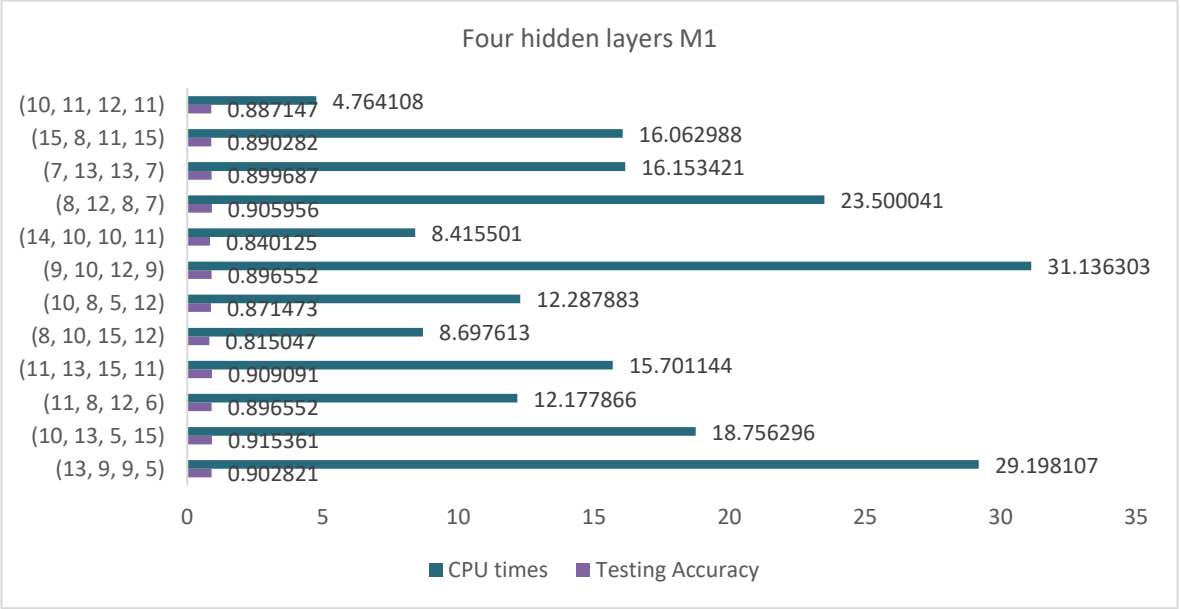

Figure 4.2.2 Accuracy and CPU times of 4 hidden layer for Model 1

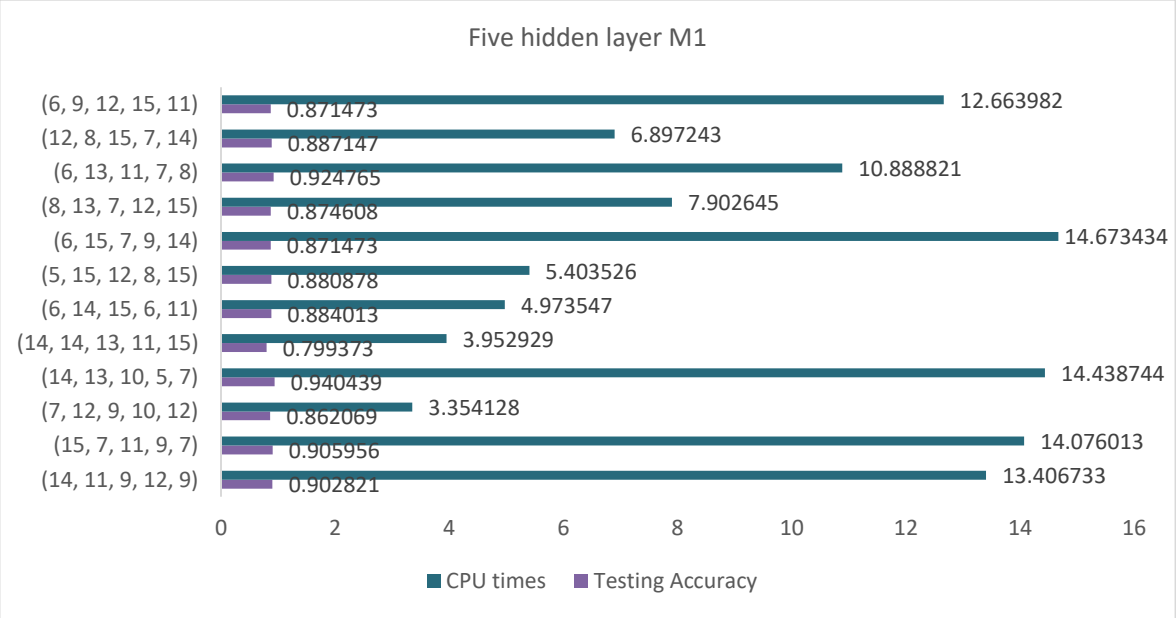

Figure 4.2.3 Accuracy and CPU times of 5 hidden layer for Model 1

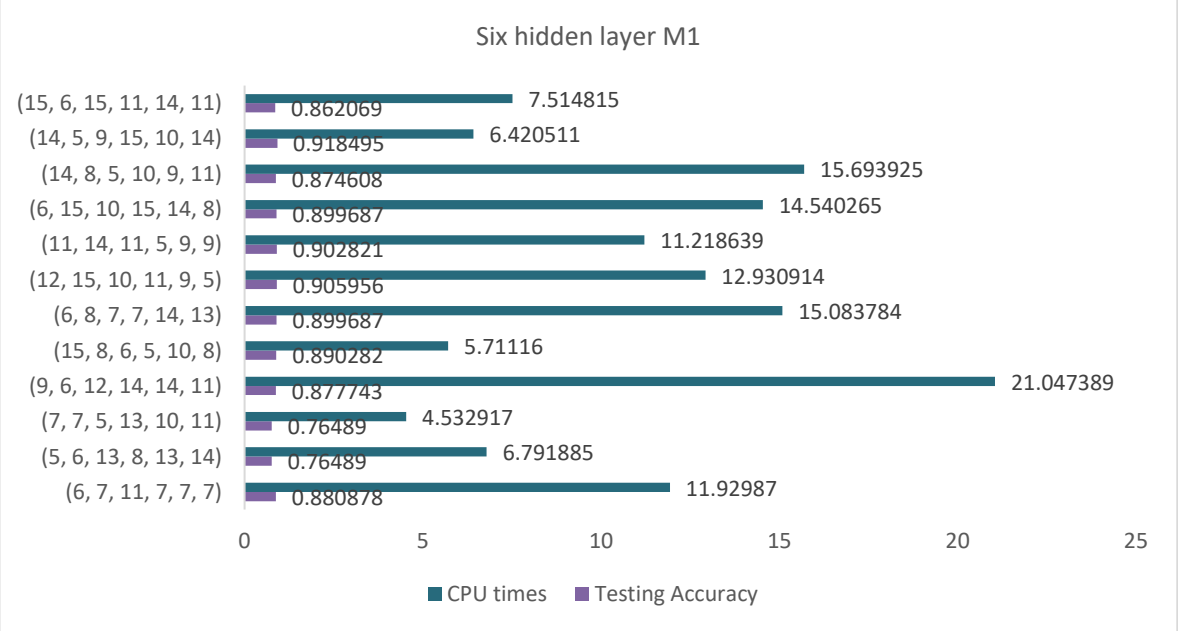

Figure 4.2.4 Accuracy and CPU times of 6 hidden layer for Model 1

These four above tables and figures basically shows the experimental outcome of Model 1 employing ReLU and Sigmoid activation functions of neural networks, along with Adam optimizer with 3,4,5,6 hidden layers sequentially. These tables and figures show distinct combination of neurons. It is clearly visible that the best performance is of 94% Testing Accuracy and the least of 0.039 MSE in the combinations of neurons of Model 1 on table 4.3.

Training of twelve combinations of neural network with Model 2.

Table 4.2.5: Attainment of Evaluation Matrices of 3 Hidden Layers for Model 2

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(8, 9, 6)	56	0.088038	0.065831	0.07837	21.48735	1.905956	0.07483	0.30371	0.046159
(7, 8, 14)	78	0.778898	**0.76489**	0.789969	23.29413	**0.413793**	0.254963	0.333333	0.288928
(10, 8, 5)	58	0.760081	0.727273	0.758621	20.52027	0.413793	0.255226	0.31694	0.282754
(15, 12, 10)	4	0.186828	0.222571	0.219436	7.395725	0.420063	0.306137	0.294204	0.155456
(10, 10, 12)	67	0.44422	0.451411	0.423197	25.35533	0.501567	0.317726	0.455565	0.273571
(14, 15, 8)	13	0.042339	0.040752	0.025078	6.343703	0.677116	0.030592	0.204887	0.038629
(14, 7, 9)	12	0.042339	0.040752	0.034483	8.007153	0.529781	0.030273	0.181704	0.051381
(13, 9, 5)	41	0.778898	**0.76489**	0.789969	14.31129	**0.413793**	0.254963	0.333333	0.288928
(5, 8, 9)	54	0.658602	0.611285	0.664577	20.64406	0.438871	0.288598	0.36346	0.301274
(13, 7, 13)	29	0.186828	0.203762	0.175549	10.58368	0.429467	0.147012	0.526003	0.219268
(8, 10, 13)	2	0.107527	0.084639	0.084639	3.099244	1.141066	0.287163	0.344262	0.059615
(12, 10, 6)	74	0.778898	0.76489	0.789969	13.76352	0.413793	0.254963	0.333333	0.288928

Table 4.2.6: Attainment of Evaluation Matrices of 4 Hidden Layers for Model 2

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(13, 9, 9, 5)	66	**0.778898**	0.76489	0.789969	13.24352	**0.413793**	0.254963	0.333333	0.288928
(10, 13, 5, 15)	44	**0.778898**	0.76489	0.789969	10.188	**0.413793**	0.254963	0.333333	0.288928
(11, 8, 12, 6)	28	0.391801	0.442006	0.39185	7.710143	0.532915	0.40279	0.475307	0.358455
(11, 13, 15, 11)	44	0.130376	0.175549	0.141066	9.719452	0.824451	0.058516	0.333333	0.099556
(8, 10, 15, 12)	7	0.040995	0.059561	0.028213	3.567234	0.394984	0.37224	0.178058	0.063183
(10, 8, 5, 12)	13	0.066532	0.065831	0.059561	6.1372	0.442006	0.044099	0.264098	0.07325
(9, 10, 12, 9)	75	0.775538	0.755486	0.793103	13.9787	0.413793	0.348757	0.393946	0.366753
(14, 10, 10, 11)	8	0.096102	0.0721	0.094044	3.781881	0.846395	0.099499	0.345551	0.081145
(8, 12, 8, 7)	81	0.778898	0.76489	0.789969	17.00639	0.413793	0.254963	0.333333	0.288928
(7, 13, 13, 7)	37	0.453629	0.416928	0.520376	10.0004	0.61442	0.255113	0.343471	0.236182
(15, 8, 11, 15)	46	0.092742	0.068966	0.0721	10.4662	1.62069	0.104975	0.316416	0.084495
(10, 11, 12, 11)	11	0.055108	0.062696	0.050157	4.069828	0.479624	0.041706	0.258145	0.069113

Table 4.2.7: Attainment of Evaluation Matrices of 5 Hidden Layers for Model 2

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	68	0.778898	0.76489	0.789969	13.46638	0.413793	0.254963	0.333333	0.288928
(15, 7, 11, 9, 7)	74	0.778898	0.76489	0.789969	14.30334	0.413793	0.254963	0.333333	0.288928
(7, 12, 9, 10, 12)	3	0.666667	0.68652	0.652038	5.214896	0.727273	0.324393	0.460958	0.362488
(14, 13, 10, 5, 7)	80	0.778898	0.76489	0.789969	15.60771	0.413793	0.254963	0.333333	0.288928
(14, 14, 13, 11, 15)	1	0.778898	0.76489	0.789969	4.006498	0.413793	0.254963	0.333333	0.28892
(6, 14, 15, 6, 11)	15	0.770161	0.758621	0.777429	5.471485	0.39185	0.2569	0.330601	0.289128
(5, 15, 12, 8, 15)	13	0.778898	0.76489	0.789969	4.719255	0.413793	0.254963	0.333333	0.288928
(6, 15, 7, 9, 14)	81	0.775538	0.761755	0.783699	15.53113	0.426332	0.254717	0.331967	0.288256
(8, 13, 7, 12, 15)	27	0.710349	0.702194	0.724138	8.065752	0.664577	0.30575	0.3869	0.339574
(6, 13, 11, 7, 8)	46	0.778898	0.76489	0.789969	10.8101	0.413793	0.254963	0.333333	0.288928

(12, 8, 15, 7, 14)	30	0.834005	**0.84326**	0.865204	6.944228	**0.213166**	0.690567	0.579461	0.605085
(6, 9, 12, 15, 11)	60	0.778898	0.76489	0.789969	13.00701	0.413793	0.254963	0.333333	0.288928

Table 4.2.8: Attainment of Evaluation Matrices of 6 Hidden Layers for Model 2

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(6, 7, 11, 7, 7, 7)	61	0.778898	0.76489	0.789969	16.90758	0.413793	0.254963	0.333333	0.288928
(5, 6, 13, 8, 13, 14)	3	0.778898	0.76489	0.789969	4.377946	0.413793	0.254963	0.333333	0.288928
(7, 7, 5, 13, 10, 11)	3	0.778898	0.76489	0.789969	3.557172	0.413793	0.254963	0.333333	0.288928
(9, 6, 12, 14, 14, 11)	88	0.778898	0.76489	0.789969	23.59283	0.413793	0.254963	0.333333	0.288928
(15, 8, 6, 5, 10, 8)	14	0.778898	0.76489	0.789969	9.584885	0.413793	0.254963	0.333333	0.288928
(6, 8, 7, 7, 14, 13)	77	0.778898	0.76489	0.789969	31.42194	0.413793	0.254963	0.333333	0.288928
(12, 15, 10, 11, 9, 5)	58	0.778898	0.76489	0.789969	23.21081	0.413793	0.254963	0.333333	0.288928
(11, 14, 11, 5, 9, 9)	48	0.778898	0.76489	0.789969	16.77217	0.413793	0.254963	0.333333	0.288928
6, 15, 10, 15, 14, 8)	74	0.778898	0.76489	0.789969	15.28497	0.413793	0.254963	0.333333	0.288928
(14, 8, 5, 10, 9, 11)	81	0.776882	**0.768025**	0.783699	16.07567	**0.401254**	0.423239	0.350877	0.321704
(14, 5, 9, 15, 10, 14)	26	0.778898	0.733542	0.789969	8.782144	0.529781	0.354592	0.432916	0.389816
(15, 6, 15, 11, 14, 11)	15	0.778898	0.76489	0.789969	5.00309	0.413793	0.254963	0.333333	0.288928

And the followings are the diagrams for 'Accuracy' and 'CPU times' on Model 2.

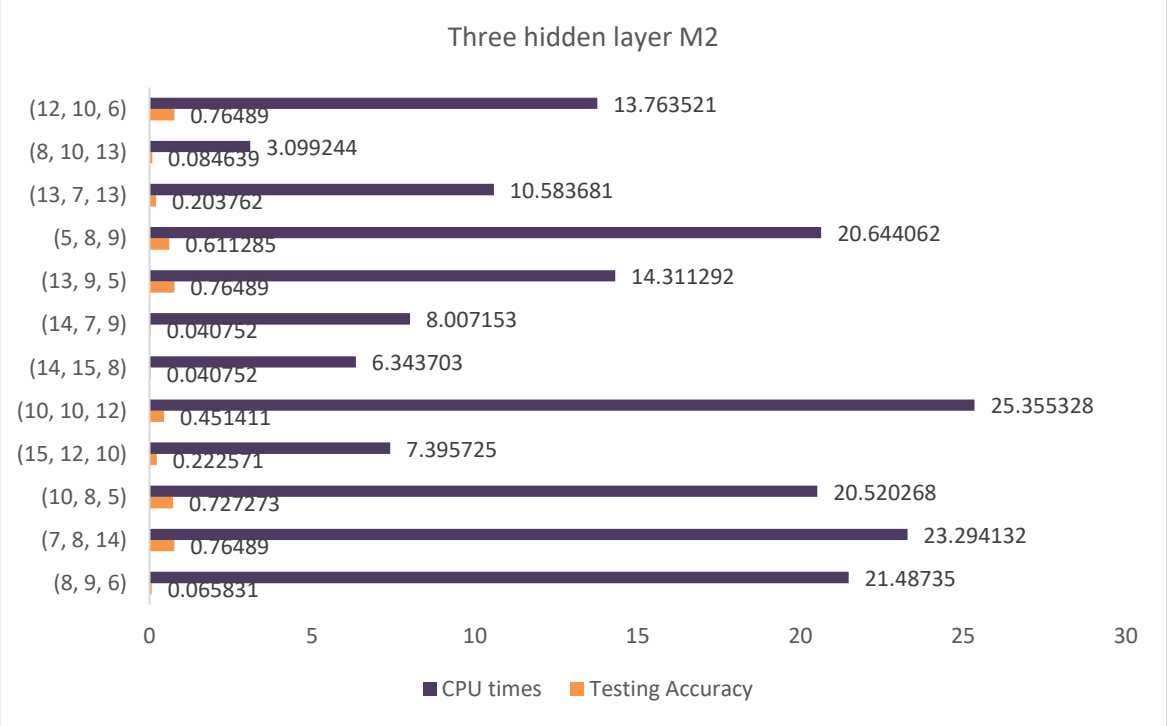

Figure 4.2.5 Accuracy and CPU times of 3 hidden layer for Model 2

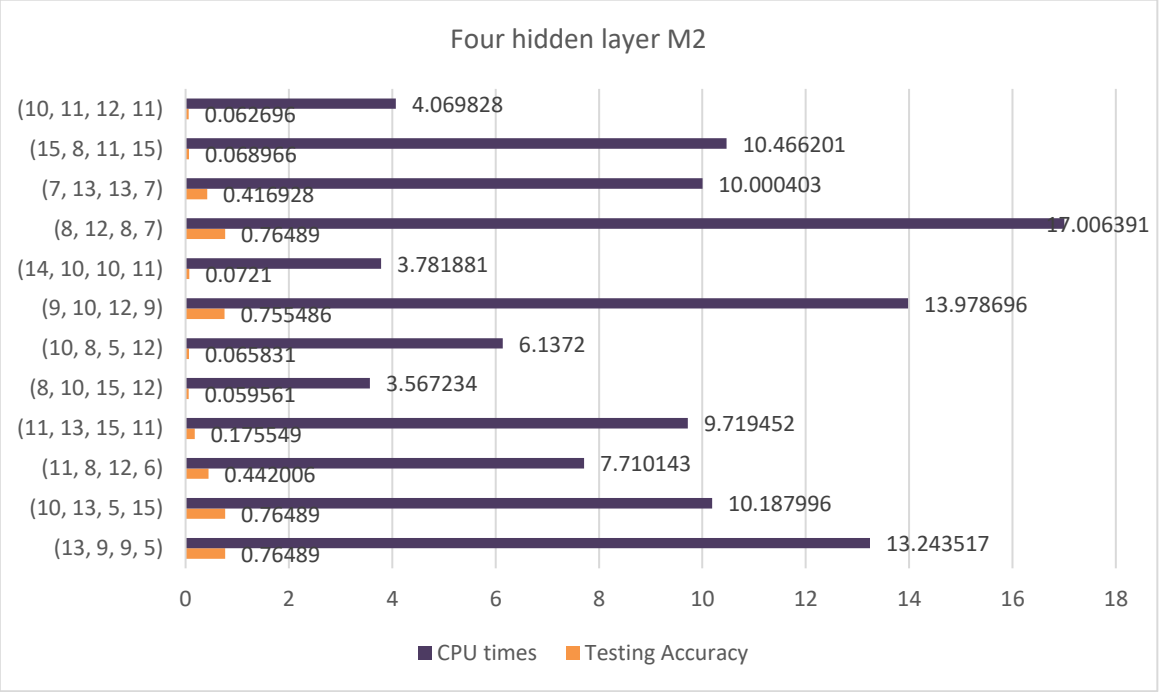

Figure 4.2.6 Accuracy and CPU times of 4 hidden layer for Model 2

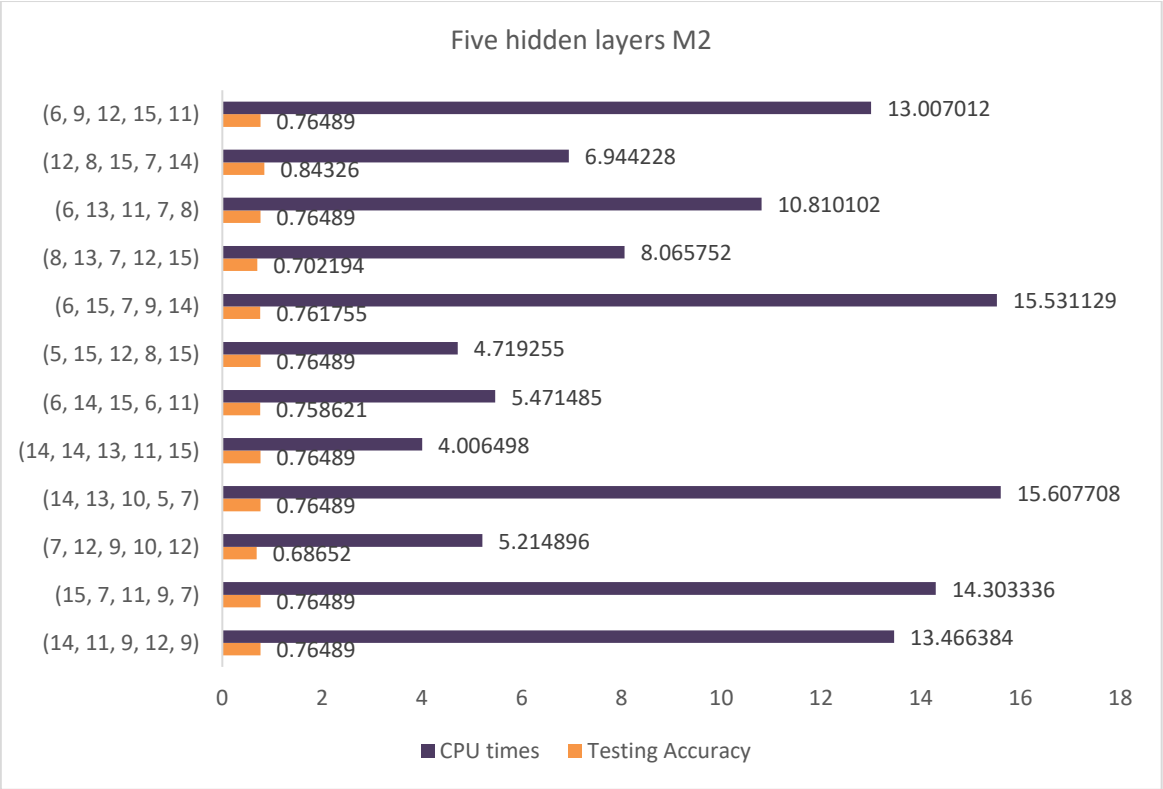

Figure 4.2.7 Accuracy and CPU times of 5 hidden layer for Model 2

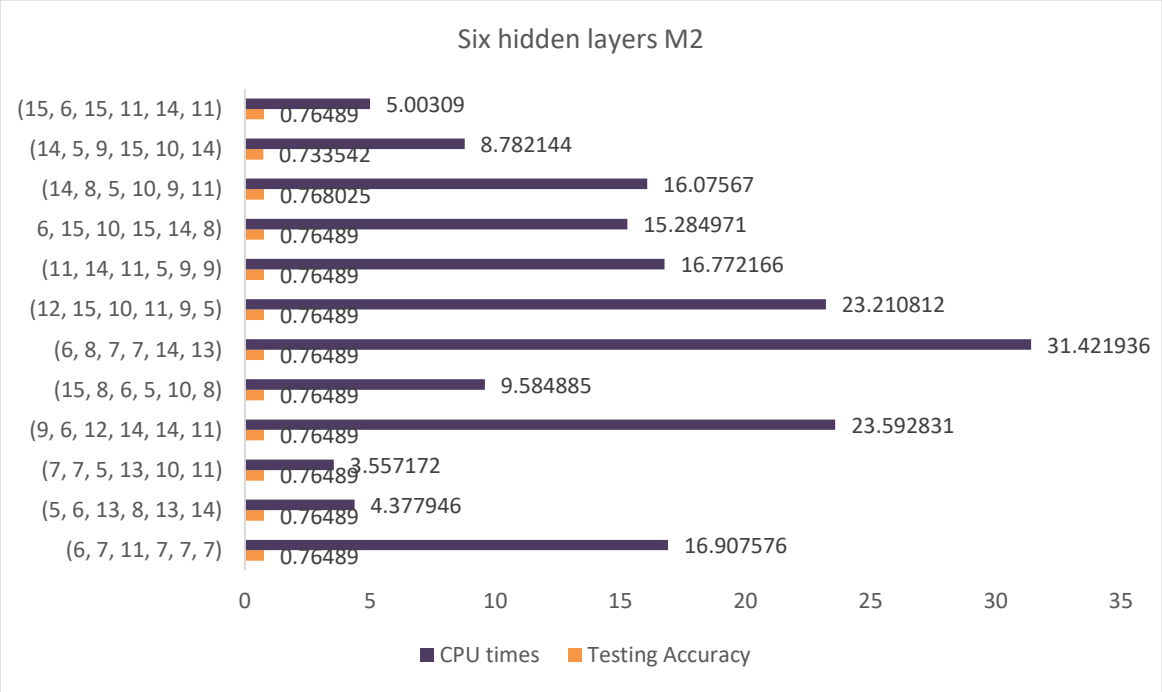

Figure 4.2.8 Accuracy and CPU times of 6 hidden layer for Model 2

These four above tables and figures basically shows the experimental outcome of Model 2 employing ReLU and Tanh activation functions of neural networks, along with Adam optimizer with 3,4,5,6 hidden layers sequentially. These tables and figures show distinct combination of neurons. It is clearly visible that the best performance is of 84% Testing Accuracy and the least of 0.21 MSE in the combinations of neurons with five number of hidden layers of Model 2 on table 4.2.7.

Training of twelve combinations of neural network with Model 3.

Table 4.2.9: Attainment of Evaluation Matrices of 3 Hidden Layers for Model 3

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(8, 9, 6)	56	0.886425	0.887147	0.893417	20.53079	0.141066	0.897056	0.690563	0.761307
(7, 8, 14)	78	0.897177	0.887147	0.899687	26.95337	0.141066	0.862804	0.692982	0.75354
(10, 8, 5)	58	0.919355	0.874608	0.896552	19.97929	0.153605	0.828781	0.731465	0.767564
(15, 12, 10)	4	0.8125	0.786834	0.833856	3.897198	0.297806	0.501299	0.449135	0.438257
(10, 10, 12)	67	0.87164	0.887147	0.887147	13.57775	0.141066	0.857574	0.662794	0.726197
(14, 15, 8)	13	0.870968	0.877743	0.874608	6.208552	0.197492	0.863894	0.677293	0.74245
(14, 7, 9)	12	0.826613	0.802508	0.84326	4.243236	0.263323	0.732279	0.497494	0.530067
(13, 9, 5)	41	0.894489	**0.893417**	0.887147	9.634407	**0.115987**	0.889588	0.690877	0.75567
(5, 8, 9)	54	0.84543	0.840125	0.865204	9.089683	0.206897	0.797116	0.550326	0.611458
(13, 7, 13)	29	0.842742	0.865204	0.874608	8.722325	0.163009	0.856736	0.567756	0.607402
(8, 10, 13)	2	0.801747	0.777429	0.818182	3.27179	0.363636	0.536564	0.419687	0.424177
(12, 10, 6)	74	0.877016	0.868339	0.890282	13.86783	0.159875	0.858452	0.610651	0.661445

Table 4.2.10: Attainment of Evaluation Matrices of 4 Hidden Layers for Model 3

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(13, 9, 9, 5)	66	0.883065	0.830721	0.830721	13.81834	0.216301	0.859151	0.629787	0.664497
(10, 13, 5, 15)	44	0.889113	0.877743	0.893417	11.90621	0.159875	0.843984	0.658696	0.719077
(11, 8, 12, 6)	28	0.90121	**0.905956**	0.899687	9.394792	**0.112853**	0.868969	0.765387	0.804404
(11, 13, 15, 11)	44	0.88172	0.890282	0.905956	18.78203	0.137931	0.904283	0.675752	0.749389
(8, 10, 15, 12)	7	0.859543	0.874608	0.890282	6.5123	0.163009	0.842756	0.627393	0.691859
(10, 8, 5, 12)	13	0.821909	0.84326	0.84953	5.841311	0.194357	0.498768	0.486729	0.487811
(9, 10, 12, 9)	75	0.824597	0.846395	0.836991	15.23806	0.181818	0.499091	0.492681	0.492139
(14, 10, 10, 11)	8	0.880376	0.871473	0.893417	8.118454	0.166144	0.873256	0.679147	0.745966
(8, 12, 8, 7)	81	0.862903	0.858934	0.880878	15.49371	0.169279	0.863393	0.593045	0.664162
(7, 13, 13, 7)	37	0.860887	0.84953	0.871473	8.572435	0.188088	0.753539	0.574685	0.599477
(15, 8, 11, 15)	46	0.897849	0.868339	0.890282	14.55242	0.159875	0.8616	0.601478	0.6567
(10, 11, 12, 11)	11	0.879704	0.893417	0.887147	6.909053	0.144201	0.89201	0.676866	0.734686

Table 4.2.11: Attainment of Evaluation Matrices of 5 Hidden Layers for Model 3

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	68	0.928091	0.896552	0.899687	15.27708	0.131661	0.897899	0.759373	0.81295
(15, 7, 11, 9, 7)	74	0.889785	0.887147	0.884013	16.24331	0.131661	0.903416	0.727757	0.785743
(7, 12, 9, 10, 12)	3	0.81922	0.786834	0.815047	4.429504	0.326019	0.557587	0.472318	0.482895
(14, 13, 10, 5, 7)	80	0.898522	0.877743	0.899687	17.50013	0.15047	0.895913	0.698308	0.760644
(14, 14, 13, 11, 15)	1	0.778898	0.76489	0.789969	4.73215	0.413793	0.254963	0.333333	0.288928
(6, 14, 15, 6, 11)	15	0.890457	0.887147	0.905956	5.83287	0.15047	0.844379	0.736929	0.768794
(5, 15, 12, 8, 15)	13	0.858199	0.868339	0.887147	7.312406	0.188088	0.774006	0.640839	0.689608
(6, 15, 7, 9, 14)	81	0.91129	**0.912226**	0.880878	16.44232	**0.115987**	0.857398	0.772705	0.806617
(8, 13, 7, 12, 15)	27	0.886425	0.893417	0.877743	7.545434	0.134796	0.845637	0.707055	0.760283
(6, 13, 11, 7, 8)	46	0.87836	0.874608	0.899687	10.41009	0.153605	0.88488	0.666754	0.739142
(12, 8, 15, 7, 14)	30	0.874328	0.880878	0.877743	9.728345	0.147335	0.884131	0.655476	0.724467
(6, 9, 12, 15, 11)	60	0.881048	0.862069	0.893417	13.84435	0.175549	0.837663	0.615175	0.682639

Table 4.2.12: Attainment of Evaluation Matrices of 6 Hidden Layers for Model 3

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(6, 7, 11, 7, 7, 7)	61	0.842742	0.868339	0.862069	13.95761	0.169279	0.526481	0.538934	0.53177
(5, 6, 13, 8, 13, 14)	3	0.778898	0.76489	0.789969	4.031484	0.413793	0.254963	0.333333	0.288928
(7, 7, 5, 13, 10, 11)	3	0.778898	0.76489	0.789969	4.554693	0.413793	0.254963	0.333333	0.288928
(9, 6, 12, 14, 14, 11)	88	0.909274	0.877743	0.890282	18.43455	0.178683	0.84848	0.698057	0.754812
(15, 8, 6, 5, 10, 8)	14	0.852823	0.858934	0.871473	5.837317	0.159875	0.843113	0.569611	0.603318
(6, 8, 7, 7, 14, 13)	77	0.893145	**0.893417**	0.884013	15.67176	**0.106583**	0.887845	0.686291	0.751933
(12, 15, 10, 11, 9, 5)	58	0.9375	**0.893417**	0.884013	13.48683	**0.125392**	0.884417	0.709222	0.764923

(11, 14, 11, 5, 9, 9)	48	0.907258	**0.893417**	0.865204	14.12153	**0.125392**	0.82066	0.790362	0.795461
(6, 15, 10, 15, 14, 8)	74	0.87164	0.887147	0.884013	15.74719	0.141066	0.911919	0.692982	0.767606
(14, 8, 5, 10, 9, 11)	81	0.907258	0.874608	0.880878	16.6107	0.153605	0.862812	0.599372	0.628643
(14, 5, 9, 15, 10, 14)	26	0.891801	0.896552	0.887147	7.934874	0.131661	0.909531	0.734023	0.796959
(15, 6, 15, 11, 14, 11)	15	0.880376	0.852665	0.874608	11.51894	0.203762	0.796745	0.680626	0.696279

And the followings are the diagrams for 'Accuracy' and 'CPU times' on Model 3.

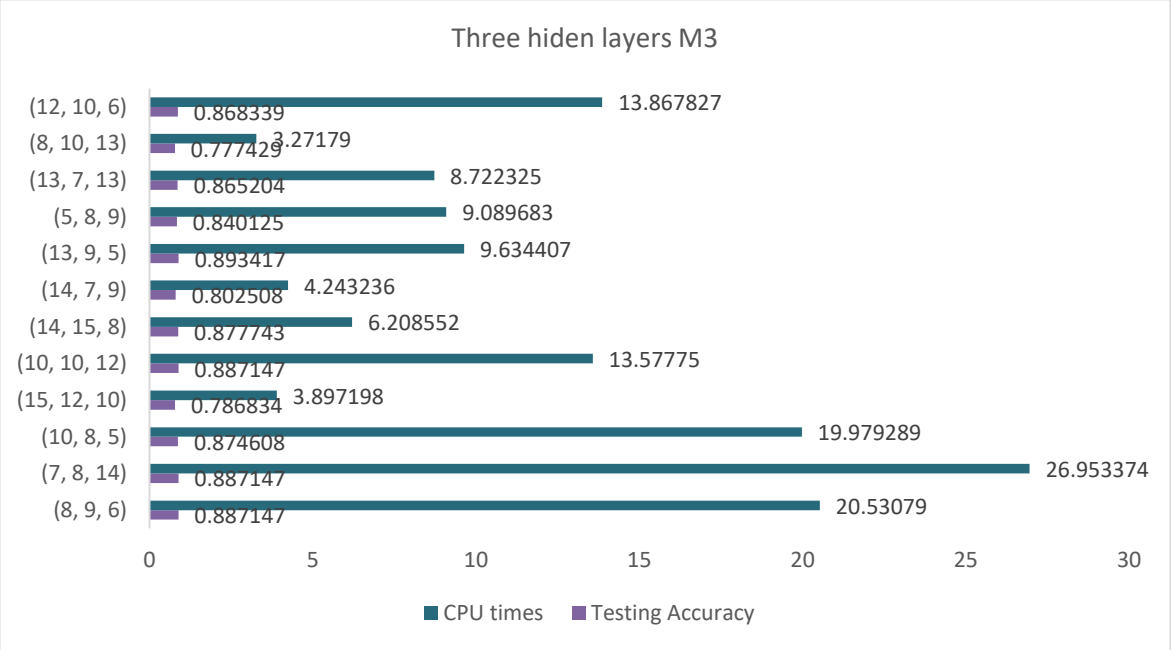

Figure 4.2.9 Accuracy and CPU times of 3 hidden layer for Model 3

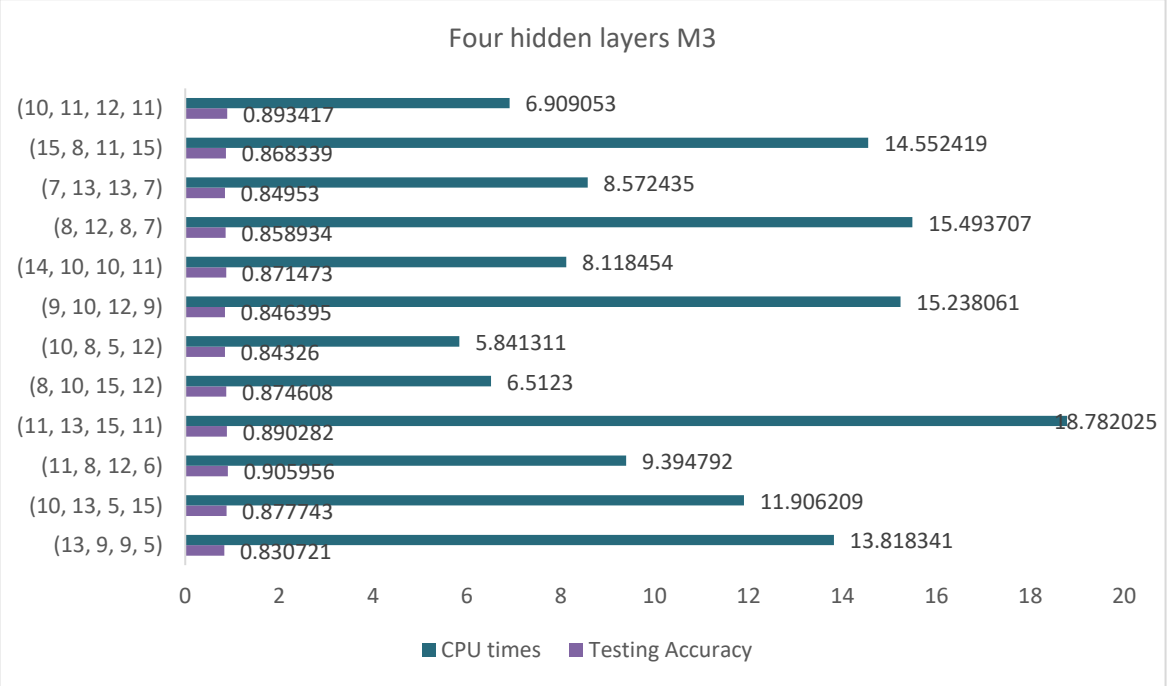

Figure 4.2.10 Accuracy and CPU times of 4 hidden layer for Model 3

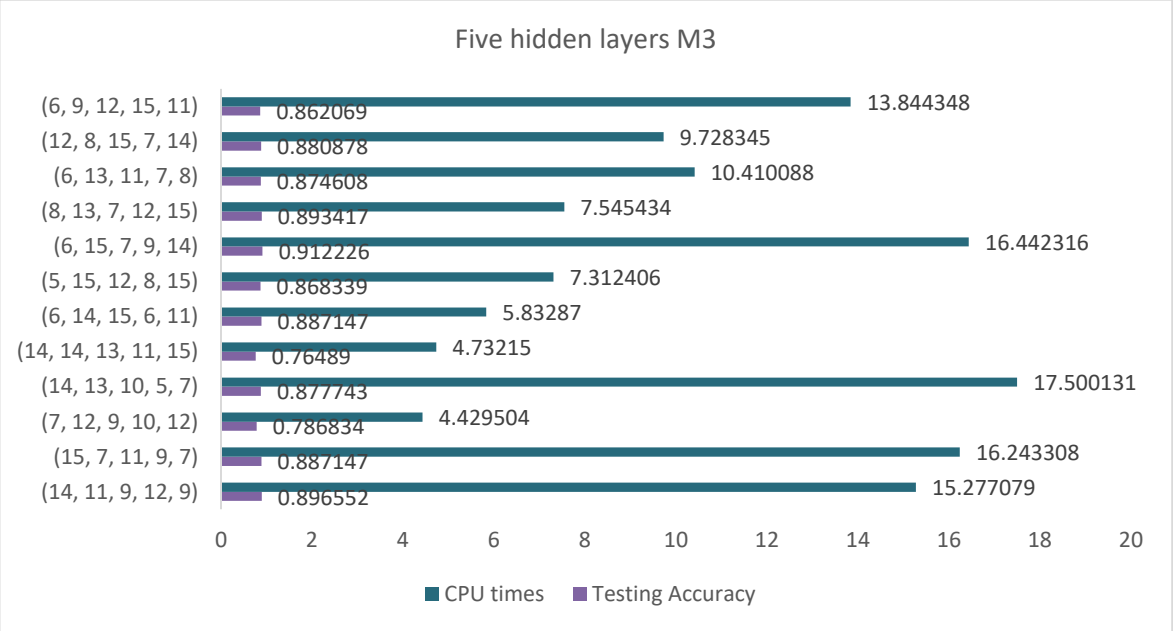

Figure 4.2.11 Accuracy and CPU times of 5 hidden layer for Model 3

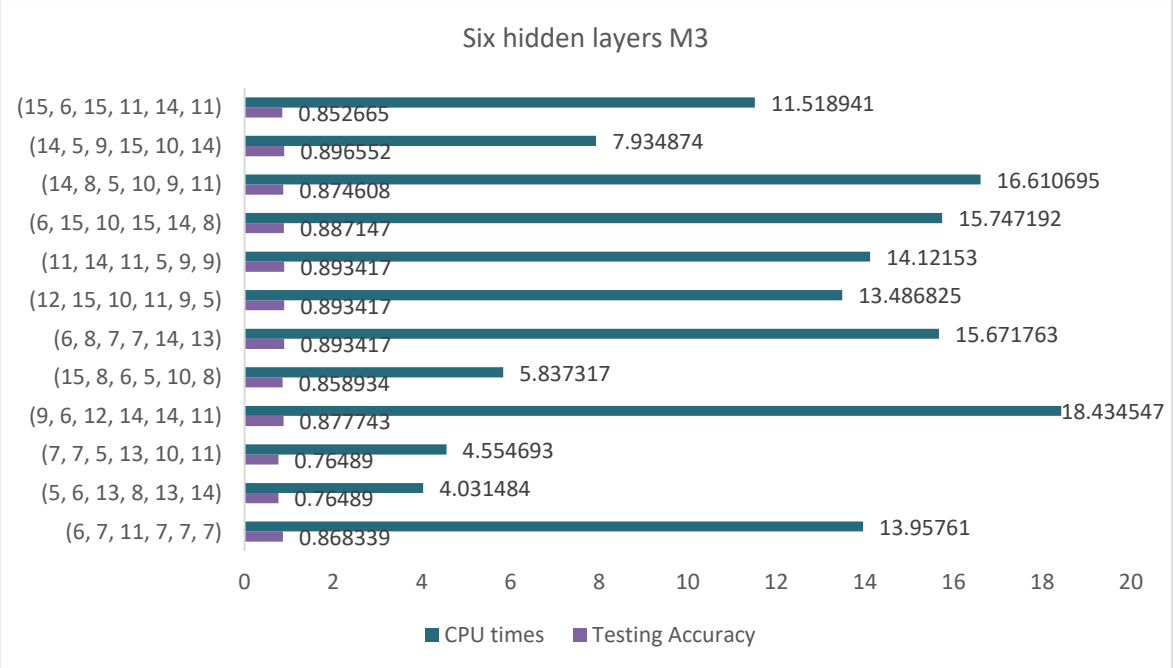

Figure 4.2.12 Accuracy and CPU times of 6 hidden layer for Model 3

These four above tables and figures basically shows the experimental outcome of Model 2 employing PReLU and Sigmoid activation functions of neural networks, along with Adam optimizer with 3,4,5,6 hidden layers sequentially. These tables and figures show distinct combination of neurons. It is clearly visible that the best performance is of 91% Testing Accuracy and the least of 0.11 MSE in the combinations of neurons with five number of hidden layers of Model 3 on table 4.2.11.

From the above, it is clear that Model 1, which uses ReLU activation in the hidden layers and Sigmoid activation in the output layer, demonstrated the best overall performance across various configurations. It consistently achieved higher accuracy, precision, and recall compared to the other models, particularly when utilizing 5 and 6 hidden layers. The combination of ReLU's efficiency in handling non-linear patterns and Sigmoid's suitability for multi-class classification made it the most effective for detecting fetal health conditions.

Training of FBNN models using different Optimizer

As the best number of layers is 5th layer from the previous part while employing ADAM optimizer, it was validated by employing different optimizers like SGD, RMSProp and Nadam optimizer to generate score for comparative analysis.

- **Training of 5th hidden layer of FBNN via SGD optimizer**

Table 4.2.13: Evaluation Matrices of 5 Hidden Layers for Model 1 Using SGD

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	66	0.952285	0.909091	0.865204	15.18483	0.054712	0.782493	0.829471	0.791757
(15, 7, 11, 9, 7)	44	0.932796	0.915361	0.896552	11.20189	0.054712	0.782493	0.829471	0.79175
(7, 12, 9, 10, 12)	28	0.932796	0.893417	0.874608	5.291155	0.054712	0.782493	0.829471	0.791757
(14, 13, 10, 5, 7)	44	0.956989	0.909091	0.905956	8.252591	0.054712	0.782493	0.829471	0.791757
(14, 14, 13, 11, 15)	7	0.899866	0.887147	0.884013	2.31952	0.054712	0.782493	0.829471	0.79175
(6, 14, 15, 6, 11)	13	0.90793	0.890282	0.890282	3.369794	0.054712	0.782493	0.829471	0.791757
(5, 15, 12, 8, 15)	75	0.965054	**0.924765**	0.896552	12.80661	**0.054712**	0.782493	0.829471	0.791757
(6, 15, 7, 9, 14)	8	0.910618	0.880878	0.893417	2.630916	0.054712	0.782493	0.829471	0.791757
(8, 13, 7, 12, 15)	81	0.938844	0.893417	0.852665	13.1958	0.054712	0.782493	0.829471	0.791757
(6, 13, 11, 7, 8)	37	0.915323	0.890282	0.884013	6.939324	0.054712	0.782493	0.829471	0.791757
(12, 8, 15, 7, 14)	46	0.948253	0.909091	0.871473	7.26652	0.054712	0.782493	0.829471	0.791757
(6, 9, 12, 15, 11)	11	0.908602	0.902821	0.884013	3.191775	0.054712	0.782493	0.829471	0.791757

Table 4.2.14: Evaluation Matrices of 5 Hidden Layers for Model 2 Using SGD

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	66	0.778898	0.76489	0.789969	12.62633	0.413793	0.254963	0.333333	0.288928
(15, 7, 11, 9, 7)	44	0.090726	0.059561	0.0721	9.302869	0.824451	0.019854	0.333333	0.037475
(7, 12, 9, 10, 12)	28	0.090726	0.059561	0.068966	5.955475	3.23511	0.019854	0.333333	0.037475
(14, 13, 10, 5, 7)	44	0.778898	0.76489	0.789969	9.785081	0.413793	0.254963	0.333333	0.288928
(14, 14, 13, 11, 15)	7	0.778898	0.76489	0.789969	3.060039	0.413793	0.254963	0.333333	0.288928
(6, 14, 15, 6, 11)	13	0.778898	0.76489	0.789969	4.93259	0.413793	0.254963	0.333333	0.28892

(5, 15, 12, 8, 15)	75	0.811156	**0.780564**	0.808777	12.42659	**0.413793**	0.51045	0.43723	0.44004
(6, 15, 7, 9, 14)	8	0.778898	0.76489	0.789969	4.150936	0.413793	0.254963	0.333333	0.288928
(8, 13, 7, 12, 15)	81	0.778898	0.76489	0.789969	13.99856	0.413793	0.254963	0.333333	0.288928
(6, 13, 11, 7, 8)	37	0.130376	0.175549	0.141066	7.053638	0.824451	0.058516	0.333333	0.099556
(12, 8, 15, 7, 14)	46	0.777554	0.761755	0.789969	9.1856	0.413793	0.254717	0.331967	0.288256
(6, 9, 12, 15, 11)	11	0.778898	0.76489	0.789969	3.53212	0.413793	0.254963	0.333333	0.288928

Table 4.2.15: Evaluation Matrices of 5 Hidden Layers for Model 2 Using SGD

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	66	0.930108	**0.896552**	0.890282	13.72679	**0.131661**	0.848567	0.687405	0.733294
(15, 7, 11, 9, 7)	44	0.921371	**0.896552**	0.912226	11.0808	**0.131661**	0.800023	0.733771	0.762849
(7, 12, 9, 10, 12)	28	0.877016	0.880878	0.887147	9.395337	0.15674	0.857248	0.65089	0.71719
(14, 13, 10, 5, 7)	44	0.903226	0.896552	0.909091	9.63515	0.131661	0.909085	0.752368	0.813725
(14, 14, 13, 11, 15)	7	0.869624	0.884013	0.880878	5.610776	0.153605	0.852984	0.666014	0.727115
(6, 14, 15, 6, 11)	13	0.843414	0.805643	0.833856	5.227732	0.231975	0.652154	0.547393	0.55596
(5, 15, 12, 8, 15)	75	0.841398	0.846395	0.84953	14.78513	0.181818	0.840499	0.546053	0.607232
(6, 15, 7, 9, 14)	8	0.860215	0.827586	0.874608	6.412521	0.219436	0.73223	0.596065	0.598448
(8, 13, 7, 12, 15)	81	0.900538	0.884013	0.899687	16.03042	0.134796	0.870016	0.691616	0.755641
(6, 13, 11, 7, 8)	37	0.874328	0.871473	0.890282	9.69287	0.175549	0.817486	0.642205	0.701556
(12, 8, 15, 7, 14)	46	0.891129	0.887147	0.884013	9.485765	0.15047	0.866543	0.678972	0.743609
(6, 9, 12, 15, 11)	11	0.821237	0.840125	0.836991	6.84453	0.225705	0.679554	0.487782	0.510782

These above three tables represents the experimental results for Model 1, Model 2 and Model 3 using the SGD optimizer. Among these models, Model 1 achieved the highest testing accuracy of 91%.

- **Training of 5th hidden layer of FBNN via RMSProp optimizer**

Table 4.2.16: Evaluation Matrices of 5 Hidden Layers for Model 1 Using RMSProp Optimizer

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	66	0.94422	0.920439	0.909091	11.84948	0.068966	0.898507	0.884234	0.891159
(15, 7, 11, 9, 7)	44	0.927419	0.896552	0.880878	7.329446	0.15047	0.802834	0.800649	0.801233
(7, 12, 9, 10, 12)	28	0.908602	0.884013	0.865204	6.972555	0.172414	0.765195	0.746903	0.748451
(14, 13, 10, 5, 7)	44	0.935484	0.890282	0.865204	7.191006	0.128527	0.749699	0.79575	0.764922
(14, 14, 13, 11, 15)	7	0.8125	0.758621	0.808777	3.508812	0.438871	0.389873	0.443845	0.413588
(6, 14, 15, 6, 11)	13	0.890457	0.877743	0.874608	4.205156	0.169279	0.756742	0.762768	0.744749
(5, 15, 12, 8, 15)	75	0.938172	0.890282	0.887147	12.90848	0.175549	0.767955	0.784158	0.769669
(6, 15, 7, 9, 14)	8	0.87164	0.874608	0.868339	2.542066	0.200627	0.759289	0.7358	0.74508

(8, 13, 7, 12, 15)	81	0.951613	**0.924765**	0.884013	21.42258	**0.094044**	0.86662	0.893581	0.879554
(6, 13, 11, 7, 8)	37	0.903226	0.884013	0.855799	7.877848	0.115987	0.750584	0.820787	0.776008
(12, 8, 15, 7, 14)	46	0.932796	0.912226	0.905956	9.414967	0.115987	0.816248	0.791302	0.802577
(6, 9, 12, 15, 11)	11	0.882392	0.855799	0.84953	3.293408	0.210031	0.689479	0.68799	0.682009

Table 4.2.17: Evaluation Matrices of 5 Hidden Layers for Model 2 Using RMSProp Optimizer

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	66	0.143145	0.128527	0.134796	12.28025	0.780564	0.323091	0.268237	0.154217
(15, 7, 11, 9, 7)	44	0.145161	0.178683	0.147335	10.28657	0.648903	0.280239	0.362469	0.15917
(7, 12, 9, 10, 12)	28	0.130376	0.175549	0.141066	7.902111	0.39185	0.058516	0.333333	0.09955
(14, 13, 10, 5, 7)	44	0.223118	0.175549	0.175549	8.022383	0.611285	0.334994	0.240447	0.15824
(14, 14, 13, 11, 15)	7	0.334005	0.366771	0.297806	4.571135	0.401254	0.298042	0.402502	0.23167
(6, 14, 15, 6, 11)	13	0.811828	0.846395	0.818182	4.589077	0.413793	0.846826	0.550136	0.555927
(5, 15, 12, 8, 15)	75	0.133065	0.178683	0.141066	14.3698	0.467085	0.392034	0.350877	0.133155
(6, 15, 7, 9, 14)	8	0.264785	0.275862	0.213166	3.377135	0.416928	0.372351	0.329774	0.235245
(8, 13, 7, 12, 15)	81	0.797043	0.777429	0.805643	14.98025	0.401254	0.52569	0.403509	0.402628
(6, 13, 11, 7, 8)	37	0.903226	**0.884013**	0.855799	7.877848	**0.115987**	0.750584	0.820787	0.776008
(12, 8, 15, 7, 14)	46	0.303763	0.360502	0.282132	9.00244	0.401254	0.488006	0.539273	0.363473
(6, 9, 12, 15, 11)	11	0.077957	0.056426	0.059561	3.577858	0.965517	0.043844	0.292607	0.05044

Table 4.2.18: Evaluation Matrices of 5 Hidden Layers for Model 3 Using RMSProp Optimizer

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	66	0.882392	0.874608	0.868339	12.05138	0.144201	0.862799	0.650325	0.706894
(15, 7, 11, 9, 7)	44	0.923387	**0.890282**	0.880878	8.909867	**0.137931**	0.807807	0.767729	0.78144
(7, 12, 9, 10, 12)	28	0.880376	0.880878	0.874608	8.079148	0.15674	0.828913	0.701842	0.746385
(14, 13, 10, 5, 7)	44	0.887769	0.880878	0.887147	8.041957	0.15674	0.80649	0.678659	0.72665
(14, 14, 13, 11, 15)	7	0.858199	0.874608	0.871473	5.268353	0.181818	0.840616	0.620388	0.681785
(6, 14, 15, 6, 11)	13	0.867608	0.877743	0.884013	4.32931	0.159875	0.806162	0.665701	0.71787
(5, 15, 12, 8, 15)	75	0.895833	0.887147	0.896552	14.07274	0.131661	0.857915	0.736678	0.785714
(6, 15, 7, 9, 14)	8	0.884409	**0.890282**	0.865204	3.796327	**0.15674**	0.797713	0.72862	0.757986
(8, 13, 7, 12, 15)	81	0.897849	0.887147	0.887147	15.51417	0.15047	0.908109	0.743935	0.792568
(6, 13, 11, 7, 8)	37	0.872312	0.887147	0.899687	8.482132	0.141066	0.89731	0.658208	0.73032
(12, 8, 15, 7, 14)	46	0.836022	0.846395	0.862069	10.71497	0.181818	0.838462	0.534461	0.588746
(6, 9, 12, 15, 11)	11	0.811156	0.840125	0.833856	4.076389	0.263323	0.529309	0.480777	0.491423

These above three tables represents the experimental results for Model 1, Model 2 and Model 3 using the RMSProp optimizer. Among these models, Model 1 achieved the highest testing accuracy of 92%. Where, Model 2 acquired 88% of highest testing accuracy and Model 3 acquired 89% of best testing accuracy.

- **Training of 5th hidden layer of FBNN via Nadam optimizer.**

Table 4.2.19: Evaluation Matrices of 5 Hidden Layers for Model 1 Using Nadam Optimizer

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	66	0.952957	0.909091	0.852665	12.80212	0.109718	0.807315	0.847642	0.825121
(15, 7, 11, 9, 7)	44	0.965054	**0.924765**	0.899687	10.85917	**0.103448**	0.871796	0.868231	0.86999
(7, 12, 9, 10, 12)	28	0.934812	0.902821	0.887147	6.200522	0.134796	0.811603	0.824145	0.817056
(14, 13, 10, 5, 7)	44	0.956989	0.905956	0.915361	10.30971	0.15047	0.825769	0.790989	0.802367
(14, 14, 13, 11, 15)	7	0.895833	0.899687	0.865204	3.664471	0.128527	0.818111	0.806853	0.804278
(6, 14, 15, 6, 11)	13	0.905242	0.877743	0.877743	7.952612	0.159875	0.764596	0.77194	0.761218
(5, 15, 12, 8, 15)	75	0.965054	0.912226	0.896552	14.29698	0.115987	0.835126	0.849008	0.841202
(6, 15, 7, 9, 14)	8	0.892473	0.858934	0.868339	3.771139	0.188088	0.709583	0.78234	0.72323
(8, 13, 7, 12, 15)	81	0.952957	0.905956	0.884013	15.01081	0.122257	0.821485	0.830098	0.825602
(6, 13, 11, 7, 8)	37	0.924059	0.896552	0.890282	16.38606	0.131661	0.805235	0.805235	0.805235
(12, 8, 15, 7, 14)	46	0.956317	0.896552	0.905956	10.84887	0.141066	0.779076	0.830837	0.79139
(6, 9, 12, 15, 11)	11	0.898522	0.862069	0.887147	4.6402	0.166144	0.727891	0.765109	0.741286

Table 4.2.20: Evaluation Matrices of 5 Hidden Layers for Model 2 Using Nadam Optimizer

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	66	0.778898	**0.76489**	0.789969	14.70092	**0.413793**	0.254963	0.333333	0.288928
(15, 7, 11, 9, 7)	44	0.395833	0.407524	0.429467	10.40785	0.489028	0.223368	0.177596	0.197869
(7, 12, 9, 10, 12)	28	0.139113	0.178683	0.15674	9.220576	0.373041	0.093267	0.362469	0.136911
(14, 13, 10, 5, 7)	44	0.088038	0.053292	0.081505	11.16358	1.153605	0.060892	0.26589	0.036108
(14, 14, 13, 11, 15)	7	0.091398	0.059561	0.068966	4.196876	1.617555	0.019854	0.333333	0.037475
(6, 14, 15, 6, 11)	13	0.74328	0.714734	0.768025	5.073642	0.401254	0.283951	0.343831	0.311017
(5, 15, 12, 8, 15)	75	0.778898	0.76489	0.789969	15.25565	0.413793	0.254963	0.333333	0.28892
(6, 15, 7, 9, 14)	8	0.662634	0.633229	0.680251	5.713995	0.413793	0.281844	0.298888	0.29011
(8, 13, 7, 12, 15)	81	0.716398	0.689655	0.730408	15.94623	0.429467	0.314073	0.33265	0.32012
(6, 13, 11, 7, 8)	37	0.778898	0.76489	0.789969	10.76312	0.413793	0.254963	0.333333	0.288928
(12, 8, 15, 7, 14)	46	0.221774	0.200627	0.206897	9.448766	1.141066	0.342433	0.394809	0.146621
(6, 9, 12, 15, 11)	11	0.130376	0.175549	0.141066	6.075926	0.626959	0.058516	0.333333	0.09955

Table 4.2.21: Evaluation Matrices of 5 Hidden Layers for Model 3 Using Nadam Optimizer

Combination of Neurons	Epochs	Training Accuracy	Testing Accuracy	Validation Accuracy	CPU times	MSE	Precision	Recall	F1 Score
(14, 11, 9, 12, 9)	66	0.91129	0.880878	0.868339	15.47873	0.137931	0.802122	0.667067	0.7176
(15, 7, 11, 9, 7)	44	0.906586	0.880878	0.862069	12.41189	0.147335	0.866313	0.657391	0.68933
(7, 12, 9, 10, 12)	28	0.910618	**0.899687**	0.884013	8.500177	**0.128527**	0.852134	0.779084	0.81039
(14, 13, 10, 5, 7)	44	0.872984	0.877743	0.884013	10.78429	0.15047	0.884423	0.644937	0.716551
(14, 14, 13, 11, 15)	7	0.860215	0.84326	0.871473	6.761468	0.231975	0.727698	0.581628	0.628182
(6, 14, 15, 6, 11)	13	0.877016	0.877743	0.887147	6.000347	0.15047	0.887657	0.702895	0.764043
(5, 15, 12, 8, 15)	75	0.882392	0.852665	0.855799	16.21777	0.184953	0.75281	0.576302	0.624452
(6, 15, 7, 9, 14)	8	0.843414	0.836991	0.852665	6.716266	0.191223	0.780314	0.57431	0.633096
(8, 13, 7, 12, 15)	81	0.891801	0.874608	0.877743	16.8046	0.163009	0.836335	0.636566	0.696914
(6, 13, 11, 7, 8)	37	0.903898	0.887147	0.884013	10.92475	0.131661	0.849798	0.796802	0.81966
(12, 8, 15, 7, 14)	46	0.892473	0.896552	0.890282	12.26268	0.122257	0.908072	0.699248	0.77091
(6, 9, 12, 15, 11)	11	0.893145	0.884013	0.890282	5.772036	0.163009	0.789146	0.730725	0.754821

These above three tables represents the experimental results for Model 1, Model 2 and Model 3 using the Nadam optimizer. Among these models, Model 1 achieved the highest testing accuracy of 91%, Where Model 3 acquired 89% of best testing accuracy, second highest among these three models. And model 2 achieved third position at only 76%, which is average in this case.

4.3 Comparative Analysis

Table 4.3.1 Comparative analysis of different optimizers

Optimizers		Best Testing Accuracy	
	Model 1	Model 2	Model 3
SGD	0.924765	0.780564	0.896552
RMSProp	0.920439	0.884013	0.890282
Nadam	0.912226	0.764896	0.899687
ADAM	**0.940439**	0.84326	0.912226

4.4 Summary

This chapter describes the practical execution of the proposed methodology, including environment setup, testing, evaluation, and analysis of results. The experimental setup

utilized an AMD Ryzen 3 PRO processor, 8 GB RAM, and Radeon Vega 6 graphics, alongside Python libraries such as TensorFlow, Keras, and scikit-learn, to develop and train Feedforward Neural Networks (FBNNs). The *"Fetal Health Classification"* dataset from Kaggle was preprocessed by normalizing features and encoding target variables, ensuring compatibility with deep learning algorithms. Three FBNN models were tested, each using different activation functions and optimizers. Model 1, with ReLU in the hidden layers and Sigmoid in the output layer, consistently outperformed the others. It achieved the highest testing accuracy of 94% with the ADAM optimizer and maintained strong results with SGD (92%), RMSProp (92%), and Nadam (91%). Comparative analyses demonstrated that Model 1's configuration was the most effective for detecting fetal health conditions, highlighting its reliability in learning and classifying complex patterns in clinical data. All of these findings underscore the importance of optimizing neural network architectures, with the combination of ReLU, Sigmoid, and ADAM emerging as the most efficient setup.

Chapter 5

Engineering Standards and Design Challenges

5.1 Compliance with the Standards

This section focuses on the engineering standards directly relevant to this project, evaluates possible alternatives, and discusses their pros, cons, and the rationale behind their selection.

5.1.1 Software Standards

For this research-based project, Python's PEP 8 coding standards were being adopted in order to maintain code readability, consistency, and maintainability. Python was chosen as the programming language due to its versatility and wide range of libraries such as TensorFlow, Keras, scikit-learn, Pandas, NumPy, Matplotlib, Seaborn, SciPy, imbalanced-learn, OpenCV, PyTorch, Plotly, SHAP, and LIME for deep learning, data analysis, and visualization. The development environment was Google Colab and Jupyter Notebook.

Alternatives Considered:

- ✓ R: Known for statistical analysis but less suitable for deep learning tasks compared to Python.
- ✓ MATLAB: Although it offers robust numerical computing, but is limited by proprietary licensing and fewer community resources for deep learning.

Rationale for Selection:

Python's integration with TensorFlow, Keras, and scikit-learn provided a flexible and efficient development environment tailored to deep learning tasks. Its open-source nature and extensive library ecosystem made it the most practical choice for this research.

5.1.2 Hardware Standards

The hardware configuration consists of an AMD Ryzen 3 PRO processor, 8 GB RAM, and

Radeon Vega 6 integrated graphics. This setup met the computational needs for training Feedforward Neural Networks (FBNNs) without requiring additional investments.

Alternatives Considered:

- ✓ NVIDIA GPUs: These GPUs provides superior performance for deep learning due to CUDA support and optimized drivers but involve higher costs.
- ✓ Cloud Computing Platforms(AWS, Google Cloud): Offers scalable resources but incurs significant costs for extended usage and has potential latency issues.

Rationale for Selection:

The existing hardware configuration was chosen for its availability, cost-efficiency, and ability to handle the dataset and model complexity effectively. Although NVIDIA GPUs could have improved performance, they were not necessary for this research's scale.

5.2 Impact on Society, Environment and Sustainability

This section explores how the research contributes to improving lives, its impact on society and the environment, adherence to ethical principles, and plans for long-term sustainability.

5.2.1 Impact on Life

The developed system for detecting fetal health conditions has a direct and significant impact on improving maternal and fetal health. By enabling early detection of potential health issues, the system supports timely medical interventions, reducing risks during pregnancy. Classifying conditions into Normal, Suspect, and Pathological categories provides clarity for healthcare providers, helping them make better decisions to ensure the safety and well-being of both mother and child. The reduction in dependence on traditional operator-based methods also makes the diagnosis process more consistent and reliable, particularly in under-resourced healthcare settings.

5.2.2 Impact on Society & Environment

This research-based project "Towards the early detection of fetal health using deep learning" will provide positive impact to the society by making advanced diagnostic tools more accessible and cost-effective. By utilizing open-source tools and existing computational resources, the system reduces the financial burden on healthcare providers

and patients. Its scalable design means it can be implemented in a variety of clinical environments, from urban hospitals to rural clinics.

Additionally, the environmental impact is actually minimized as the project relies on locally available hardware rather than high-power, resource-intensive servers or cloud computing systems. This reduces the overall energy consumption and carbon footprint, making the project environmentally sustainable. Additionally, by avoiding disposable diagnostic tools, the system indirectly reduces medical waste, further contributing to environmental protection.

5.2.3 Ethical Aspects

- ✓ Data Privacy and Anonymization: The research uses anonymized dataset to ensure no sensitive personal information is included, guaranteeing the privacy and confidentiality of patients.
- ✓ Informed Consent for Future Data Use: While the current dataset is publicly available, any future data collection will follow ethical guidelines, including obtaining informed consent from patients to use their medical data for research.
- ✓ Responsible Use of Results: The outcomes of this research are intended to supplement, not replace, medical expertise, ensuring that the system supports healthcare professionals rather than making standalone decisions.

5.2.4 Sustainability Plan

- ✓ Use of Open-Source Tools: The project relies on open-source frameworks and libraries, significantly reducing long-term costs and dependency on proprietary software.
- ✓ Scalability for Larger Datasets: The system is designed to scale efficiently, enabling it to handle larger datasets and adapt to future technological advancements in fetal health diagnostics.
- ✓ Long-Term Impact on Healthcare: By addressing current challenges and staying adaptable, the project supports continuous improvement in prenatal care

5.3 Project Management and Financial Analysis

This section outlines the project's financial requirements and management strategies, focusing on cost-effective implementation and scalability for future use.

Cost Analysis: This research basically utilizes free, open-source software packages and available hardware, minimizing expenses. However, potential costs might rise for high-performance computing resources or cloud services, such as AWS or Google Cloud, in order to train and test larger models efficiently. Below is the cost breakdown of proposed budget and alternate budget for scalability for this thesis project:

Table 5.3.1 Proposed Budget

Serial No	Components	Rationale	Estimated Cost (BDT)
01	Existing Hardware Utilization	Current resources (AMD Ryzen 3 PRO, 8 GB RAM, Radeon Vega 6) are sufficient for the initial scope	No additional cost
02	Open-Source Software	Tools like TensorFlow, Keras, and Python libraries eliminate the need for software purchases	No additional cost
03	Documentation and Reporting	Covers the preparation of research documentation and academic publications	3,000
04	Model Testing and Optimization (Computation Costs)	Includes the electricity and resource costs for multiple iterations of training and optimization	5,000
Total Estimated Cost			8,000

This proposed budget (Table 5.3.1) basically explains existing hardware and open-source software to minimize costs, focusing on the immediate research and development needs.

Table 5.3.2 Alternate Budget for Scalability

Serial No	Components	Rationale	Estimated Cost (BDT)

01	High-Performance GPU	Required for faster training and handling large-scale datasets	50,000
02	Cloud Services (AWS/Google)	To support real-time data processing and scalability	20,000
03	Model Deployment Infrastructure	For integrating the system into clinical environments or telemedicine platforms	30,000
04	Model Testing and Optimization (Computation Costs)	Additional costs for extensive computation during scalability testing with larger datasets	8,000
Total Estimated Cost			108,000

Rationale for Alternate Budget:

While the proposed budget focuses on research and development using existing resources, the alternate budget addresses the requirements for broader clinical implementation. This includes computational upgrades and infrastructure for real-time applications, ensuring the system's adaptability for diverse clinical environments.

Revenue Model:

The system can be monetized by offering it as a subscription-based tool to clinics and hospitals. Additionally, licensing the software to healthcare providers and integrating it into existing diagnostic platforms can generate sustainable revenue. The model emphasizes affordability and scalability, ensuring its widespread adoption in various healthcare settings.

5.4 Complex Engineering Problem

5.4.1 Complex Problem Solving

This section evaluates the complex problem-solving aspects of the thesis, "Towards the Early Detection of Fetal Health Using Deep Learning," using a mapping table to categorize challenges based on engineering practices and knowledge profiles.

Table 5.4.1: Mapping with complex problem solving.

EP1 Dept of Knowledge	EP2 Range Of Conflicting Requirements	EP3 Depth of Analysis	EP4 Familiarity of Issues	EP5 Extent of ApplicableCodes	EP6 Extent Of Stake-holder Involvement	EP7 Interdependence
✓		✓	✓			✓

Mapping with Knowledge Profile for EP1

This following table 5.4.2 is designed to map the EP1 to the Knowledge Profile

Table 5.4.2: Mapping with knowledge Profile.

K3 Engineering Fundamentals	K4 Specialist Knowledge	K5 Engineering Design	K6 Engineering Practice	K8 Research Literature
✓	✓	✓	✓	✓

Rationale for Selected Categories

The project involves complex engineering challenges requiring expertise in deep learning, FBNNs, and clinical data processing (EP1), relying on K3, K4, K5, K6, and K8 knowledge. Balancing accuracy, computational efficiency, and resource constraints (EP2) necessitates trade-offs in architecture design. The analysis explores configurations of activation functions, optimizers, and hidden layers, addressing issues like class imbalances (EP3). Tackling uncommon challenges in automating CTG data analysis bridges technical expertise with clinical applications (EP4). The project's interdependence is evident in integrating data preprocessing, model training, evaluation, and interpretation, ensuring seamless functionality across all stages (EP7).

5.1.1 Engineering Activities

Mapping with complex engineering activities

The project utilizes locally available computational resources and open-source tools, making it cost-efficient while addressing scalability challenges. It introduces an innovative use of Feedforward Neural Networks (FBNNs) for detecting fetal health conditions, providing a more effective alternative to traditional CTG-based diagnostics.

Table 5.4.3: Mapping with complex engineering activities

EA1 Range of resources	EA2 Level of Interaction	EA3 Innovation	EA4 Consequences for society and environment	EA5 Familiarity
✓		✓	✓	

5.5 Summary

This chapter focuses on the engineering standards, societal impacts, financial considerations, and problem-solving aspects of the research on "Towards the Early Detection of Fetal Health Using Deep Learning." Python's PEP 8 coding standards and open-source libraries were utilized alongside existing hardware for cost-effective implementation. The system enhances maternal and fetal health outcomes by enabling early detection, improving diagnostic consistency, and minimizing environmental impact through resource-efficient tools. Ethical considerations like data privacy, informed consent, and responsible use are integral, with a sustainability plan ensuring adaptability to larger datasets and future advancements. A proposed budget of 8,000 BDT leverages available resources, while an alternate budget of 108,000 BDT supports scalability through high-performance GPUs and cloud services. The research addresses complex engineering challenges, including deep learning optimization and data integration, relying on engineering fundamentals (K3), specialist knowledge (K4), and research literature (K8). The project balances technological innovation with societal and environmental responsibility.

Chapter 6

Conclusion

This chapter provides a comprehensive overview of the research findings, highlights the limitations encountered during the study, and outlines potential areas for future work to expand and enhance the project.

6.1 Summary

This research promoted the establishment of an early detection system of fetal health greatly using cardiotocography (CTG) data based on deep learning. The study carried an investigation on Feedforward Neural Networks (FBNNs) in classifying fetal health into three: Normal, Suspect, and Pathological. It has been employed several activation functions such as ReLU, PReLU, Tanh, and Sigmoid along with optimizers such as Adam, RMSprop, and SGD.

The "Fetal Health Classification" dataset available on Kaggle contained 2,126 clinical records and was preprocessed before being divided into training, validation, and test datasets [18]. Model 1, using ReLU and Sigmoid, showed the best outcome with high levels of accuracy and precision, and also high recall, in particular for the 5 hidden layers.

The results manifest the potential of deep learning models to solve problems associated with existing methods of diagnosis, scalable and efficient in fetal health classification. The study is part of the ongoing domain of AI in healthcare focusing more on prenatal care.

6.2 Limitation

- ✓ Dataset Size: The dataset size (2,126 records) was relatively small, limiting the model's generalizability for larger and more diverse populations.
- ✓ Class Imbalance: Imbalanced data among the three categories may have influenced the accuracy for minority classes, such as *Pathological*.
- ✓ Resource Constraints: The absence of high-performance GPUs and advanced computational resources increased training time and restricted experimentation

with deeper neural networks.

- ✓ Clinical Validation: The model was not validated on real-time clinical data, restricting its practical implementation for now.

6.3 Future Work

- ✓ Larger Dataset: Incorporate larger and more diverse datasets to improve the model's generalizability and reliability.
- ✓ Real-Time Testing: Implement the system in clinical environments to validate its effectiveness on real-time data.
- ✓ Deployment: Develop a user-friendly interface for clinical use and integrate the system into existing hospital workflows for broader adoption.

References

[1] Akbulut, A., Ertugrul, E., & Topcu, V. (2018). Fetal health status prediction based on maternal clinical history using machine learning techniques. Computer methods and programs in biomedicine, 163, 87-100.

[2] Deressa, T. D., & Kadam, K. (2018). Prediction of fetal health state during pregnancy: a survey. Int. J. Comput. Sci. Trends Technol.(IJCST), 6(1).

[3] Miao, J. H., & Miao, K. H. (2018). Cardiotocographic diagnosis of fetal health based on multiclass morphologic pattern predictions using deep learning classification. International Journal of Advanced Computer Science and Applications, 9(5).d

[4] Piri, J., & Mohapatra, P. (2019, December). Exploring fetal health status using an association based classification approach. In 2019 International Conference on Information Technology (ICIT) (pp. 166-171). IEEE.

[5] Mushtaq, G., & Veningston, K. (2024). AI driven interpretable deep learning based fetal health classification. SLAS technology, 29(6), 100206.

[6] Selvathi, D., & Chandralekha, R. (2022). Fetal biometric based abnormality detection during prenatal development using deep learning techniques. Multidimensional systems and signal processing, 33(1), 1-15.

[7] Petrozziello, A., Jordanov, I., Papageorghiou, T. A., Redman, W. C., & Georgieva, A. (2018, July). Deep learning for continuous electronic fetal monitoring in labor. In 2018 40th annual international conference of the IEEE engineering in medicine and biology society (EMBC) (pp. 5866-5869). IEEE.

[8] Cao, Z., Wang, G., Xu, L., Li, C., Hao, Y., Chen, Q., ... & Wei, H. (2023). Intelligent antepartum fetal monitoring via deep learning and fusion of cardiotocographic signals and clinical data. Health Information Science and Systems, 11(1), 16.

[9] Singha, A., Noel, J. R. S., Adhikrishna, R. V., Suthahar, N., Abinesh, S., & Poorni, S. J. S. (2023). Fetal health status prediction during labor and delivery based on cardiotocogram data using machine and deep learning. In AI and Blockchain in Healthcare (pp. 105-135). Singapore: Springer Nature Singapore.

[10] Salini, Y., Mohanty, S. N., Ramesh, J. V. N., Yang, M., & Chalapathi, M. M. V. (2024). Cardiotocography Data Analysis for Fetal Health Classification Using Machine Learning Models. IEEE Access.

[11] Bhowmik, P., Bhowmik, P. C., Ali, U. M. E., & Sohrawordi, M. (2021). Cardiotocography data analysis to predict fetal health risks with tree-based ensemble learning. Inf. Technol. Comput. Sci, 5, 30-40.

[12] Sridevi, S., Devi, M. S., Reddy, K. V. K., & Dadi, R. H. (2021). Integrated UnderOversampling Responsive based Fetal Health Prediction using Cardiotocographic Data. Turkish Journal of Computer and Mathematics Education, 12(9), 1743-1757.

[13] Cömert, Z., & Kocamaz, A. F. (2016). Evaluation of fetal distress diagnosis during delivery stages based on linear and nonlinear features of fetal heart rate for neural network community. Int. J. Comput. Appl, 156(4), 26-31.

[14] Activation Functions in Neural Networks [12 Types & Use Cases]. Available online:https://www.v7labs.com/blog/neural-networks-activation-functions (accessed on 22nd October 2023).

[15] Butt, U. A., Mehmood, M., Shah, S. B. H., Amin, R., Shaukat, M. W., Raza, S. M., ... & Piran, M. J. (2020). A review of machine learning algorithms for cloud computing security. Electronics, 9(9), 1379.

[16] Zhu, S., Han, H., Guo, M., & Qiao, J. (2017). A data-derived soft-sensor method for monitoring effluent total phosphorus. Chinese journal of chemical engineering, 25(12), 1791-1797.

[17] Chinatamby, P., & Jewaratnam, J. (2023). A performance comparison study on PM2. 5 prediction at industrial areas using different training algorithms of feedforward-backpropagation neural network (FBNN). Chemosphere, 317, 137788.

[18] Learn about the Dataset – "Fetal Health Classification", available at << https://www.kaggle.com/datasets/andrewmvd/fetal-health-classification/>>.

[19] Learn about maternal mortality rate by country available at << https://data.unicef.org/topic/maternal-health/maternal-mortality/>>.

[20] Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013, June). Rectifier nonlinearities improve neural network acoustic models. In Proc. icml (Vol. 30, No. 1, p. 3).

[21] Sharma, S., Sharma, S., & Athaiya, A. (2017). Activation functions in neural networks. Towards Data Sci, 6(12), 310-316.

[22] Han, J., & Moraga, C. (1995, June). The influence of the sigmoid function parameters on the speed of backpropagation learning. In International workshop on artificial neural networks (pp. 195-201). Berlin, Heidelberg: Springer Berlin Heidelberg.

[23] Han, J., & Moraga, C. (1995, June). The influence of the sigmoid function parameters on the speed of backpropagation learning. In International workshop on artificial neural networks (pp. 195-201). Berlin, Heidelberg: Springer Berlin Heidelberg.

[24] Wallach, D., & Goffinet, B. (1989). Mean squared error of prediction as a criterion for evaluating and comparing system models. Ecological modelling, 44(3-4), 299-306.

[25] Swets, J. A. (1988). Measuring the accuracy of diagnostic systems. Science, 240(4857), 1285-1293.

[26] Powers, D. M. (2020). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061.

[27] Stehman, S. V. (1997). Selecting and interpreting measures of thematic classification accuracy. Remote sensing of Environment, 62(1), 77-89.

[28] Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Artificial-neural-network-architecture-ANN-i-h-1-h-2-h-n-o_fig1_321259051 [accessed 4 Jan, 2024].

ID_211-15-4073_FYDP_REPORT.pdf

ORIGINALITY REPORT

12%	9%	6%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dspace.daffodilvarsity.edu.bd:8080 Internet Source	2%
2	Submitted to Daffodil International University Student Paper	1%
3	Submitted to United International University Student Paper	1%
4	Jagmohan Kaur, Baljit S. Khehra, Amarinder Singh. "Back propagation artificial neural network for diagnose of the heart disease", Journal of Reliable Intelligent Environments, 2022 Publication	1%
5	deepai.org Internet Source	<1%
6	ebin.pub Internet Source	<1%
7	Vivek S. Sharma, Shubham Mahajan, Anand Nayyar, Amit Kant Pandit. "Deep Learning in Engineering, Energy and Finance - Principles and Applications", CRC Press, 2024	<1%