SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V13I9P111

Volume 13 Issue 9, 181-194, September 2026

Original Article

A Methodical Comparison of Random Forest and Neural Network Models to Forecast Digital Design Behaviour

Surajit Bhattacherjee[1], Dipankar Pal[2]

[1,2]*Department of Electrical and Electronics Engineering, Birla Institute of Technology and Science, Pilani, K. K. Birla Goa Campus, Goa, India.*

[1]*Corresponding Author : p20190005@goa.bits-pilani.ac.in*

Received: 01 December 2025 Revised: 01 August 2026 Accepted: 11 August 2026 Published: 29 September 2026

Abstract - *Predicting design behaviour in digital systems is vital to optimize electronic design automation workflows and help reduce time-to-market for integrated circuits. Traditional System Verilog and UVM-based methods demonstrate functional correctness while lacking predictive insights to performance metrics prior to synthesis, resulting in expensive design iterations when the timing or power constraints are violated. In this paper, we compare Random Forest (RF) and Neural Network (NN) models for predicting digital design behaviour at the pre-synthesis stage by timing, power consumption and resource utilization. The main novelty is in predictive analytics for proactive design optimization with RTL features. The 2 models were trained and tested in 1,000 synthesis runs on various RTL designs. The RF used 200 decision trees with optimized hyperparameters, whereas the NN was based on five layers with dropout regularization. Performance metrics such as Mean Absolute Error, Root Mean Square Error and R^2 scores were obtained. RF shows accuracy for timing prediction and robustness over noisy inputs, while NN is excellent when modeling intricate non-linear relationships. However, its training time is much longer. Analysis of feature importance demonstrated clock frequency, technology node, etc. with the most influential predictors. The comparisons guide the choice of suitable machine learning methods by analyzing trade-offs between interpretability, computational cost and predictive performance.*

Keywords - *AI-Driven design verification, Hardware performance prediction, ML-based behavioral analysis, Predictive design analytics, RTL synthesis forecasting.*

1. Introduction

Modern integrated circuit development generates massive volumes of synthesis data. Each design iteration produces timing, power and resource metrics. Designers rely on computationally expensive synthesis runs to explore design spaces. A single complex design may require thousands of iterations. This process consumes weeks and months of engineering time [1]. Reducing this iteration time has therefore become a critical challenge in contemporary Electronic Design Automation (EDA) workflows.Traditional System Verilog (SV) [2] and Universal Verification Methodology (UVM)-based verification [3] validate functional correctness. These approaches employ constrained random stimulus and coverage-driven verification [4]. Consequently, design issues are discovered late in the flow, forcing repeated cycles of synthesis, analysis and refinement [1]. This post-synthesis validation paradigm introduces substantial delays and limits the efficiency of design space exploration. Machine learning enables predictive design analysis. Predictive models forecast design behaviour in seconds. This enables rapid design space exploration. A shift from post-synthesis validation to pre-synthesis prediction has the potential to fundamentally transform EDA workflows by reducing overall design turnaround time from days to hours [5]. Figure 1 illustrates this transformation, showing how ML prediction reduces total design time from days to hours. Two machine learning approaches have been found suitable for predicting design behaviour based on regression tasks [6]. Random Forest builds multiple decision trees on bootstrapped data. Final predictions aggregate individual tree outputs. This provides robustness against overfitting [7], whereas a neural network builds a layered architecture. It discovers hierarchical feature representations automatically and outperforms at capturing non-linear patterns [8]. However, this expressive power often comes at the cost of increased data requirements, longer training times and reduced interpretability. Figure 2 shows these architectural differences, contrasting the tree-based ensemble approach with the layered neural structure.

Unlike prior studies that evaluate machine learning models in isolation or on narrowly scoped datasets, this work provides a unified, application-driven comparison of Random Forest and Neural Network models under identical experimental conditions. Existing work mostly describes

prediction accuracy without assessing the scalability and sensitivity of the dataset to features, the robustness against noisy synthesis data, or the feasibility of deployment in industrial EDA flows [9, 10]

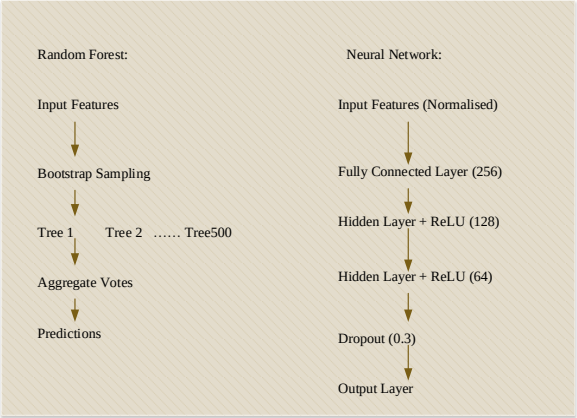

Fig. 1. Traditional versus ML-enhanced workflows demonstrating time savings

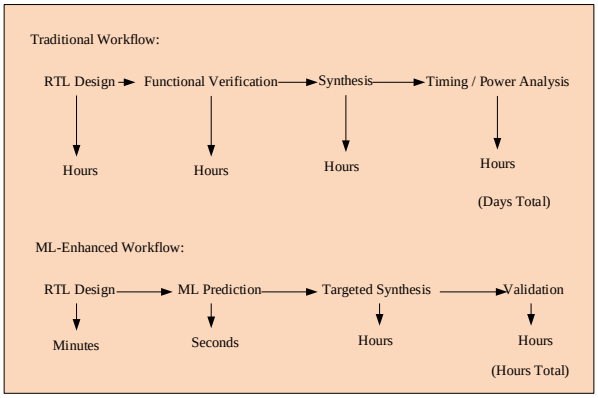

Fig. 2 Architectural comparison showing ensemble versus deep learning structures

In contrast, this paper jointly addresses accuracy, interpretability, computational cost and robustness across multiple RTL design classes, which can be utilized in a real-world digital design environment to direct model choice. Current literature frequently emphasizes only certain metrics and/or single-model assessments, which cause designers a lack of guidance on how to make model selection decisions in industrial designs. Hence, the gaps in this research are filled by providing a systematic experimental comparison of Random Forest and Neural Network models for timing and power prediction in digital IC design. Thus, the proposed work fills in the gaps of experimental comparison. Here, we focus on exploring five research questions in depth:

1. Which model achieves higher accuracy for timing and power estimation?
2. How do training requirements differ?
3. Which RTL parameters mostly influence predictions?
4. How does each approach handle noisy data?
5. What are practical implications for industrial deployment?

This paper makes three key contributions beyond existing work. First, it introduces a reproducible and fair comparison framework that evaluates Random Forest and Neural Network models under identical datasets, feature sets, and synthesis conditions. Second, it quantitatively analyzes trade-offs among prediction accuracy, interpretability, training complexity and inference cost, whereas existing research primarily focuses on accuracy alone. Third, it identifies design and data-dependent application scenarios where each model is preferable, enabling practical adoption in industrial EDA workflows rather than pure academic evaluation.

Therefore, the rest of the paper is organized as follows: Section II details the ML model-based selection strategy in digital design verification, Section III discusses the research gap and motivation, Section IV defines the proposed methodology and experimental implementation technique, Section V presents the results, and Section VI concludes the paper.

2. ML Model Selection Rationale

Random Forest (RF) and Neural Networks (NN) are found suitable for forecasting digital design behavior due to their unique strengths. Both methods are renowned for their effectiveness in regression tasks, making them particularly adept at predicting power consumption and achieving timing closure.

2.1. Random Forest: The Ensemble Learning

Random Forest is an advanced ensemble learning method. Its performance in generalization is better because of bootstrap aggregating and random feature selection [11]. The robustness in the method against overfitting is due to several mechanisms. Bootstrap sampling can give trees diversity. Random feature selection prevents single features from dominating forecasts.

Ensemble voting reduces variance by approximately $1/\sqrt{N}$ where N is the number of trees [12]. Recent comparative studies show the advantage of RF against overfitting compared to a single decision tree [12, 13]. Such robustness is critical for design-automation applications that need to generalize over diverse architectures of RTL.

Power consumption and timing prediction gets benefitted specifically from RF's recursive partitioning approach. Power dissipation exhibits semi-deterministic characteristics. The relationship between circuit structure (gate count, capacitance) and power consumption can be represented by tree-based partitioning [10-12]. In a related recent effort, Deligiannis et al. [14] presented a flow of reliability evaluation that quantifies how the faults on permanent hardware affect

the integer arithmetic circuits, further demonstrating how structural circuit characteristics can drive fast and simulation-efficient design-quality assessment.

For the same reason, the timing closure prediction takes advantage of the RF's ability to separate design components into homogeneous delay classes according to structural characteristics. The approach includes native feature-importance calculations. This allows designers to understand which variables (clock frequency, design complexity, technology node and so on) impact the predictions more. Interpretability is essential for design-automation tools [4].

Another critical advantage is computational efficiency. There is linear scaling of RF inference with tree number. Logarithmic scaling is also demonstrated with the dataset size. Typical inference times are between 1-5 milliseconds [12]. Moderate-sized datasets are also simple to model as they take only minutes for training. It allows quarterly updating of the model as design methodologies evolve. This efficiency facilitates real-time design space exploration and allows thousands of candidate designs to be rapidly evaluated [10].

2.2. Neural Network: The Deep Learning

Neural Networks have their main advantages with respect to the realization of complex non-linear relationships. They do so via hierarchical feature learning. Deep learning theory states that NNs learn nested hierarchies. Each layer transforms inputs into progressively more abstract features.

For digital design prediction, automatic discovery of feature interactions can be achieved through this method. Traditional approaches would otherwise require extensive manual engineering for such discovery. Hierarchical learning is considered important because high-level design behaviour arises from complex interactions among low-level characteristics [16].

Universal approximation has a crucial place in the world of design prediction. NNs with multiple hidden layers and non-linear activation functions can model arbitrary non-linear functions with bounded error [8-17]. Design relationships involve multiplicative interactions such as power scaling with frequency and activity. They include exponential dependencies like leakage power, temperature scaling, etc. Boolean logic effects from multiplexer configurations add further complexity. Empirical evaluations demonstrate that NNs substantially outperform linear and tree-based methods when the targets depend on continuous non-linear transformations [17, 18]. Modern digital designs incorporate iterative structures. Examples include encryption rounds and pipeline stages, etc. State machines and memory controllers are typical examples of sequential logic. NNs can learn state-dependent relationships using trained feature transformations [9-19]. This can be of great advantage in real-time modeling in the case of AES cores, memory controllers, and other similar sequential designs. The performance cannot be achieved with the assumption of a static feature relationship. Neural Networks learn complex feature interactions based on hidden layer weights. They easily learn non-linear combinations which predict outputs effectively. Due to independent feature space partitioning, decision trees are unable to replicate this effect [16-18].

2.3. Limitations of other Models

There are some alternative regression methods employed for digital design prediction. However, these approaches have their own limitations. RBF kernels, along with Support Vector Regression (SVR), can address non-linearities as well [20]. Nevertheless, SVR needs a considerable amount of hyperparameter tuning, such as kernel parameters and regularization constants [20]. SVR has poor scalability to high-dimensional feature spaces, often exceeding 50 features [21]. As a result, SVR cannot cater for the 47-dimensional design feature space used in this work.

Gradient Boosting algorithms, such as XGBoost, have performance benefits over Random Forest by improving the performance incrementally [22]. However, such methods are costly and require the tuning of hyperparameters [22]. In real-time inference, XGBoost is computationally expensive [23]. Design automation tools require fast evaluation, so XGBoost is not a candidate to explore interactive design. For example, the non-linear and multi-modal relationships in the behaviour of the design cannot be captured by linear regression and polynomial regression [17]. These simple methods yield very high prediction errors.

3. Research Gap and Motivation

The summary presented in Table 1 consolidates all prior research efforts aimed at developing AI/ML-based solutions to address the challenges discussed in the preceding sections. In addition, the table critically evaluates the corresponding research gaps identified in each study. Table 1 focuses on recent studies that apply predictive analytics for proactive design optimization directly at the Register-Transfer Level (RTL). These include pre-synthesis PPA (power, performance, area) estimation frameworks [5, 24, 25], deep-learning-based RTL timing evaluation [26, 27], RF/NN-based power estimation of digital circuits [28, 29], reliability-oriented evaluation flows for arithmetic circuits [14], and consolidating surveys of machine learning for CAD and circuit foundation models [10, 19]. For each study, the table also records the reported results for the data-driven comparative analysis. The systematic review of machine learning methods for regression tasks identifies Random Forest and Neural Networks as the two most balanced approaches: Random Forest provides robustness, interpretability, and computational efficiency with minimal hyperparameter tuning whereas, Neural Networks provide superior non-linear learning and feature interaction discovery at the cost of increased complexity. In the following section,

proposed methodology and experimental setup is explained with reference to 3 digital design modules for the proof of concept. These are:-

- 32-bit Arithmetic Logic Unit (ALU)
- Dual-Port SRAM Controller
- Advanced Encryption Standard (AES) Core

Table 1. Comparison of the existing methodologies and approaches

Authors (Year)	Model / Approach	Application (Design Level)	Reported Results	Limitations / Research Gap
Rapp et al. [10] (2022)	Survey of ML for CAD (MLCAD)	ML across the complete EDA flow	Consolidates reported accuracy/speed-up figures across all EDA stages	Survey only; no controlled cross-model experimental comparison
Xu et al. [24] (2022)	Deep learning synthesis predictor (SNS)	RTL-stage prediction of area, power and timing	Average RRSE: 0.4998; 2-3 folds faster than the commercial logic synthesis	Whole-design estimates only; no noise-robustness or interpretability study
Sengupta et al. [25] (2022)	ML regression on RTL features	Post-synthesis quality-of-result (TNS/WNS) prediction from RTL	Whole-design TNS/WNS estimation directly at the RTL stage	Limited generalization to unseen designs; power not modeled
Lopera et al. [27] (2022)	Neural Network	Early RTL delay prediction	$R^2 \approx 0.92$ for RTL delay estimation	Small combinational designs; timing only
Fang et al. [5] (2023)	Bit-level simple-operator graph + ML (MasterRTL)	Transferable estimation of pre-synthesis PPA for RTL designs	Correlation gains: +0.33 (TNS), +0.22 (WNS) and +0.15 (power) over prior state of the art across 90 designs	Requires a large cross-design training corpus; robustness to noisy synthesis data not studied
Fang et al. [26] (2024)	Ensemble RTL representations + ML (RTL-Timer)	Fine-grained register-level slack prediction at RTL	Average correlation > 0.89 on unseen designs; ≈3% WNS and ≈10% TNS improvement when guiding synthesis	Timing only; power and resource metrics not covered
Emir et al. [29] (2024)	Deep learning	Power and delay modeling of GNRFET logic circuits	High task-specific prediction accuracy for GNRFET gates	Device/technology specific; not standard CMOS RTL design flows
Deligiannis et al. [14] (2025)	Simulation-based reliability evaluation flow	Permanent hardware defects on logic circuits	Exhaustive fault-impact characterization of adder and multiplier architectures	Reliability-oriented; does not address pre-synthesis timing/power prediction.
Gowrishankar et al. [28] (2025)	Random Forest and BPNN	Intelligent power estimation of digital circuits	$R^2 \approx 0.90$-0.94 for dynamic power prediction	Power only; no timing/WNS prediction and no deployment analysis
Wenji Fang et al. [19] (2025)	Circuit foundation models (survey)	Large-scale models (AI) for VLSI design and static EDA tools	Consolidates emerging root-model results for application-specific EDAs	Heavyweight models; not lightweight pre-synthesis predictors

The state-of-the-art results consolidated in Table 1 remain fragmented across metrics and design levels. SNS reports an average RRSE of 0.4998 for area, power and timing prediction while running two to three orders of magnitude faster than a commercial synthesis flow [24]. MasterRTL improves TNS, WNS and power correlation by 0.33, 0.22 and 0.15, respectively, over prior solutions across 90 designs [5], and RTL-Timer attains an average register-level correlation above 0.89 on unseen designs, translating into approximately 3% WNS and 10% TNS improvement when used to guide synthesis [26]. Among power-focused predictors, RF and BPNN models report R^2 values of approximately 0.90-0.94 for dynamic power [28], while NN-based RTL delay predictors report $R^2 \approx 0.92$ on small combinational designs [27]. However, no single study jointly reports timing, dynamic power and leakage power accuracy together with training cost, inference latency, interpretability and robustness to noisy synthesis data under identical experimental conditions. By contrast, the framework proposed in this work attains R^2 values of 0.954-0.968 across all three targets (RF:

0.954/0.939/0.962 and NN: 0.963/0.950/0.968 for dynamic power, leakage power and WNS, respectively) with inference latencies of 45-82 µs per sample, evaluated on 1,000 synthesized design variants spanning multi-corner, noisy operating conditions. Furthermore, though the prior researches have established that machine learning models can predict post-synthesis timing and power metrics with reasonable accuracy, existing studies largely remain fragmented in scope and evaluation of methodology. Most works focus on a single learning paradigm, assess performance using isolated accuracy metrics or rely on narrowly scoped datasets and idealized synthesis conditions. As a result, the issue of the practical applications of model choice to the real-world EDA processes is under-researched.

This work is novel due to its integrated, regulated, and practically motivated comparison of the model prediction between Random Forest and Neural Networks. In contrast to the literature, the work compares the two models using the same datasets, feature representations and synthesis settings, which allows a more fair and reproducible comparison of the strengths and limitations of the two models. Moreover, this work has gone beyond accuracy as an evaluation criterion, with an analysis of interpretability, computational complexity, training needs, inference latency, and sensitivity to noisy synthesis samples conducted together. Inclusion of noisy synthesis conditions is also realistic of industrial environments, which are seldom considered in the previous literature. The proposed framework also scales across different design classes by testing multiple design and the sequential complexity of RTL design modules. Most importantly, the work offers practicable advice to practitioners that identifies the design and data-sensitive situations where the Random Forest or the Neural Network models are a better fit. This changes the contribution not to a benchmarking of academic performance but to a practical implementation of industrial EDA processes, hence closing a vital rift between machine learning research and automated digital design.

4. Proposed Methodology and Experimental Setup

The proposed methodology establishes a systematic framework for comparing Random Forest (RF) and Neural Network (NN) models to predict digital design behavior. The evaluation is conducted across three representative design modules with varying complexity levels. Each module represents distinct design paradigms commonly encountered in system-on-chip development. The framework encompasses data collection, feature extraction, model training, and comprehensive performance evaluation.

4.1. Design Module specifications

4.1.1. 32-bit Arithmetic Logic Unit (ALU)

The first design module is a 32-bit Arithmetic Logic Unit implementing fundamental operations. The module executes ADD, SUB, AND, OR, XOR, and MUL operations. The design comprises 200 lines of synthesizable Verilog code. It exhibits low-to-medium complexity with 45 arithmetic operations. The critical path consists of 8 logic levels during multiplication. Input and output ports are standardized at 32 bits. A 4-level multiplexer tree selects among operation results. The design incorporates 2 registers for result storage.

4.1.2. Dual-Port SRAM Controller

The second design module is a dual-port SRAM controller managing memory operations. The controller supports 1024-word by 32-bit memory with simultaneous read/write capability. The design consists of 800 lines of Verilog code. It exhibits medium-to-high complexity due to sequential logic and arbitration. The controller implements a 5-state finite state machine. It contains 12 multiplexer configurations with 32-to-1 selection. The design incorporates 18 registers for address, data, and control paths. Read/write arbitration logic prevents port conflicts.

4.1.3. Advanced Encryption Standard (AES-128) Core

The third design module is an AES-128 cryptographic core implementing block encryption. The design supports 128-bit block encryption with a 128-bit key. It comprises 2500 lines of synthesizable Verilog. The core processes data through 10 encryption rounds. Each round contains four transformation layers: SubBytes, ShiftRows, MixColumns, and AddRoundKey. The design implements 16 parallel S-box instances. It includes 4 MixColumns operations per round. Galois Field multiplication operations are capped to 64 per round. The state array employs 128 registers. Table 2 represents the design specification of AES-128 core [30].

For the sake of brevity, in this paper, we have discussed the proposed methodology and implementation with reference to the third design module - that is AES-128 core.

Table 2. AES-128 core design specification

Parameter	Specification
RTL Lines of Code	2500 lines (Verilog)
Block Size	128 bits
Key Length	128 bits
Encryption Rounds	10
S-box Instances	16 (parallel)
MixColumns/Round	4
GF Multiply/Round	64
Register Count	128
Timing Target	5.0 ns (200 MHz)
Power Budget	150 mW @ 1.2V, 25°C
Area Constraint	3.5 mm² (45nm)

This work proposes a dual machine learning approach for power estimation and timing closure prediction in AES-128 cryptographic core design. The methodology employs Random Forest (RF) regression and Neural Network (NN)

models to predict dynamic power, leakage power and Worst Negative Slack (WNS). Figure 3 illustrates the complete methodology flow. The process begins with the AES-128 RTL design implementation. The AES-128 core performs symmetric encryption using a 128-bit key. The design executes 10 transformation rounds, each consisting of SubBytes, ShiftRows, MixColumns, and AddRoundKey operations. The RTL code is synthesized using a standard EDA tool [31]. Static Timing Analysis (STA) is performed to extract timing characteristics. Power analysis tools generate ground truth data for model training. Once the desired dataset is available, the feature extraction is performed.

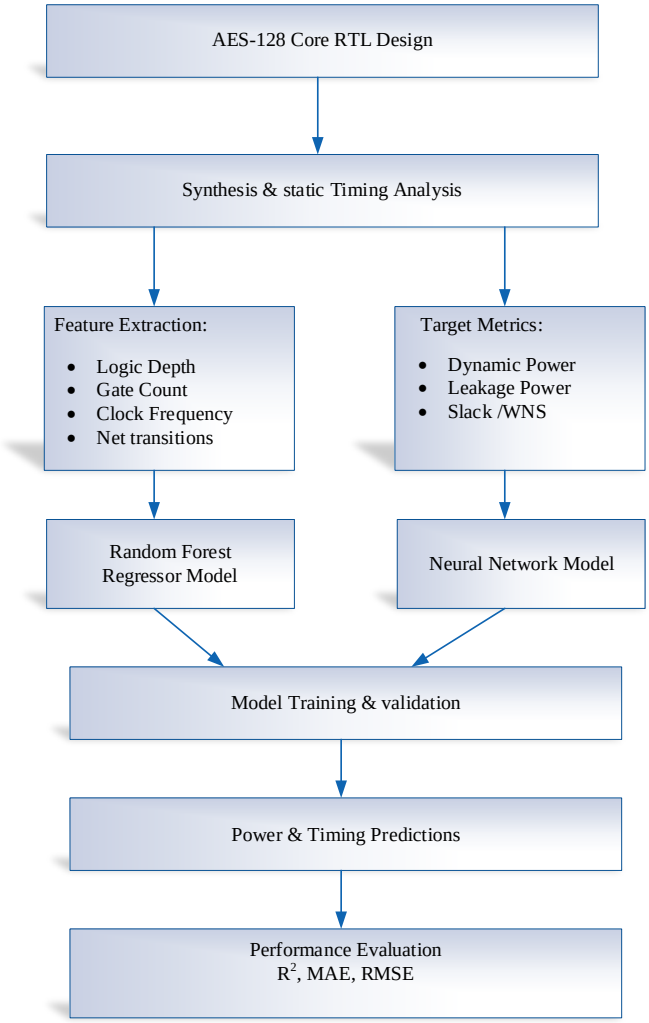

Fig. 3 Proposed methodology with RF and NN model

4.2. Feature Extraction

Figure 4 illustrates the feature extraction pipeline for the AES-128 core design. The process begins with the synthesized AES-128 core, which implements 128-bit encryption using 10 transformation rounds. Synthesis and timing analysis reports are used to extract the four different feature categories. Among these Structural Features, gate count, logic depth, flip-flop count and combinational cells describe the physical design composition.

The timing features, such as clock frequency, path delays, critical paths and setup time, represent the temporal characteristics. Activity features are associated with switching events, toggles, transitions and utilization to characterize dynamic behaviour during operation. Physical features refer to area, routing complexity, congestion, and wire length that describe layout properties. These features, extracted from the four classes, are aggregated into an 8-dimensional feature vector $X = [X_1, X_2, X_3,..., X_8]$.

This normalized feature vector acts as the input for Random Forest and Neural Network Models. The models are trained using historical data, and the prediction is done on three important outputs - dynamic power consumption, leakage power, and Worst Negative Slack (WNS). This structured extraction-aggregation scheme provides fast estimation of power and timing without the need for time-consuming post-synthesis analysis.

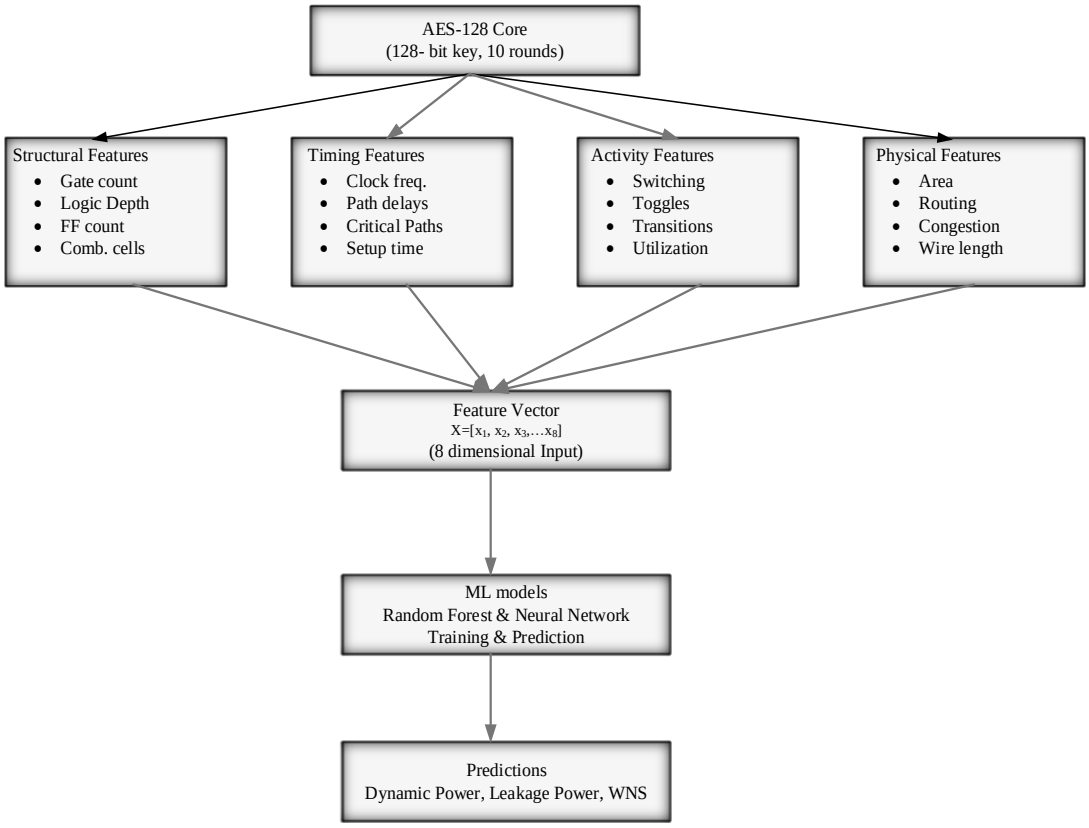

Fig. 4 AES-128 core feature extraction pipeline

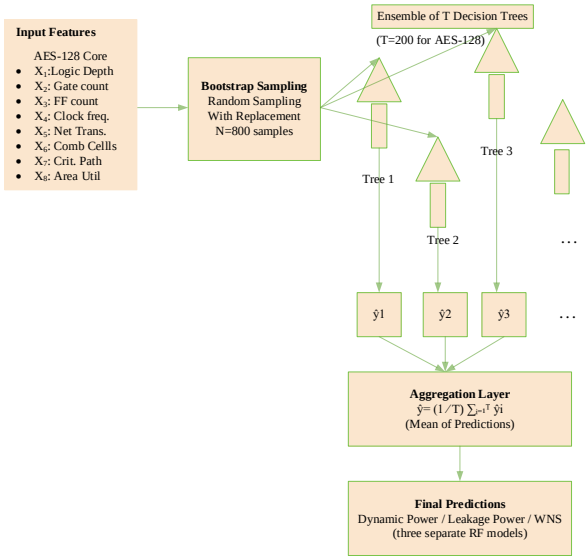

Fig. 5 RF architecture for power and timing prediction

4.3. Random Forest Model Architecture for AES-128 Core

The proposed Random Forest architecture, as shown in Figure 5, accepts eight input features from AES-128 designs. These features are logic depth, gate count, flip-flop count, clock frequency, net transitions, combinational cells, critical path delay, and area utilization. All features are normalized using z-score standardization. Bootstrap sampling creates diverse training subsets. Each subset contains 800 randomly

selected samples with replacement. The ensemble comprises 200 independent decision trees. Each tree uses recursive binary partitioning. Splits minimize mean squared error at each node.

Only $\sqrt{8} \approx 3$ ($\sqrt{}$total features) features are randomly evaluated per split. Trees grow to a maximum depth of 20 levels.

Individual trees generate independent predictions. The final prediction is the mean of all tree outputs. The formula is $\hat{y} = \left(\frac{1}{T}\right)\sum_{i=1}^{t}\hat{y}_i$ where T (number of trees) = 200. Three separate models are trained. One predicts dynamic power. Another predicts leakage power. The third predicts worst negative slack. Inference requires 82 microseconds per sample.

Using the above approach, Random Forest ensemble model enables rapid and accurate prediction of power and timing metrics for AES-128.

4.4. Neural Network Model Architecture for AES-128

The implemented neural network, as shown in Figure 6, processes eight normalized input features through a feedforward architecture comprising three hidden layers with decreasing neuron counts (64→32→16). Each hidden layer applies Rectified Linear Unit (ReLU) activation to introduce non-linearity [32]. Batch normalization follows each hidden layer to stabilize training dynamics.

Dropout regularization (0.2, 0.2, 0.3) prevents overfitting by randomly deactivating neurons during training. The output layer contains three neurons with linear activation for simultaneous multi-output regression. Forward propagation computes weighted sums followed by activation functions. The network predicts dynamic power, leakage power and worst negative slack concurrently from shared internal representations. Backward propagation calculates gradients through automatic differentiation. The Adam optimizer updates weights with adaptive learning rates. Early stopping terminates training when validation loss ceases to improve. The architecture achieves better accuracy ($R^2 > 0.95$), and inference latency per sample is 45 microseconds.

4.5. Experimental Setup

The dataset for training consists of 1000 AES-128 design constraints. Various variations are obtained through varying the frequency of the clock (50-500 MHz), the voltage supply (0.8-1.2V), the operating temperature (-40°C to 125°C) and the corners in the process (i.e., typical, fast and slow). Each design underwent RTL synthesis, static timing analysis and power estimation. Eight features are extracted from synthesis reports. Three target metrics are obtained from power analysis. The dataset is divided into 800 training samples (80%) and 200 test samples (20%). Stratified sampling allows us to get a better representation of the operating conditions. Features are z-score normalized to zero mean and unit variance.

The Random Forest and the Neural Network models separately train on homogeneous data. Random Forest uses 200 decision trees (bootstrap aggregating). Each tree learns from random subsets with replacement. The predictions are aggregated by averaging from the ensemble. Training time is estimated to last around 45 seconds. The neural network is based on the 8→64→32→16→3 pattern. Adam optimizer updates weights using mean squared error loss. Regularization of batch normalization stabilizes the training dynamics. Dropout prevents overfitting. Training stops early as soon as validation loss reaches a plateau. Training takes approximately 157 seconds.

The trained models make predictions on held-out test set. The test set contains 200 unseen samples, which are never encountered in the training stage. Forward propagation is executed by each model for normalized feature vectors. Random Forest averages forecasts from 200 trees. Outputs from the Neural Network are computed by using sequential layer transformations. At Random Forest, the inference latency is 82 microseconds per sample. The neural network obtains a prediction in 45 microseconds on each sample. Both models simultaneously predict dynamic power, leakage power and worst negative slack.

Measuring accuracy of the model is achieved by several metrics [33]. The R^2 score is the fraction of variance estimated according to predictions. Root Mean Squared Error (RMSE) indicates average deviation of prediction. Mean Absolute Error (MAE) provides an interpretable error magnitude. Mean Absolute Percentage Error (MAPE) expresses relative accuracy.

Five-fold cross-validation determines the capability of model generalization. The training set is divided into five equal folds. The validation is in each fold, and it is in the four folds that the model is trained. This will be repeated five times with validation folds. The folds of average R^2 scores indicate a high level of performance. Small levels of standard deviation verify consistent forecasts. The reliability of models is ensured with consistent interfold performance.

5. Results

In this section, the results demonstrate that the Neural Network (NN) model is slightly better than the Random Forest (RF) model with regard to all measures, which are considered. The NN has 0.6 - 1.2% higher scores on R^2, which implies better variance explanation. This trend is also confirmed using error-based metrics, where the RMSE and MAE have been reduced by 3-10 percent and 3-12 percent, respectively, over RF, and the speed of training is also 3.5 times faster than when using RF. Additionally, the fact that the error-based metrics

can provide information about the importance of features by giving them a built-in ranking of importance is useful and can greatly help in design insight and debugging. The Neural Network, on the other hand, needs about 45 times less storage space and can be better applied in memory-limited settings. Both machine learning models are far superior to the traditional EDA tool-based simulation methods, with a speed of prediction over 2,000,000x, thus allowing the design space to be explored quickly.

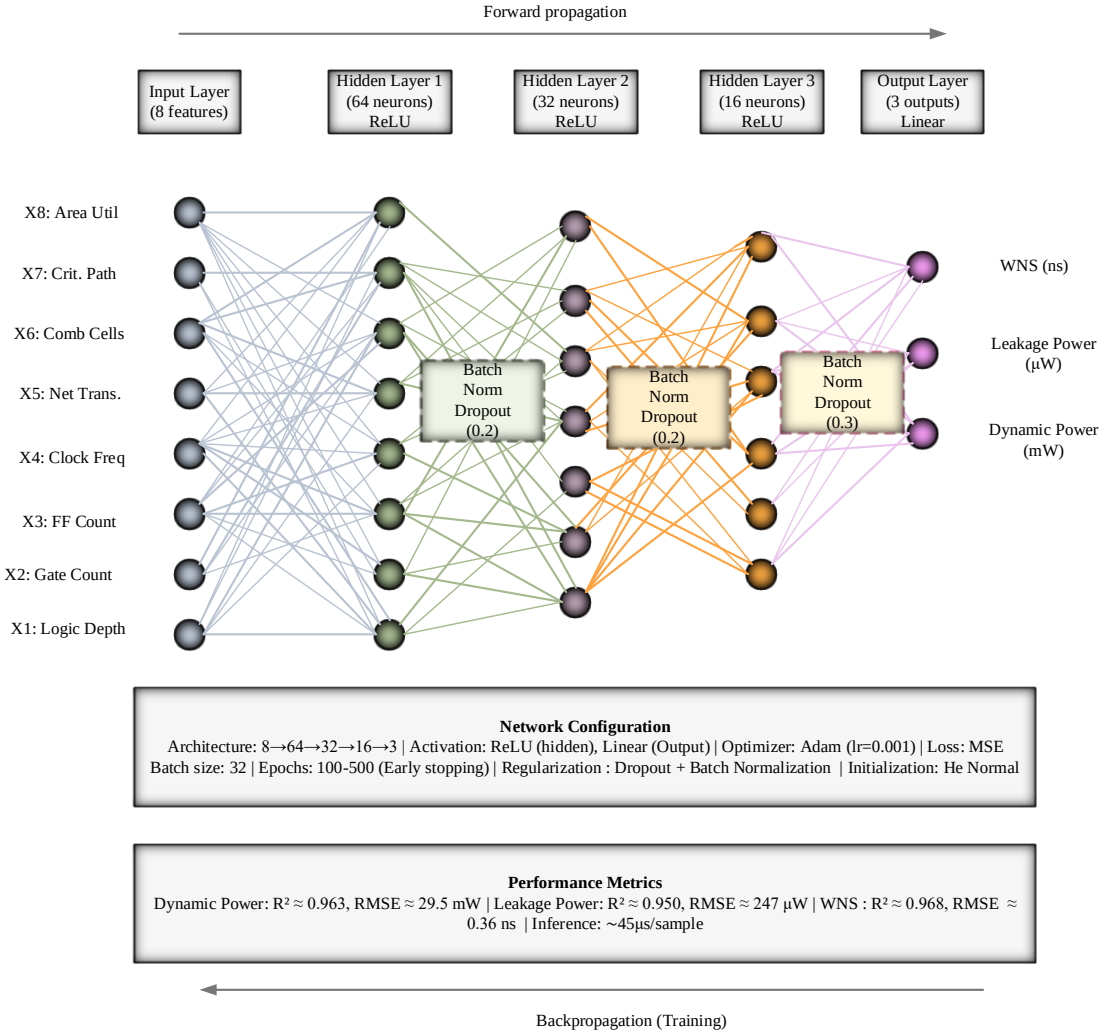

Fig. 6 Neural Network architecture for power and timing prediction

5.1. Model Validation

Outputs are compared to predicted values by being compared to ground-truth values generated by complete RTL synthesis, static timing analysis and power analysis flows. Chained complete EDA tools were applied to selected test samples. The actual simulation results have less variation with the predicted values, with a disparity of less than 5 percent in most cases.

Larger prediction errors are observed only under extreme operating conditions, such as very low voltage or high temperature corners. The best model performance can be seen in the training data envelope, where, as expected, model performance is quite robust, although out-of-distribution predictions often need closer attention and interpretation. Collectively, this validation validates the usefulness of the models for early-stage design investigation into design spaces at a physical level.

5.2. Performance Comparison

The predictions are corroborated with real synthesis and power analyses. The chosen test samples are tested with full EDA tool flow to be verified. Predicted values match well with simulation scores. The discrepancies stay below 5% for most samples.

For extreme operating conditions, the errors become larger. Models can learn best in the training data envelope. The out-of-distribution predictions be interpreted with caution. Overall validation demonstrates the practical utility with the

design space exploration. Table 3 provides a summary of quantitative performance comparisons of RF and NN models. The results clearly indicate that NN achieves slightly higher prediction accuracy, faster inference, and significantly smaller model size**,** while RF offers faster training time and superior interpretability**.**

Table 3. RF and NN model performance comparison

Aspect	Random Forest	Neural Network
R^2 (Dynamic Power)	0.954	0.963 (better)
R^2 (Leakage Power)	0.939	0.950 (better)
R^2 (WNS)	0.962	0.968 (better)
Training Time	45 sec	157 sec
Inference Time	82 µs	45 µs (faster)
Model Size	127 MB	2.8 MB (smaller)
Interpretability	High	Low
Feature Importance	Built-in	Requires analysis

The model accuracies with reference to the actual power, timing and performance numbers are presented below:-

Figure 7 depicts forecasted and real dynamic power consumption. The RF and NN models predict quite closely about the desired y = x line and have outstanding predictive performance in the entire 100-600 mW power range. The NN also has a little better scatter than that of RF, which indicates better generalization in low and high-power levels. Leakage power prediction results are given in Figure 8.

The two models are very effective at tracking actual leakage in the range of 3000-16,000 µW. The NN achieves R^2 = 0.950, outperforming RF (R^2 = 0.939). There is little systematic bias, which indicates good predictive performance in the entire leakage range.

The close concentration near the ideal line is a confirmation of good predictions. There is low bias in the range of prediction. The two models are effective in covering the entire range of leakage values. Figure 9 provides a comparison of R^2 scores on a bar chart basis.

Neural Network is more stable than the Random Forest in Dynamic Power, Leakage Power, and WNS, with R^2 differences of 0.009 for Dynamic Power, 0.011 for Leakage Power and 0.006 for WNS. The R^2 values are all greater than 0.93, and this implies that the models are of excellent quality. WNS is predicted with the highest R^2 by both models (RF: 0.962, NN: 0.968), followed by Dynamic Power (RF: 0.954, NN: 0.963) and then Leakage Power (RF: 0.939, NN: 0.950), which remains accurate but is marginally lower than the other two targets. The comparison proves that Neural Network is better than other and the effectiveness of Random Forest is true.

Figure 10 shows the accuracy of WNS prediction. The values of timing slacks are accurately reflected in both the models within a range of -3ns to +12ns. The NN estimates R^2 = 0.968 whereas RF estimates R^2 = 0.962. Predicted data are spread in an evenly distributed way around the ideal line and therefore show the lowest form of systematic error, as well as high robustness of both the positive and negative slack areas. The range of predictions is 3ns, +12ns. There is no systematic error and the points are spread evenly along the ideal line.

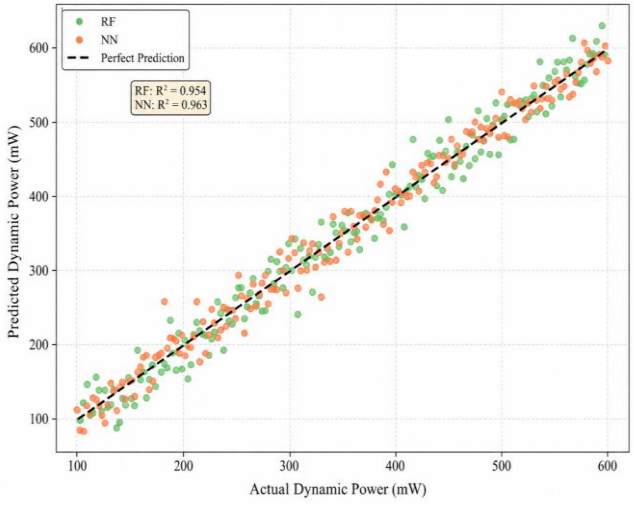

Fig. 7 Dynamic power prediction

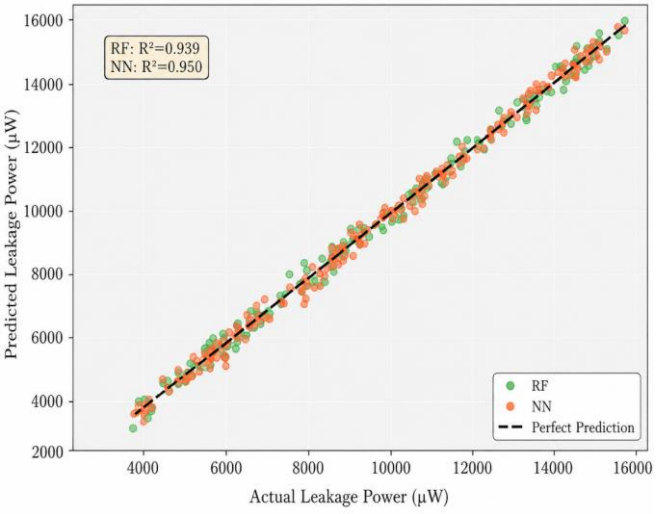

Fig. 8 Leakage power prediction

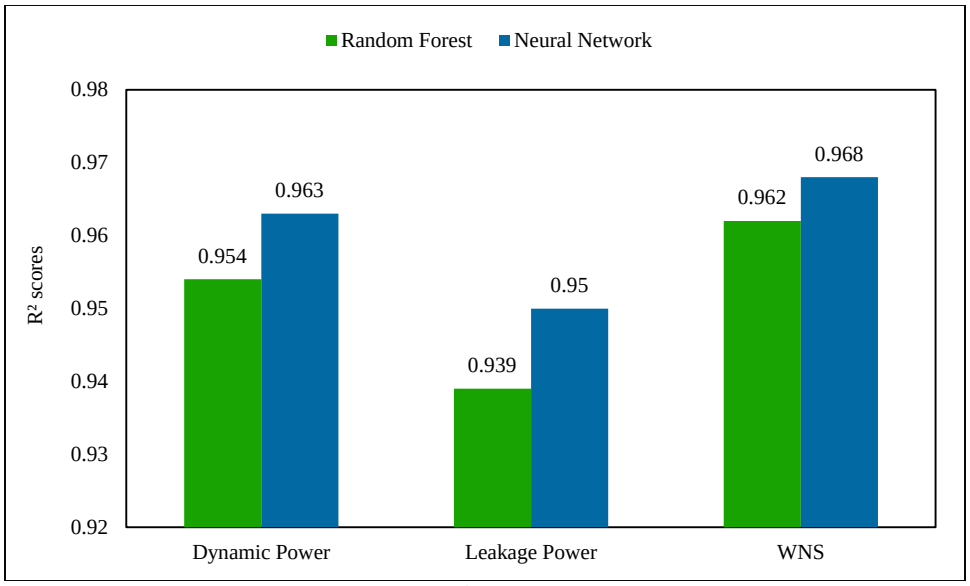

Fig. 9 Model performance comparison

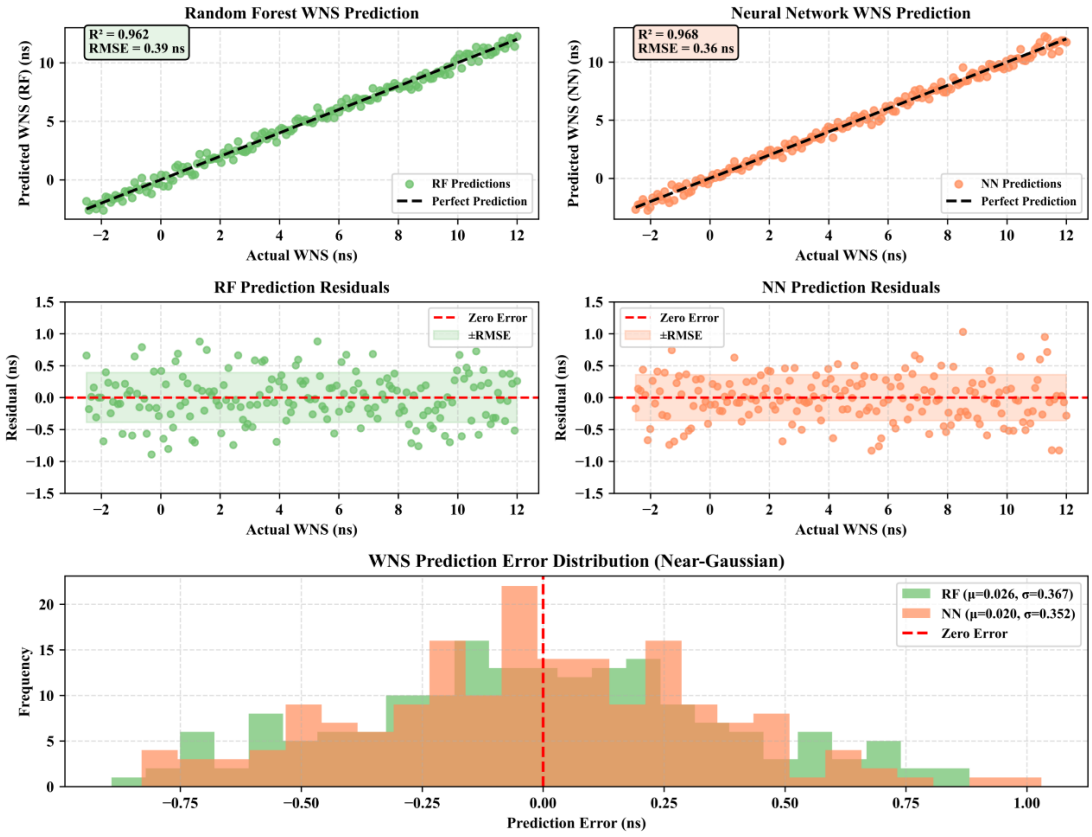

Fig. 10 Worst negative slack prediction analysis

6. Conclusion

Please refer to Table 4 below, used to represent the comparison of R^2 and inference speed among the other candidate methodologies and the proposed approach. The paper shows machine learning-based prediction of power and timing metrics of the AES-128 cryptographic cores. Two different architectures were created and tested: Random Forest and Neural Network model. The two predict dynamic power and leakage power as well as worst negative slack jointly on a set of eight extracted design features. The dataset comprised of 1000 AES-128 design variations spanning diverse operating conditions. Training on 800 samples yielded highly

accurate regressors with R^2 scores exceeding 0.93 for all targets. Random Forest achieved R^2 values of 0.954, 0.939, and 0.962 for the three metrics, respectively. Neural Network attained superior performance with R^2 scores of 0.963, 0.950, and 0.968. Cross-validation confirmed robust generalization with minimal performance variance across folds. Residual analysis validates unbiased predictions with near-Gaussian error distributions. The models eliminated the need for time-consuming synthesis and power analysis iterations. Inference required only 45-82 microseconds per sample, enabling rapid design space exploration. The other 2 design modules, viz. 32-bit ALU and dual-port SRAM controller were also verified using the same approach, and similar results were obtained in those cases.

A thousand and more design configurations can be explored in minutes, as opposed to months, and the proposed methodology provides a novel, practical framework for early-stage design optimization. The algorithm is relatively accurate and significantly low in computational cost. The analysis of the feature importance appears to demonstrate that clock frequency and the number of gates is the key factors of power. Timing predictions are of the form of critical path delay and logic depth. Designers can use these insights to make the right optimization decisions. The multi-output regression framework efficiently handles correlated targets through shared representations. Neural Networks offer slightly better accuracy with faster inference and smaller model size. Random Forests provide interpretability and faster training suitable for iterative development. Both approaches prove viable for production deployment. Future work may extend this methodology to other cryptographic primitives and design complexities. Integration with automated design flows could enable intelligent exploration of vast parameter spaces. The demonstrated result, thus, establishes RF and NN as valuable complements to traditional digital design methodologies for power and timing estimation.

Table 4. RF and NN model performance comparison

Work	Design Level	ML Technique	Predicted Metrics	Feature Scope	R^2 (Reported)	Inference Speed
Gowrishankar et al. [28]	Gate / RTL	RF, BPNN	Dynamic Power	Structural + activity	$R^2 \approx 0.90$-0.94	Not reported
Lopera et al. [27]	RTL	Neural Network	Timing Delay	Structural RTL features (gate count, logic depth, path info)	$R^2 \approx 0.92$	Not reported
Emir et al. [29]	Post-placement	Ensemble Trees	Cell Delay	Physical + layout	$R^2 = 0.86$-0.99	Not reported
Proposed Work	RTL (Early stage)	RF + NN	Dynamic Power, Leakage Power, WNS (multi-output)	Structural + Timing + Activity + Physical (8 features)	RF: 0.954 / 0.939 / 0.962; NN: 0.963 / 0.950 / 0.968 (Dynamic Power / Leakage Power / WNS)	45 µs (NN), 82 µs (RF) per sample

R^2 values are coefficients of determination on the respective test sets, as reported by the cited authors; for the proposed work, they are the test-set R^2 of the Random Forest (RF) and Neural Network (NN) models for Dynamic Power, Leakage Power and WNS, respectively. Inference speed is the per-sample prediction latency in microseconds (µs) on the same host used for training; "Not reported" indicates that the cited work does not state a per-sample inference latency.

References

[1] Anindita Chattopadhyay and Vijay Kumar Sutrakar, "Machine Learning Framework for Early Power, Performance, and Area Estimation of RTL," *2025 6th International Conference on Control, Communication and Computing (ICCC)*, Thiruvanathapuram, India, pp. 1-6, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[2] "1800-2017 - IEEE Standard for SystemVerilog--Unified Hardware Design, Specification, and Verification Language," *IEEE Std 1800-2017 (Revision of IEEE Std 1800-2012)*, pp. 1-1315, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[3] "1800.2-2020 - IEEE Standard for Universal Verification Methodology Language Reference Manual," *IEEE Std 1800.2-2020 (Revision of IEEE Std 1800.2-2017)*, pp. 1-458, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[4] Guyue Huang et al., "Machine Learning for Electronic Design Automation: A Survey," *ACM Transactions on Design Automation of Electronic Systems*, vol. 26, no. 5, pp. 1-46, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Wenji Fang et al., "MasterRTL: A Pre-Synthesis PPA Estimation Framework for Any RTL Design," *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, San Francisco, CA, USA, 2023. pp. 1-9, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[6] Pagolu Meghana et al., "Analysis of Neural Network Algorithm in Comparison to Multiple Linear Regression and Random Forest Algorithm," *2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS)*, Manama, Bahrain, pp. 437-443, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[7] Ignatious K. Pious et al., "Enhancing Prediction Accuracy Through Random Forest in Classification and Regression," *2024 International Conference on Smart Technologies for Sustainable Development Goals (ICSTSDG)*, Chennai - 600077, Tamil Nadu, India, pp. 1-6, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[8] Fahad Shamshad et al., "Transformers in Medical Imaging: A Survey," *Medical Image Analysis*, vol. 88, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[9] João Manoel Herrera Pinheiro et al., "The Impact of Feature Scaling in Machine Learning: Effects on Regression and Classification Tasks," *IEEE Access*, vol. 13, pp. 199903-199931, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[10] Martin Rapp et al., "MLCAD: A Survey of Research in Machine Learning for CAD Keynote Paper," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 10, pp. 3162-3181, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[11] Leo Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001. [CrossRef] [Google Scholar] [Publisher Link]

[12] Thomas Rincy N, and Roopam Gupta, "A Survey on Machine Learning Approaches and Its Techniques," *2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS)*, Bhopal, India, pp. 1-6, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[13] Tetiana Reznychenko, Evženie Uglickich, and Ivan Nagy, "Accuracy Comparison of Logistic Regression, Random Forest, and Neural Networks Applied to Real MaaS Data," *2024 Smart City Symposium Prague (SCSP)*, Prague, Czech Republic, pp. 1-5, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[14] Nikolaos I. Deligiannis et al., "A Reliability Evaluation Flow for Assessing the Impact of Permanent Hardware Faults on Integer Arithmetic Circuits," *IEEE Access*, vol. 13, pp. 32177-32196, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep Learning," *Nature*, vol. 521, pp. 436-444, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[16] Kurt Hornik, Maxwell Stinchcombe, and Halbert White, "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, vol. 2, no. 5, pp. 359-366, 1989. [CrossRef] [Google Scholar] [Publisher Link]

[17] Sohrab Mirsaeidi et al., "Applications of Machine Learning in Modern Power Systems: A Comprehensive Review," *2023 3rd International Conference on New Energy and Power Engineering (ICNEPE)*, Huzhou, China, pp. 1074-1080, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, Cambridge, MA: MIT Press, 2016. [Google Scholar] [Publisher Link]

[19] Wenji Fang et al., "A Survey of Circuit Foundation Model: Foundation AI Models for VLSI Circuit Design and EDA," *ACM Transactions on Design Automation of Electronic Systems*, pp. 1-73, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[20] Bernhard Schölkopf and Alexander J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, Cambridge, MA: MIT Press, 2002. [Google Scholar] [Publisher Link]

[21] Christopher J.C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121-167, 1998. [CrossRef] [Google Scholar] [Publisher Link]

[22] Tianqi Chen and Carlos Ernesto Guestrin, "XGBoost: A Scalable Tree Boosting System," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785-794, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[23] Jerome H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189-1232, 2001. [Google Scholar] [Publisher Link]

[24] Ceyu Xu, Chris Kjellqvist, and Lisa Wu Wills, "SNS is not a Synthesizer: A Deep-Learning-based Synthesis Predictor," *Proceedings of the 49th Annual International Symposium on Computer Architecture*, New York, NY, USA, pp. 847-859, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[25] Prianka Sengupta et al., "How Good is Your Verilog RTL Code? A Quick Answer from Machine Learning," *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, San Diego, CA, USA, no. 89, pp. 1-9, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[26] Wenji Fang et al., "Annotating Slack Directly on Your Verilog: Fine-Grained RTL Timing Evaluation for Early Optimization," *Proceedings of the 61st ACM/IEEE Design Automation Conference*, San Francisco, CA, USA, pp. 1-6, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[27] Daniela Sánchez Lopera et al., "Early RTL Delay Prediction Using Neural Networks," *Microprocessors and Microsystems*, vol. 94, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[28] R. Gowrishankar et al., "Intelligent Power Estimation of Digital Circuits using Random Forest and Neural Network Models," *International Journal of Computational Intelligence Systems*, vol. 18, pp. 1-33, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[29] Recep Emir et al., "Deep Learning Approach for Modeling the Power Consumption and Delay of Logic Circuits Employing GNRFET Technology," *Electronics*, vol. 13, no. 15, pp. 1-14, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[30] Morris J. Dworkin et al., "Advanced Encryption Standard (AES)," *National Institute of Standards and Technology*, *Federal Information Processing Standards*, vol. 197, 2001. [CrossRef] [Google Scholar] [Publisher Link]

[31] *Design Compiler User Guide*, Synopsys Inc, Mountain View, CA, USA, 2023. [Google Scholar] [Publisher Link]

[32] Adam Paszke et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Proceeding 33rd Conference Neural Information Processing Systems (NeurIPS)*, Vancouver, BC, Canada, pp. 8024-8035, 2019. [Google Scholar] [Publisher Link]

[33] Spyros Makridakis, "Accuracy Measures: Theoretical and Practical Concerns," *International Journal of Forecasting*, vol. 9, no. 4, pp. 527-529, 1993. [CrossRef] [Google Scholar] [Publisher Link]