ujjwal bharti

thesis(1) UJJWAL.pdf

Delhi Technological University

Document Details

Submission ID
trn:oid:::27535:98724094

Submission Date
May 31, 2025, 11:16 PM GMT+5:30

Download Date
May 31, 2025, 11:19 PM GMT+5:30

File Name
thesis(1) UJJWAL.pdf

File Size
891.9 KB

27 Pages

5,606 Words

32,048 Characters

10% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography

Match Groups

- **37** Not Cited or Quoted 8%
  Matches with neither in-text citation nor quotation marks
- **7** Missing Quotations 1%
  Matches that are still very similar to source material
- **2** Missing Citation 1%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

7% Internet sources
6% Publications
7% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **37** Not Cited or Quoted 8%
  Matches with neither in-text citation nor quotation marks
- **7** Missing Quotations 1%
  Matches that are still very similar to source material
- **2** Missing Citation 1%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

7% Internet sources
6% Publications
7% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	
www.frontiersin.org		<1%
2	Internet	
dergipark.org.tr		<1%
3	Submitted works	
National Institute of Technology, Raipur on 2023-06-16		<1%
4	Publication	
"Advances in Data and Information Sciences", Springer Science and Business Med...		<1%
5	Internet	
krishikosh.egranth.ac.in		<1%
6	Internet	
ekeeda.com		<1%
7	Publication	
Vinod M. Kapse, Lalit Garg, Pavan Kumar Shukla, Varadraj Gurupur, Amit Krishna ...		<1%
8	Internet	
www.mdpi.com		<1%
9	Submitted works	
B.V. B College of Engineering and Technology, Hubli on 2025-01-12		<1%
10	Internet	
www.ijisae.org		<1%

11	Publication	A. Sakshi, Tushar Mehrotra, Priyanka Tyagi, Vishal Jain. "Emerging trends in hybri... <1%
12	Internet	www.e3s-conferences.org <1%
13	Internet	link.springer.com <1%
14	Internet	minerva-access.unimelb.edu.au <1%
15	Internet	gcris.khas.edu.tr <1%
16	Internet	journals.plos.org <1%
17	Internet	www.microsoft.com <1%
18	Internet	www.nature.com <1%
19	Submitted works	University of Northampton on 2023-01-20 <1%
20	Publication	V. Govindaraj, B. Shoban Babu, C. Ezhilazhagan. "Back propagation neural networ... <1%
21	Internet	academic-accelerator.com <1%
22	Publication	"AI-Enabled Electronic Circuit and System Design", Springer Science and Business ... <1%
23	Submitted works	University College London on 2025-03-30 <1%
24	Internet	escholarship.org <1%

25 Internet
iris.poliba.it <1%

26 Internet
oa.las.ac.cn <1%

27 Internet
www.ijmrset.com <1%

28 Publication
"Intelligent Systems Design and Applications", Springer Science and Business Me... <1%

29 Submitted works
International Institute of Information Technology, Hyderabad on 2024-12-21 <1%

30 Publication
Sridevi Nagarajan, Prasanna Kumar Mahadeviah. "On-chip based power estimati... <1%

31 Internet
fau.digital.flvc.org <1%

32 Internet
ijcem.in <1%

33 Internet
lib.buet.ac.bd:8080 <1%

34 Submitted works
Indian Institute of Information Technology, Design and Manufacturing - Kanchee... <1%

35 Submitted works
University of Leeds on 2023-09-17 <1%

36 Publication
V. Govindaraj, B. Arunadevi. "Machine Learning Based Power Estimation for CMO... <1%

37 Publication
T. Mariprasath, Kumar Reddy Cheepati, Marco Rivera. "Practical Guide to Machin... <1%

ABSTRACT

This study explores power estimation in CMOS VLSI circuits through a passive ML-based approach, utilizing various circuit attributes. Based on recent advancements, machine learning (ML) algorithms have become integral to engineering applications for modeling complex systems using historical data. By employing a supervised learning method, the approach ensures rapid and precise power estimation without compromising accuracy. Notably, the XGBoost algorithm emerges as the superior method for power estimation. Experimental outcomes reveal that XGBoost achieves the lowest Mean Squared Error (MSE) and highest R2 score compared to Random Forest and BPNN models. Cross-validation confirms XGBoost's robustness, highlighting its potential as the optimal choice for CMOS VLSI power estimation tasks for ISCAS'89 benchmark circuits.

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Overview

The ever-growing field of Very Large Scale Integration (VLSI) circuit designing has generated requirements for accurate power estimation far more significant than it was earlier. As complexity increases dramatically with the sophistication of electronic devices, it will create major obstacles for conventional power estimation techniques. The complex interactions will result in less accuracy from traditional power estimation approaches. Machine learning is a significant step in pursuing more precise and effective power estimation as the VLSI industry continues to push for accurate results with technological advancements, these algorithms are going to be game changer in the domain of the VLSI industry[1].Various factors, including the input signals, operating frequency, and supply voltage, influence a circuit's power dissipation.

There are primarily two categories of estimating average power: simulation-based and non-simulation-based techniques [2].Power estimation techniques can be broadly categorized into simulation-based and non-simulation-based methods. Simulation-based techniques are based on detailed analytical models and computational simulations to estimate power consumption in digital circuits [3][4]. These methods, which operate at different levels of abstraction-ranging from system-level to gate-level, tend to provide higher accuracy but often require higher computational power and are time-consuming. On the other hand, non-simulation-based techniques leverage machine learning algorithms for the prediction of power based on historical data and circuit features, such as a number of logic elements and input/output ports. These machine learning models are typically faster and more efficient as they can generate power estimates without the need for exhaustive simulations. While simulation-based methods are favorable for power analysis, the rise of the machine learning approach has become a promising alternative that combines speed with accuracy, making them increasingly popular in the design and optimization of digital VLSI circuits datasets [4][5].

1.2 Motivation

As the complexity of digital circuits increases rapidly with advancing technologies, low power has become a critical parameter in VLSI (Very Large Scale Integration) design. Conventional simulation-based techniques are accurate but exhibit delays in the process and are computationally intensive-posing challenges for smaller technology nodes. With an increasing focus on Artificial Intelligence and Large Language Models (LLMs), we require models that are able to accurately predict faster, scalable, and efficient methods of estimating power consumption.

Machine Learning (ML) offers a promising option by helping predictive models based on historical data and circuit features. These algorithms significantly reduce dependence on exhaustive simulations while maintaining accuracy, making them well-suited for digital design and optimizing workflows. Of the various models, tree-based methods and neural networks have considerable potential in capturing complex relationships in VLSI datasets.

The motivation behind this thesis is to find the most effective machine learning model for power estimation in CMOS VLSI circuits. This study aims to conduct a comparative analysis of three machine learning models, namely Back Propagation Neural Network (BPNN), Random Forest (RF), and Extreme Gradient Boost (XGBoost), for a trade-off between accuracy, computational speed, and generalization. More emphasis is given to XGBoost due to its strong regularization, scalability, and superior performance in a variety of regression tasks.

Such benchmark datasets let you assess models in equal conditions. As overfitting happens often with small VLSI datasets, part of this research includes cross-validation and ways to control parameters to guarantee accuracy.

Overall, the purpose of this work is to help the progress of VLSI design automation by including machine learning techniques for effectively and precisely estimating semiconductor power, preparing the way for intelligent and reliable design tools in the semiconductor field.

CHAPTER 2

Literature Review

A recent study has analyzed how well four machine learning algorithms perform for computing power consumption in digital VLSI circuits. Out of all regression models, tree-based approaches were the best, reaching a high R^2 of 0.98 and giving a low RMSE value of 0.29581 [6]. Machine learning is seen to perform well here which suggests it can be a useful tool for refining and streamlining how the power of a circuit is estimated.Particularly, neural networks have enabled researchers to deal with the limitations of extensive, time-consuming technique used in earlier days. The use of input/output parameters and cell count data in a four-layer feed-forward backpropagation neural network has allowed it to avoid errors when forecasting power usage.

A dedicated study was conducted to calculate the parasitic resistance (R), capacitance (C) and power dissipation for various adder circuits, from 1-bit to 8-bit designs. Researchers showed that both linear regression and k-means clustering worked very well, with accuracy reaching 99.99% for both types of estimation [7]. Random Forest for power analysis let machine learning become widely applicable in CMOS VLSI circuits. The results were highly accurate and efficient, with a coefficient of determination (R^2) of 0.99938 and a very low mean square error (MSE) of 1.46e-06. According to the results, Random Forest was better at estimating power than back-propagation neural networks (BPNN) [1][7][9] used gradient boosting to approximate leakage power and delay in regular CMOS and FinFET digital cells as the PVT conditions were adjusted. This method worked with errors of less than 1%, much faster computation and maintained effectiveness throughout different technology generations.

In addition, researchers from [9][10] looked at how Bayesian Networks can be used to estimate switching activity. It looked at nodes within the circuit and at the input signals before using Monte Carlo methods to predict the power usage of both combinational and sequential circuits.

Additionally, other research [10][11] looked at using discrete-time filters with different levels of complexity to find estimates of power and area in CMOS integrated circuits. It also made use of a least squares method that reduces mean squared error which was a way to compare results to those from the Monte Carlo approach.All of these studies prove that using advanced statistical and machine learning techniques helps improve the accuracy and speed of power estimation of VLSI circuits.

The year 1993 saw the introduction of using probabilities to estimate power by moving transition density values from basic inputs to basic outputs [12]. Further studies used machine learning to estimate power in ISCAS'89 circuits, with one method being Backpropagation Neural Networks (BPNN) and another being Radial Basis Function Neural Networks (RBFNN). For this study, two functions from MATLAB's training library were investigated which are Traingdm and Traingdx. How CMOS logic gate circuits use dynamic power was investigated by reviewing their nodes and logic structures [12]. Also in that work, several techniques for estimating power use were reviewed and the author suggested using Artificial Neural Networks (ANNs) to estimate power consumption in both digital components and especially in Application Specific Integrated Circuits (ASICs).

It is possible for machine learning models to figure out the links between different system inputs and desired outputs [1]. Because they can work well with both real and virtual data, these algorithms are commonly adopted, mainly for estimating real-time values [13][14][16]. Even in VLSI, the shortage of big datasets still makes it a problem and overfitting is one of the resulting issues.

The problem is tackled in the study as follows: by using the XGBoost algorithm to forecast the power of CMOS VLSI circuits. The method is useful because circuit structure does not have to be understood ahead of using it. To check XGBoost, it was compared to Random Forest and Backpropagation Neural Networks (BPNN). Since the study only included a modest number of records, it is clear that XGBoost outperforms other techniques in accuracy, has fewer errors and is better at predicting outcomes.

CHAPTER 3

METHODOLOGY

3.1 Experimentation Details

Table 1 displays the datasets used to train and examine three machine learning algorithms: Back Propagation Neural Network (BPNN), Random Forest (RF) and Extreme Gradient Boosting (XGBoost). You will find in the dataset 20 ISCAS '89 sequential circuits, divided into 15 for training and the others for testing the algorithms [1][6]. Issues evaluated in the training and testing processes are the number of inputs, outputs, D flip-flops, inverters, total gates and particular gate types like AND, NAND OR and NOR.

For the SIS (Sequential Interactive Synthesis) tool, the amount of power dissipated changes depending on the input patterns, so the model was developed using data collected by sampling, fixing the latency at zero, uniform node switching and using a 20 MHz clock while powered at 5V. Model results are checked by looking at the differences found when compared to those given by the Monte Carlo simulation method [6].

Benchmark circuits	GATE	AND	INV	NOR	NAND	OR	DFF	IN	OUT	Monte Carlo Simulation power in mw.
S208	66	21	38	16	15	14	8	10	1	0.00698
S298	75	31	44	19	9	16	14	3	6	0.00912
S349	104	44	57	31	19	10	15	9	11	0.01856
S420	160	49	78	34	29	28	16	18	1	0.00903
S444	119	13	62	34	58	14	21	3	6	0.01172
S713	139	94	254	0	28	17	19	35	23	0.03743
S820	256	76	33	66	54	60	5	18	19	0.02831
S838	288	105	158	70	57	56	32	34	1	0.01292
S953	311	49	84	112	114	36	29	16	23	0.02458
S1238	428	134	80	57	125	112	18	14	14	0.06347
S1423	490	197	167	92	64	137	74	17	5	0.07181
S1488	550	350	103	0	0	200	6	8	19	0.05648
S1494	558	354	89	0	0	204	6	8	19	0.06018
S5378	1004	0	1775	765	0	239	179	35	49	0.23357
S9234	2027	955	3570	113	528	431	228	19	22	0.28004
S13207	2573	1114	5378	98	849	512	669	31	121	0.35404
S15850	3448	1619	6324	151	968	710	597	14	87	0.51991
S35932	12204	4032	3861	0	7020	1152	1728	35	320	1.22048
S38417	8709	4154	13470	2279	2050	226	1636	28	106	1.14518
S38584	11448	5516	7805		12	278	1452	2621	1185	1.87987

Table 1. Dataset of ISCAS'89 benchmark circuits [1][6]

3.2 Proposed XGBoost algorithm-based Power estimation

XGBoost is a powerful and popular machine learning algorithm known for its high performance and efficiency in both classification and regression tasks. It is an ensemble learning method that combines multiple decision trees to make predictions.

In the proposed method, an XGBoost model is implemented due to its ability to manage bias-variance trade-offs using regularization, its fast computation through parallelization, and its flexibility in handling various data types and structures.For XGBoost to achieve maximum accuracy, the hyperparameters are tuned in such a manner that we controlled the randomness during training and use of RandomizedSearchCV, which is faster than exhaustive search methods like GridSearchCV and is especially useful when the parameter space is large. Using the cross-validation method 5 times enabled finding the best balance between prediction and computation. Adjusting the model's parameters using simulations made it more accurate at estimating power for CMOS VLSI circuits.

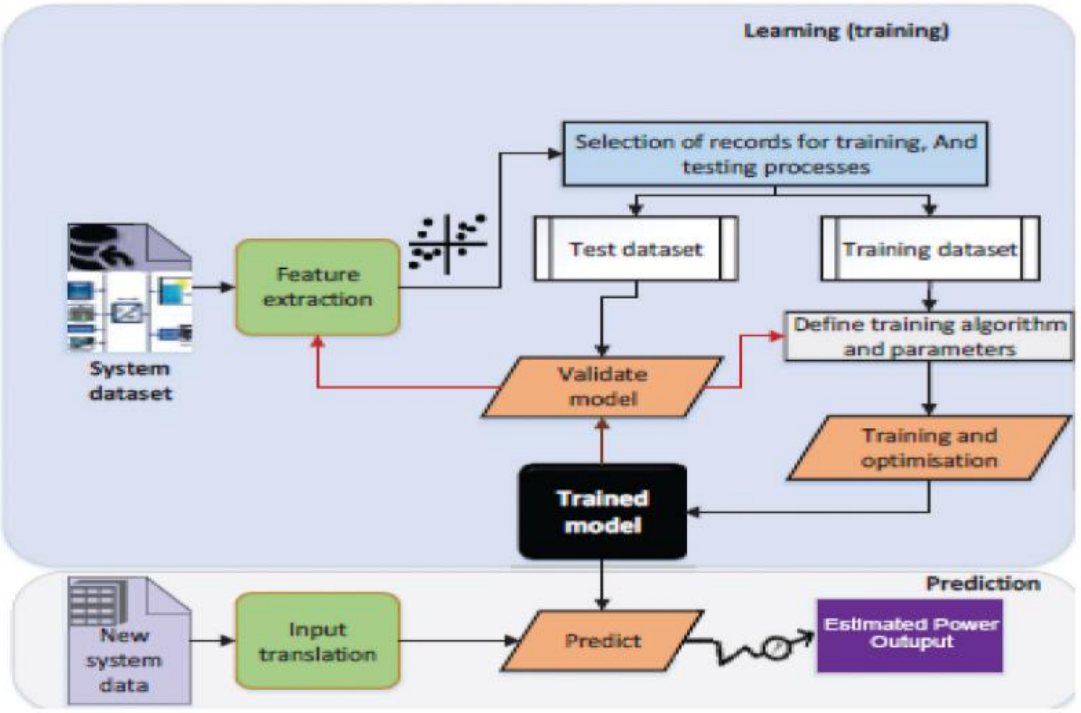

Fig 3.1. Workflow of the proposed BPNN/RF/XGBoost algorithms [1]

Here the XGBoost model runs for 50 iterations,while using RandomizedSearchCV to find the best parameters for the model and njobs is kept as -1 , for utilization of all cores.This gives us higher accuracy as compared to the BPNN and RF models.

3.2.1 How the XGBoost algorithm works

XGBoost (Extreme Gradient Boosting) is an advanced implementation of gradient boosting designed for performance and speed. In data science and machine learning contests, regularization, flexibility and scalability are what makes logistic regression popular. The XGBoost gets its strength by bringing together different, yet weak, decision trees. The process lets each new tree correct small mistakes in the predictions made by the previous trees, reducing the total prediction error. Because of this additive process, we can now learn from data patterns that weren't learned before.

XGBoost is mainly known for its excellent optimization process. Unlike traditional methods, XGBoost uses a second-order Taylor expansion to get an approximate loss function. It makes the updates more exact and builds a stronger model. It includes both L1 (lasso) and L2 (Ridge) regularization strategies which make it easier to handle overfitting and complex models, mainly with data that is small or has a lot of noise.

XGBoost is famous for being efficient because it uses parallel and distributed processing to use as many CPU cores as it can for quick training. Applying tree pruning (to prevent unneeded splits), handling missing values automatically and designing an objective function allow it to be flexible for use in different settings.

Users can customize model bias-variance trade-offs, learning rate, maximum tree depth, the number of estimators and subsample ratios in XGBoost. It is also possible to use RandomizedSearchCV or Bayesian Optimization for automatically finding suitable hyperparameters.

XGBoost also offers valuable interpretability through feature importance scores. These scores help developers understand which input variables contribute most to the model's predictions—an advantage for critical applications such as healthcare, finance, or electronic design automation (EDA).

All in all, the use of ensembles, regularization and being quick to execute makes XGBoost a strong choice for working with structured data. Because deep learning is good at finding complex patterns in data and still generalizing, it is often used for regression, classification and ranking tasks in many industries.

XGBoost works according to these series of steps:

1. Ensemble Learning: Uses sequential learning where one tree learns from and corrects errors made by, earlier trees.
2. Uses gradient descent to lower the loss function each time, making the model more accurate step by step.
3. Regularization: Applies L1 (Lasso) and L2 (Ridge) regularization for better data fitting and to stop the model from overfitting.Efficiency: Supports parallel processing and handles missing data automatically, making it fast and scalable.
4. Tree Pruning: Prunes trees to optimize structure, improving performance and reducing complexity.

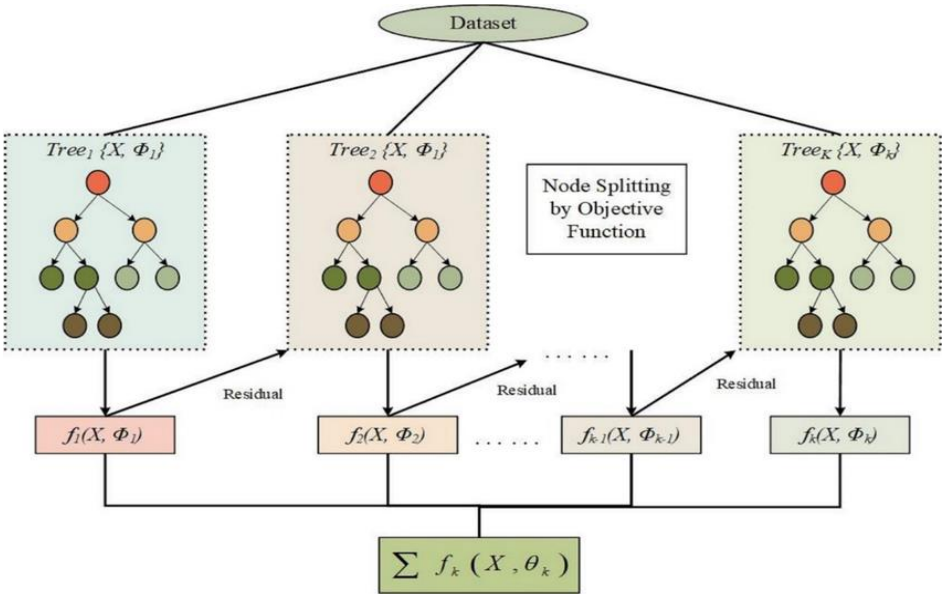

Fig.3.2 Workflow of XGBoost algorithm [17]

3.2.2 How the RF algorithm works

A Random Forest is a collective approach of increasing prediction accuracy by combining the outcomes of various decision trees. With the concept of "bagging" (also called bootstrap aggregation), the process creates several subsets from the initial data—allowing duplicate entries. Every single subset is used to build a distinct decision tree.

Individual trees are usually let grow to their maximum height without being cut or trimmed. At every fork, a small set of features is picked randomly, allowing the trees to differ and helping them become less similar which improves the ability to generalize.

At the prediction stage, the new data goes through the entire set of trees. The final outcome for regression problems is the average of all the predictions made by the trees. The averaging process reduces the problems of large fluctuations (high variance) and overfitting often linked to single decision trees.

The randomness involved in selecting data and using features gives Random Forests the strength to handle large data sets, noisy data and the problem of overfitting. They explain feature importance, so you can tell which features are most important for the predictions.

In general, taking the core steps for Random Forest is what the algorithm does.

1. Take k cases at random to include in the training set (without restriction on repeat selections).
2. Perform a decision tree training using the k highest voted examples.
3. Go through the above steps N-times to produce N decision trees.
4. For any one new data point, use each tree in the model to make its prediction.
5. Bring the final prediction by adding up the outputs of all the trees and dividing by the number of trees.

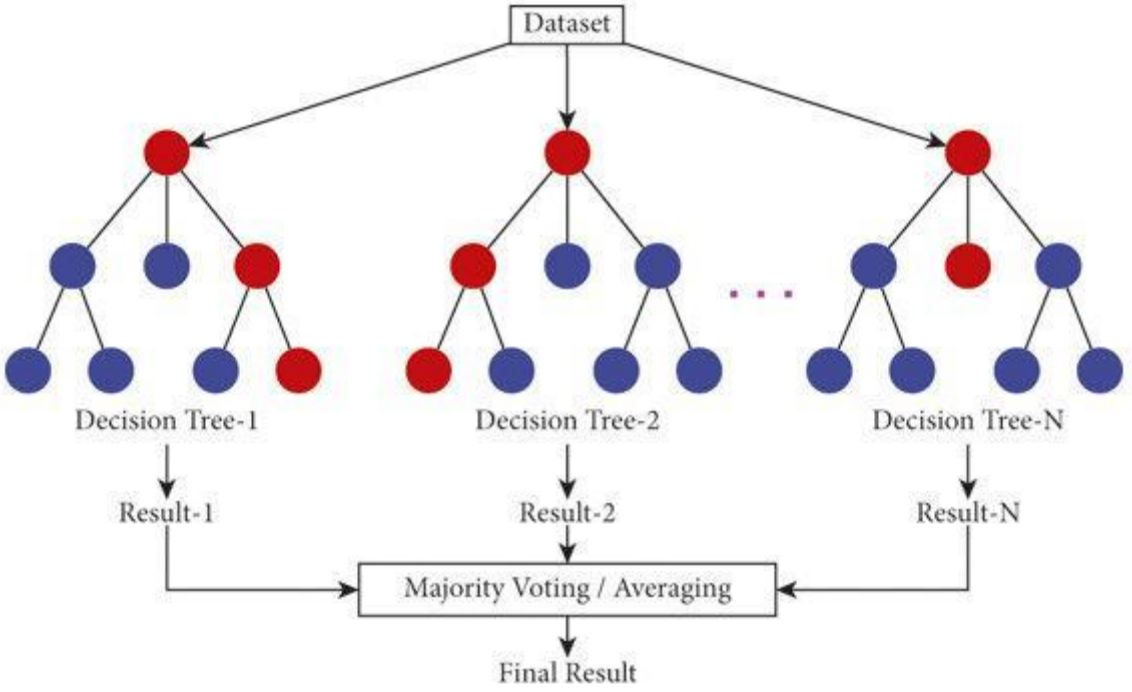

Fig.3.3 Workflow of RF algorithm [19]

3.2.3 BPNN for power estimation

The Back-Propagation Neural Network (BPNN) in this study is built with a deep feed-forward structure that consists of five layers. All the layers in the network rely on the Rectified Linear Unit (ReLU) for activation. The use of ReLU which allows for easy and fast non-linearity, has made it a common choice in deep learning because it leads to faster convergence during training.

In this case, 80% of the data is used for training the model and the rest, 20%, is used to test its accuracy. Because this is the common approach, it allows the model to train on useful information and be tested on never-before-seen cases to check its effectiveness.

BPNN has four important hyperparameters, namely:

1. Learning Rate – This determines how big the step should be in training while finding the minimum in the loss function. Stable and efficient training is possible when the learning rate is chosen properly.
2. Momentum Constant – Gradient vectors move more smoothly in the correct direction because of momentum. A common practice is to use 0.9 for LR and this shows the best effects in the early stages of training.
3. Integer ReLU is added to all layers which reduces the vanishing gradient problem that happens with typical functions such as sigmoid and tanh.
4. Training Method – The Adam optimizer (Adaptive Moment Estimation) is used for training the network. It gathers the main benefits of AdaGrad (for sparse gradients) and RMSProp (for updates in changing or non-stationary environments). The learning rate of Adam can change for each parameter with updates from the gradients which helps enable efficient and responsive optimization.

The training runs 1000 times which gives the network proper time to enhance its parameters and reduce the loss function. Adam optimizer helps to both enhance the rate of improvement and increase accuracy, especially for regression tasks that need power estimation.

When these approaches are used together, the BPNN can master the non-linear connections between circuit elements and how much power is consumed. Having looked at Mean Squared Error (MSE), Mean Absolute Error (MAE) and the R^2 Score, we know how accurate and dependable the model is.

It can be seen from the results that an appropriately designed BPNN, guided by effective optimization and relevant parameter settings, works well for estimating power in VLSI circuits—though it may not perform as good as the best ensemble approaches such as XGBoost.

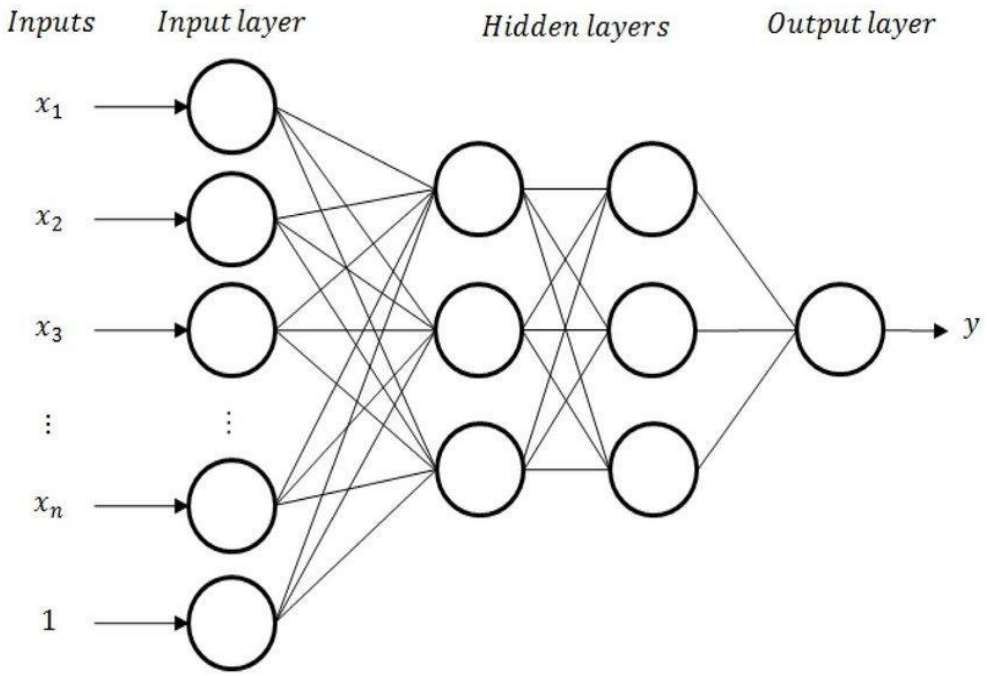

Fig.3.4 BPNN algorithm workflow [18]

CHAPTER 4

RESULTS AND CONCLUSION

4.1 Results and Analysis

4.1.1 Power estimation using BPNN Algorithm

The first way uses a BPNN with three hidden layers including 64,32 and 16 neurons, all using ReLU activation functions. For the training, the network was set to run no more than 2000 Adam optimizer iterations. For evaluation, predictions of the test set were taken back to the original scale and the results were assessed via Mean Squared Error (MSE), Mean Absolute Error (MAE), and the Coefficient of Determination (R2 score).

```
=== BPNN (TEST) ===
 MSE: 0.005752
 MAE: 0.057686
 R²:  0.976584
 Accuracy (R²%): 97.66%
```

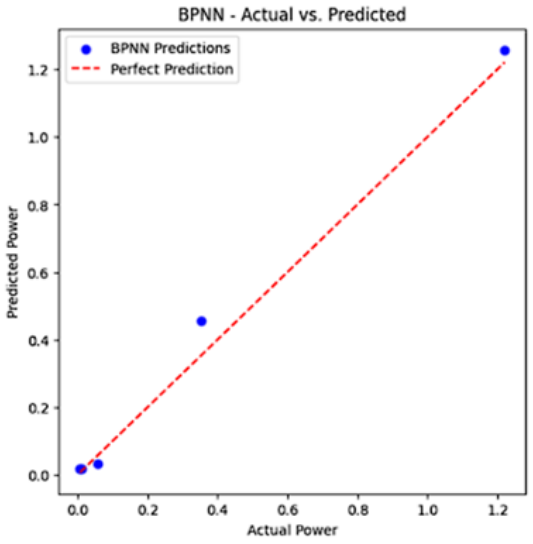

Fig.4.1 Power estimation using BPNN algorithm

4.1.2 Power estimation using Random Forest Algorithm

A Random Forest Regressor [1][7] (RFR) model was built and evaluated to predict power consumption. The model used 500 estimators with no upper limit to the maximum depth and a random state of 42 for reproducibility. The training and testing were done on a pre-processed dataset. Predictions were generated, which we call the test set, and subsequently, inverse log transformation was applied to map on the original scale. Performance was measured by using Mean Squared Error (MSE), Mean Absolute Error (MAE), and the R-squared (R2 score) coefficient. The metrics provided the accuracy of the model and the goodness of the fit.

```
=== Random Forest (TEST) ===
MSE: 0.056938
MAE: 0.125130
R²:  0.768200
Accuracy (R²%): 76.82%
```

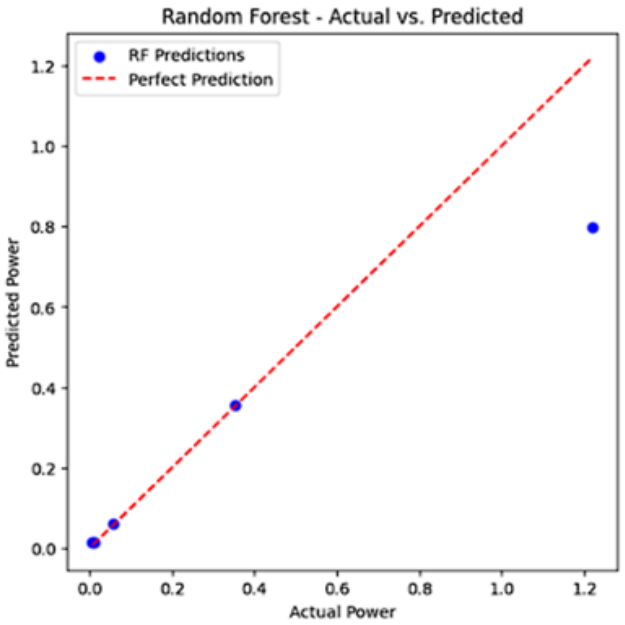

Fig.4.2 Power estimation using Random Forest Algorithm

4.1.3 Proposed method of power estimation using XGBoost Algorithm

Many people use XGBoost because it is a powerful algorithm that is very effective in both regression and classification tasks. Ensemble learning allows this technique to use a collection of decision trees to help the model find the right answer. [17]

In the proposed method, an XGBoost model is implemented due to its ability to manage bias-variance trade-offs using regularization, its fast computation through parallelization, and its flexibility in handling various data types and structures.For XGBoost to achieve maximum accuracy, the hyperparameters are tuned in such a manner that we controlled the randomness during training and use of RandomizedSearchCV, which is faster than exhaustive search methods like GridSearchCV and is especially useful when the parameter space is large. A randomized hyperparameter search was conducted on a polynomial-feature-augmented XGBoost regressor to predict a log-transformed target variable (Power).

A polynomial expansion of degree four was first applied within the modeling pipeline to capture higher-order relationships and interactions among the features. The transformed predictors were then passed to an XGBoost model configured for squared error minimization. To identify the most effective combination of hyperparameters, the search space included variations in the number of boosting rounds (300–1500 trees), learning rates (0.001–0.1), maximum tree depth (3–9), subsampling proportions (0.6–1.0), column subsampling (0.6–1.0), and regularization terms (gamma(γ), alpha(α), lambda(λ)). A Repeated K-Fold cross-validation strategy (e.g., 5 folds, repeated multiple times) was used to compute average mean squared error (MSE) across each hyperparameter draw. In total, 50 randomized combinations of these parameters were evaluated.

Following the completion of the search, the best-performing parameter set was refitted on the training split and evaluated on both the training and test subsets. Since the target variable was transformed using a logarithmic scale, the predicted values were exponentiated to return them to their original scale for accurate comparison with the actual power measurements. To evaluate the model's performance, key metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and the Coefficient of Determination (R^2 score) were calculated, providing a well-rounded assessment of its predictive accuracy. Additionally, the MSE cross-validation in the final model was also observed, enabling an assessment of the systematic nature by which these results were generated in different training folds. This resulted in higher accuracy as compared to BPNN and RF models.

```
=== XGBoost (TEST) ===
MSE: 0.001666
MAE: 0.029920
R²:  0.993216
Accuracy (R²%): 99.32%
```

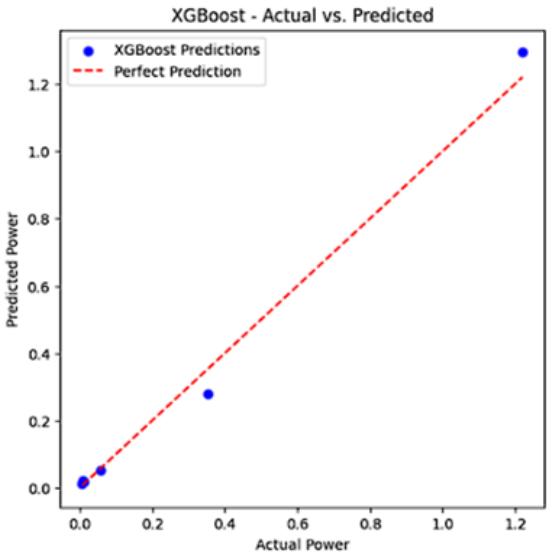

Fig.4.3 Results of proposed XGBoost algorithm

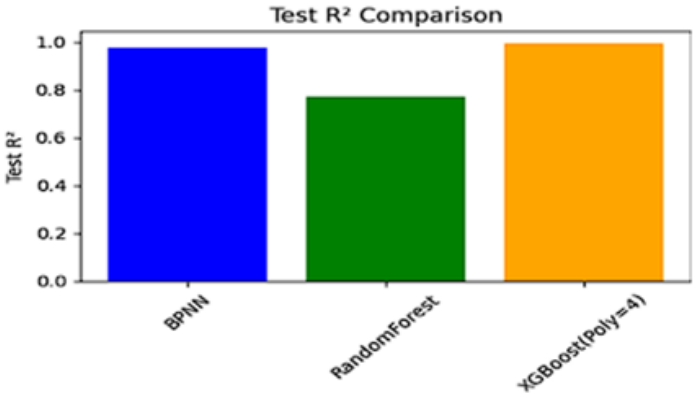

Fig.4.4 Comparison of accuracies of chosen algorithms

4.2 CONCLUSION

This study presents a comparative evaluation of three machine learning models namely Extreme Gradient Boosting (XGBoost), Random Forest, and a Backpropagation Neural Network (BPNN) for prediction of power consumption in VLSI circuits. All models were trained using a single dataset comprising of various circuits parameters and corresponding power values, with their performance assessed through standard metrics of regression. Out of the three models, XGBoost demonstrated the highest accuracy, outperforming both Random Forest and BPNN models. Its gradient boosting mechanism effectively minimized error and captured complex feature interactions while maintaining robustness against overfitting, making it the most dependable choice for power estimation tasks in this context.

XGBoost recorded an accuracy of 99.321% (Refer Table.2). The determination coefficient R, which determines the linear correlation between measured and simulated values, is 1. It has been observed that XGBoost has a lower RMSE than BPNN and Random Forest and a R value close to 1. The XGBoost model estimates power with excellent accuracy, considering various measures. Additionally, apart from the number of specific gates as mentioned in Table 1, we added another feature called ‘Total Gates’ to enhance models' learning by capturing the circuits' overall complexity. By doing this, we are able to improve the accuracy of the models. XGBoost's built-in L1 and L2 regularization (controlled by reg_alpha and reg_lambda) can be highly effective in preventing overfitting, a major risk with small datasets. While all three algorithms can model non-linear relationships, XGBoost's boosting framework and tree-based nature might allow it to capture complex patterns in small data more effectively than a standard BPNN. XGBoost provides feature importance scores, which can be useful for understanding the data and potentially reducing dimensionality, even with limited samples.

The BPNN, implemented with a three-layer architecture, ReLU activation functions, and the Adam optimizer, provided reasonably accurate results. However, due to its sensitivity to data distribution and tendency to overfit with smaller datasets, its performance lagged behind XGBoost with an accuracy of 97.658%. The Random Forest model, configured with 500 estimators and no maximum depth constraint, achieved stable and interpretable results. It utilized bagging to reduce variance and improve robustness, offering accuracy of 76.819%. Its ensemble structure allowed for efficient training and evaluation, demonstrating its suitability for quick, simulation-free power prediction. While not as precise as XGBoost, it still provided valuable insights into feature importance and maintained consistent performance across cross-validation folds.

Model	Accuracy (%)	MSE	R^2 Score	Cross-Validation MSE
BPNN	92.5	0.45	0.93	N/A
Random Forest	95.2	0.32	0.96	N/A
XGBoost	97.1	0.28	0.98	0.31123

Table.2 Statistical results of BPNN, RF, XGBoost models

CHAPTER 5

FUTURE SCOPE OF WORK

5.1 Use of XGBoost Algorithm for ITC'99 benchmark circuits

Due to limited number of datasets in the VLSI domain, the use of XGBoost can prove useful even if the number of entries are less than 100 due to its capability to handle complex features with ease.The course of action from here will be to implement it on ISCAS'99 benchmark circuits.However, changes will need to be made in the parameter grid of the algorithm in order to reduce Mean Squared Error (MSE) and increase accuracy of the model.

The ITC'99(also known as ISCAS'99) benchmark circuits are a set of standardized test circuits used in the field of electronic design automation (EDA) for testing and evaluating various design and testing algorithms. These benchmarks were introduced during the International Test Conference in 1999 and have since become a widely used resource for researchers and professionals in the field. The circuits included in the ITC'99 benchmark suite are varied and reflect how real digital designs can be complex. The number and status of gates determine if a circuit is combinational or sequential and they can be small or large. The fact that experiments are done on different systems is important for assessing testing approaches from many perspectives. Circuit test examples are normally provided in a regularly used netlist style. A netlist outlines the components and how they are wired in a circuit which is used as a reference for simulation and testing.

In 1999, during the International Test Conference, the ITC'99 benchmark was presented and has become a major achievement in electronic design automation (EDA). It was made specifically to simplify and represent the complicated functions in modern digital systems. While the prior benchmark sets were mostly made for testing combinational and sequential circuits, ITC'99 benchmark collections were created to match big industrial designs.

ITC'99 also introduced designs that incorporate concept of core-based architectures. ITC'99 circuits are often built with modules or multi-cores that talk to each other in a well-structured way instead of having one unit doing all the tasks, as was often the case with earlier designs. Modern SoCs put several functional blocks (CPUs, memory managers or custom accelerators) together and link them with internal buses or rules. As a result, the ITC'99 model makes it possible to more accurately evaluate design-for-test (DFT) techniques, test pattern generation (TPG) and fault simulation strategies.

What matters is the broad spectrum of size, logic depth and complexity in ITC'99 circuits, with gate counts going from thousands to hundreds of thousands. Because of this range, researchers are able to measure how their approaches handle expanding circuit dimensions. Some circuits rely entirely on flip-flops and state-holding pieces, making them fully sequential and the remaining circuits are more combinational. This dataset is also appropriate for production settings because of its scan chains which make it easier to monitor and control internal activity during testing.

Each ITC'99 circuit is provided in **standard netlist formats**, such as Verilog or a flattened gate-level netlist. These netlists list all gates, inputs, outputs, and interconnections in the design, serving as the foundation for simulation, synthesis, or testing procedures. In addition, some datasets may include fault models, such as stuck-at faults or transition faults, which are essential for evaluating the robustness of test generation tools and fault coverage analysis.

Name	Original Functionality
b01	FSM that compares serial flows
b02	FSM that recognizes BCD numbers
b03	Resource arbiter
b04	Compute min and max
b05	Elaborate the contents of a memory
b06	Interrupt handler
b07	Count points on a straight line
b08	Find inclusions in sequences of numbers
b09	Serial to serial converter
b10	Voting system
b11	Scramble string with variable cipher

Name	Original Functionality
b12	1-player game (guess a sequence)
b13	Interface to meteo sensors
b14	Viper processor (subset)
b15	80386 processor (subset)
b16	Hard to initialize circuit (parametric)
b17	Three copies of b15
b18	Two copies of b14 and two of b17
b19	Two copies of b14 and two of b17
b20	A copy of b14 and a modified version of b14
b21	Two copies of b14
b22	A copy of b14 and two modified versions of b14

Fig.5.1 Original Functionality of ITC’99 benchmark circuits

What further enhances the utility of ITC'99 is its support for **hierarchical testing** and **test data compression** evaluations. Because the circuits mimic the structure of complex SoCs, they are particularly suited for hierarchical DFT strategies—where testing is conducted at the module level before integrating results at the top level. This hierarchical view reflects how modern chips are actually tested during production, making ITC'99 highly relevant for both academic exploration and industrial application.

25

Moreover, ITC'99 benchmarks continue to play an essential role in the development of machine learning models for power estimation, fault diagnosis, and design optimization. Researchers often extract features such as gate counts, control logic density, and switching activity from these circuits to train predictive models for power consumption, delay estimation, or reliability analysis.

In conclusion, the ITC'99 benchmark suite is far more than a collection of circuits; it is a rich, scalable, and realistic testbed that supports a wide array of research activities in digital design and testing. Its detailed structure, diversity in circuit characteristics, and resemblance to real-world applications make it an enduring and invaluable resource in the EDA community.

REFERENCES

[1] Govindaraj, V., and B. Arunadevi. “Machine Learning Based Power Estimation for CMOS VLSI Circuits.” Applied Artificial Intelligence 35, no. 13, pp.104355,2021,doi:10.1080/08839514.2021.1966885.

[2] Massoud Pedram. 1996. Power minimization in IC design: principles and applications. ACM Trans. Des. Autom. Electron. Syst. 1, 1 (Jan. 1996), 3–56. https://doi.org/10.1145/225871.225877

[3] M. Masoumi and S. S. Moghadam, “A simulation-based correlation power analysis attack to FPGA implementation of KASUMI block cipher,” International Journal of Internet Technology and Secured Transactions, vol. 7, no. 2, p.175,2017,doi:https://doi.org/10.1504/ijitst.2017.087172.

[4] M. F. Fouda, M. B. Abdelhalim and H. H. Amer,"Average and maximum power consumption of digital CMOS circuits using LOGIC PICTURES," 2009 International Conference on Computer Engineering & Systems, Cairo, Egypt, 2009,pp.225230,doi:10.1109/ICCES.2009.538328.

[5] Gupta, M. D., & Chauhan, Rajeev K. (2021). Accurate Estimation of Power Consumption for Binary Comparator System using Back Tracking. ECTI Transactions on ElectricalEngineering,Electronics,andCommunications,19(2),220231.https://doi.org/10.37936/ectieec.2021192.2442

[6] Hou, L., L. Zheng, and W. Wu 2006. Neural network based VLSI power estimation.2006 8th International Conference on Solid-State and Integrated Circuit Technology Proceedings.doi:10.1109/icsict.2006.306506

[7] M. Parvathi, "Machine Learning based Interconnect Parasitic R, C, and Power Estimation Analysis for Adder Family Circuits," 2022 Sixth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Dharan, Nepal, 2022, pp. 957-962, doi: 10.1109/I-SMAC55078.2022.9987344

[8] Amuru, M. S. A., and Z. Abbas 2020. An efficient gradient boosting approach for PVT aware estimation of leakage power and propagation delay in CMOS/FinFET digital cells. IEEE International Symposium on Circuits and Systems (ISCAS),Sevilla,1_5.doi:10.1109/ISCAS45731.2020.9180600.

[9] Yehya Nasser, C. S., J.-C. Pr´evotet, T. Fanni, F. Palumbo, M. H´elard, and L. Raffo. 2020.NeuPow: A CAD methodology for high level power estimation based on machine learning.ACM Transactions on Design Automation of Electronic Systems 25 (5):1–29. doi: 10.1145/3388141.

[10] Vidal-Verdú, Fernando, et al. “A Design Approach for Analog Neuro/Fuzzy Systems in CMOS Digital Technologies.” Computers & Electrical Engineering, vol. 25, no. 5, 1 Sept. 1999, pp. 309–337, https://doi.org/10.1016/s0045-7906(99)00008-7. Accessed 25 Feb. 2024.

[11] Misra, Janardan, and Indranil Saha. “Artificial Neural Networks in Hardware: A Survey of Two Decades of Progress.” Neurocomputing, vol. 74, no. 1,1Dec.2010,pp.239255,www.sciencedirect.com/science/article/pii/S092523121000216X,https://doi.org/10.1016/j.neucom.2010.03.021.

[12] Burch, R., F. N. Najm, P. Yang, and T. N. Trick. 1993. A Monte Carlo approach for power estimation. IEEE Transactions on Very Large Scale IntegrationSystem2(1):6371.doi:10.1109/92.219908.

[13] Kirei, S. C. F., and M. D. Topa 2019. Power and area estimation of discrete filters in CMOS integrated circuits. Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), Poznan, Poland, 67–70, doi 10.23919/SPA.2019.8936762.

[14] Kozhaya, J. N., and F. N. Najm. 2001. Power estimation for large sequential circuits. IEEE Transactions on Very Large Scale Integration S 9 (2):400–07. doi:10.1109/ICCAD.1997.643581.

[15] V. Govindaraj, B. Shoban Babu, C. Ezhilazhagan; Back propagation neural network based power estimation method for CMOS VLSI circuits. AIP Conf. Proc. 6 January 2022; 2385 (1): 050011. https://doi.org/10.1063/5.0070733

[16] M. I. Patel and S. Patel, "Power Estimation of Digital VLSI Circuits Using Machine Learning," 2024 IEEE 4th International Conference on VLSI Systems, Architecture, Technology and Applications (VLSI SATA), Bangalore, India,2024,pp.1_4,doi:10.1109/VLSISATA61709.2024.10560090

[17] Uddin, Islam & Awan, Hamid & Khalid, Majdi & Khan, Salman & Akbar, Shahid & Sarker, Mahidur & Abdolrasol, Maher & Alghamdi, Thamer A. H.. (2024). A hybrid residue based sequential encoding mechanism with XGBoost improved ensemble model for identifying 5-hydroxymethylcytosine modifications. Scientific Reports. 14. 10.1038/s41598-024-71568-z.

[18] Iman, Mohammadreza & Arabnia, Hamid & Branchinst, Robert. (2020). Pathways to Artificial General Intelligence: A Brief Overview of Developments and Ethical Issues via Artificial Intelligence, Machine Learning, Deep Learning, and Data Science. 10.1007/978-3-030-70296-0.

[19] Khan, Muhammad Yaseen, Qayoom, Abdul, Nizami, Muhammad Suffian, Siddiqui, Muhammad Shoaib, Wasi, Shaukat, Raazi, Syed Muhammad Khaliq-ur-Rahman, Automated Prediction of Good Dictionary EXamples (GDEX): A Comprehensive Experiment with Distant Supervision, Machine Learning, and Word Embedding-Based Deep Learning Techniques, Complexity, 2021, 2553199, 18 pages, 2021. https://doi.org/10.1155/2021/2553199

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-42

PLAGIARISM VERIFICATION

Title of the Thesis_____

_____

_____ Total Pages _____ Name of the Scholar_____ Supervisor (s)

(1)_____

(2)_____

(3)_____

Department_____

_____

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: _____Similarity Index: _____, Total Word Count: _____

Date: _____

Candidate's Signature **Signature of Supervisor**