09-06-2026

by Ankita Negi

Submission date: 09-Jun-2026 09:01PM (UTC+0530)
Submission ID: 2979827268
File name: 11-40.pdf (1.44M)
Word count: 5484
Character count: 30167

CHAPTER-1

INTRODUCTION

1.1 BACKGROUND

The continuous scaling of CMOS technology has made it possible placing billions of transistors on a single chip, which leads to improvements in computing capability. Device dimensions moving further into nanometer range, circuit design and verification have become more challenging now. Amomg the key issues in modern VLSU design, the most important is the precise and efficient estimation of timing parameters, propagation delay because it has a direct impact on circuit speed and the overall system performance. Propagation delay is the time taken for a change at the input of a logic gate to appear at the output. This delay does not depend on a single factor in CMOS circuits but it is affected by several parameters, like transistor width, length, supply voltage, load capacitance, temperature, and process-related variations. With growing complexity in circuits and increasing demand for higher clock frequencies in multi-gigahertz range, even a small error in dely prediction can cause us timing issues, malfunctioning, or expensive redesign efforts.

Traditionally. The SPICE simulations have been used to analyze the delay behavior of the logic circuits. These simulations are well known for their high accuracy because they are based on detailed semiconductor device equations, but also making them computationally expensive. A single simulation may take several seconds, and when a large design in there the total runtime becomes extremely high and difficult to manage.

This computational bottleneck has motivated researchers to explore alternative approaches based on machine learning and deep learning techniques. These data-driven methods can learn the complex relationships that are non linear between circuit parameters and delay values from a training dataset, and subsequently make predictions in a fraction of the time required by SPICE simulation. The potential for orders-of-magnitude speedup in delay prediction makes deep learning an attractive tool for accelerating VLSI design exploration and optimization.

1.2 Propagation Delay in CMOS Circuits

Propagation delay is one of the most critical performance parameters of a logic circuit designed using the CMOS process. It has a direct effect on the performance of the digital system and is a key consideration during the design phase, especially as system complexity and clock frequency continue to increase. The propagation delay of a CMOS logic gate is defined as the time elapsed from the point at which the input signal crosses 50% of its voltage transition until the time at which the output signal crosses 50% of its corresponding voltage transition. Two components of propagation delay are typically measured: Rise delay (tPLH): The delay measured when the output transitions from logic low to logic high (low-to-high propagation delay). Fall delay (tPHL): The delay measured when the output transitions from logic high to logic low (high-to-low propagation delay). The average propagation delay (tpd) is generally defined as:

$$tpd = (tPLH + tPHL) / 2$$

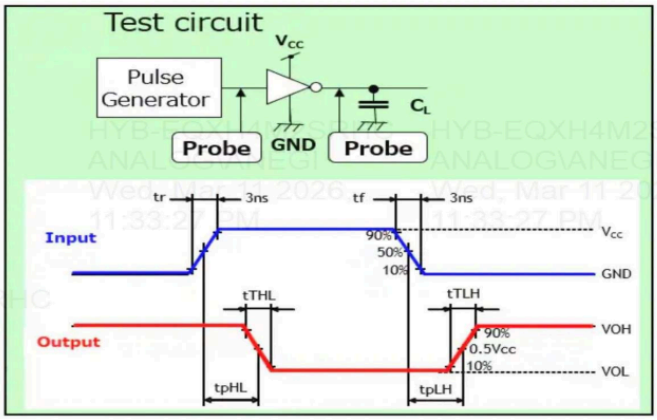


Fig.1.1: Propagation Delay in CMOS Circuits

The average propagation delay indicates the overall switching delay of the CMOS gate and serves as a key parameter in digital timing analysis. It is affected by many factors like:

1. **Transistor Width (w):** Wider the transistors greater the drive current, reducing the charging and discharging time, thereby reducing propagation delay.

2. Channel Length (L): Shorter channel lengths improve transconductance and drive current, reducing the propagation delay. However, at smaller technology nodes, short-channel effects become more prominent.

3. Supply Voltage (VDD): High supply voltage increases the overdrive voltage of the transistor, which results in greater drive current and low propagation delay. However, higher VDD is also responsible for increase in dynamic power consumption.

4. Load Capacitance (CL): The output load capacitance includes the gate capacitance, interconnect capacitance, and parasitic capacitances, which directly affects the delay. A large CL will require more time to charge or discharge, thus increasing the propagation delay.

5. Temperature: Temperature has effects on carrier mobility and threshold voltage. At higher temperatures, carrier mobility reduces which leads to lower drive current and increase in propagation delay.

6. Process Variations: Manufacturing variations in Vth, tox, and doping concentration introduces variability in the propagation delay across different dies and wafers.

1.3 CMOS Logic Gates

This thesis focuses on the most fundamental CMOS logic gates:

A CMOS Inverter: It is the fundamental logic gate, it consists of one PMOS transistor which is connected in series with one NMOS transistor between the supply voltage (VDD) and ground (GND). Both transistors gate are connected with the input, and the output is taken from the common drain connection. When our input is logic high, the NMOS transistor will turn on and the PMOS transistor will turn off, pulling the output to ground (logic low). Conversely, when the input is logic low, the PMOS transistor turns on and the NMOS transistor turns off, pulling the output to VDD (logic high). The inverter performs signal inversion and buffering and is the fundamental building block from which all other CMOS logic gates are constructed.

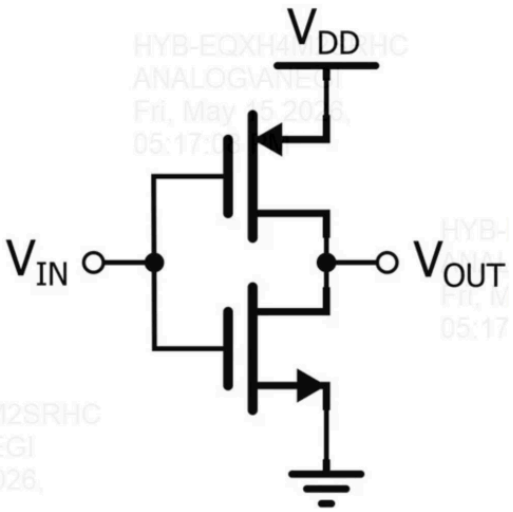


Fig. 1.2: CMOS Inverter Circuit

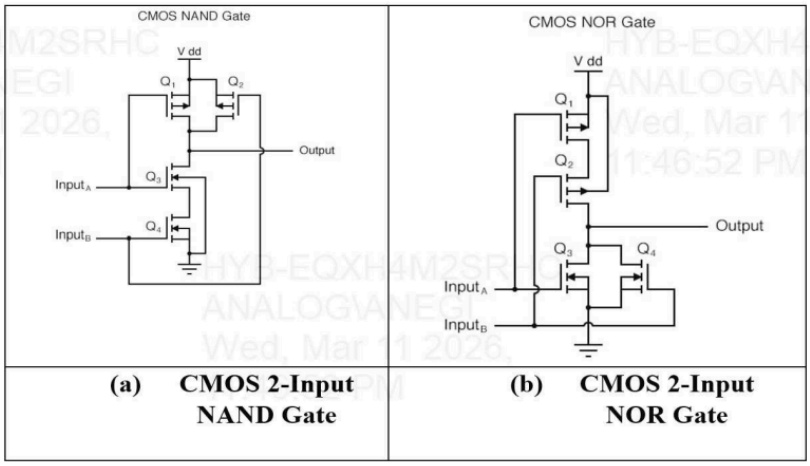
CMOS 2-Input NAND Gate:

The 2-input NAND gate consists of two NMOS transistors which are connected in series between the output node and the ground, and two PMOS transistors connected in parallel between VDD and the output node. The output will be logic low only when both the inputs will be logic high (both NMOS transistors conducting, both PMOS transistors off). For any other input combination, at least one PMOS transistor conducts and pulls the output to VDD. The NAND gate is one of the most widely used gates in digital design because it is a universal gate- any Boolean function can be implemented using only NAND gates. It is extensively used in the construction of adders, multiplexers, memory cells, and other complex digital circuits.

CMOS 2-Input NOR Gate:

It consists of 2 PMOS transistors which areconnected in series with one node on VDD and other with the output node, and 2 NMOS transistors connected in parallel one with output and other with ground. The output is logic high only when both inputs are logic low (both PMOS transistors conducting, both NMOS transistors off). For any other input combination, at least one NMOS

transistor conducts and pulls the output to ground. Like the NAND gate, the NOR gate is also a universal gate and is used in constructing various complex digital circuits.



(a) CMOS 2-Input NAND Gate **(b) CMOS 2-Input NOR Gate**

These three gates are selected for this study because they encompass the fundamental electrical characteristics and switching properties of CMOS circuits, including rise delay, fall delay, and loaddependent propagation delay behavior. Understanding and accurately predicting the delay of these fundamental gates provides a foundation for predicting the timing behavior of more complex circuits built from these building blocks.

1.4 Traditional Delay Estimation - SPICE Simulations

SPICE which is the gold standard for circuit simulation in the semiconductor industry. SPICE solves the nonlinear differential equations that describe the behavior of semiconductor devices and circuit elements, providing highly accurate results for voltage, current, and timing analysis. For estimation of propagation delay, SPICE apply input stimulus (typically a pulse or ramp signal) and performs transient analysis. The delay is then measured as the time difference between 50% crossing points of i/p and o/p wavefotms.

While SPICE simulation provides good accuracy but it has significant limitations for large-scale designs:

1. Computational Cost: Each SPICE simulation run requires solving a set of differential equations at every time step, which can take time for even simple circuits. For the 45 nm technology node used in this study, a single SPICE run takes around 12.5 seconds.

2. Scalability: In design optimization it is often necessary to analyze thousands of different parameters. The total simulation time increases with the evaluated number of points, performing exhaustive search becomes very impractical.

3. Iterative Design: Modern VLSI design involves multiple iterations of synthesis, placement, routing, and timing analysis and the delay estimation must be performed for millions of gates, which makes SPICE simulation impractical. This motivates the development of a faster delay estimation methods which can maintain accuracy and speed up the simulation process also.

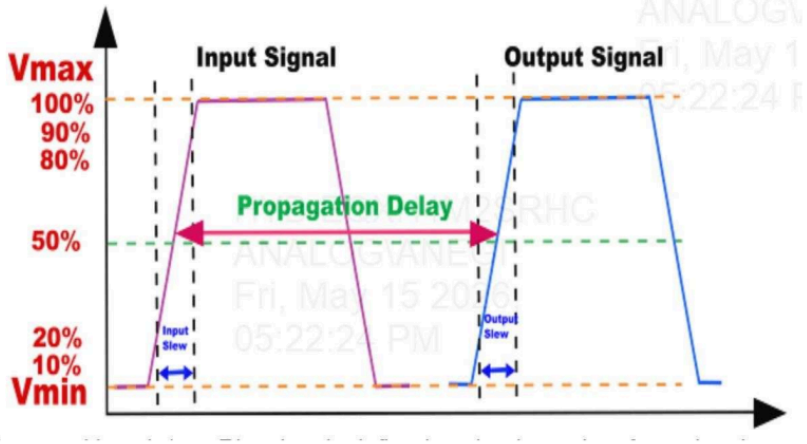


Fig.1.3: Propagation Delay

1.5 Deep Learning and Machine Learning in VLSI Design

The ML and DL techniques application in VLSI design problems has increased significantly in recent years. These approaches can learn from complex nonlinear patterns of training data and make fast predictions, which makes them idle for computationally intensive simulation tasks.

Timing Prediction: ML models that are trained on SPICE simulation data can predict the delay with high accuracy like the gate delays, path delays, and timing slack and also significantly reducing the computational time.

Power Estimation: Neural networks predict consumption of both dynamic and static power using the design parameters and activity factors.

Process Variation Modeling: ML techniques help capture the statistical behavior of the performance of circuit under variations in manufavturing.

Design Optimization: Deep learning models can act as surrogate models in optimization frameworks, enabling exploration of large design spaces faster.

Among the different ML techniques available, the Multilayer Perceptron (MLP), which is a feedforward artificial neural network, has shown strong potential for regression tasks in circuit modeling. Multiple layers of neurons are there in MLPs, and each apply a nonlinear activation function to a weighted combination of it's inputs. Because of this architecture we are able to learn highly complex relationships between the input parameters and output.

1.6 Problem Definition and Objectives Problem Definition:

Prediction of propagation delay in CMOS circuit having low computational cost compared to standard SPICE simulation methods is the primary concern. It is necessary to identify the major circuit parameters like load capacitance, voltage supply, and transistor size

Objectives:

1. Predicting CMOS logic circuit delay accurately using Deep learning model which will target rise delay, fall delay and propagation.

2. Comparison of traditional Machine Learning methods like linear regression, polynomial regression, random Forest, SVR, Gradient Boosting with the current model performance.

3. Sensitivity analysis to be done to identify key parameters that affect propagation delay and quantifying the relative importance.

4. Maintaining high prediction accuracy Demonstrate significant computational speedup over conventional SPICE simulation while maintaining high prediction accuracy.

CHAPTER 2

LITERATURE REVIEW

Machine learning and deep learning techniques integrated in circuit analysis and VLSI design has emerged as an active research area. This chapter presents a review of the major contributions in this field, categorizing them according to the key research themes relevant to this thesis.

2.1 Machine Learning for VLSI Timing Prediction

Cheng et al. [1] presents a ML-based prediction methodology for designing and system technology cooptimization sensitivity analysis. In this review it is demonstrated that ML techniques can capture complex relationships wrt technology parameters and circuit performance metrics, including timing, in advanced VLSI systems effectively. The study highlighted the potential of ML to accelerate the DTCO process by replacing computationally expensive simulations.

Nie et al. [3] came up with a method that leverages complex network theory which is combined with ML to then estimate the circuit timing. Representing the circuit as a graph structure, delay estimation is done at the network level by extracting features. The findings indicated accounts for circuit topology that leads to a notably more accurate prediction compared to relying on local parameter-based models.

Jang [4] carried out an in-depth investigation to get timing prediction and optimize it during the physical design phase. The study evaluated several ML methods for estimating the timing in post route using the pre-route characteristics.

Hu et al. [5] delay prediction algorithm, which was tailored for FPGA technology mapping, which was built on machine learning models was proposed. The results confirmed that ML models can predict the gate-level delays in FPGA designs accurately, further enhancing the overall performance.

Fakir and Mande [9] performed analysis on a CMOS full adder with the help of ML techniques. In the work they used ML models for predicting the delay and power, demonstrating the application of ML-based prediction model to more complex logic circuit beyond the standard gates.

2.2 Deep Learning Approaches for Circuit Modeling

Emir et al. [2] came up with a deep learning framework to estimate the power consumption and delay of logic circuits using graphene nanoribbon field effect transistor (GNRFET) technology. Their study showed that deep neural networks can be helpful in predicting multiple circuit parameters with high accuracy, and that this framework can be applicable to newer device technologies beyond conventional CMOS.

Sajjadi et al. [6] came up withs a strategy to increase and improve accuracy of electronic circuit design using a DNN-based optimization. The DNNs here served as surrogate models in place of a full circuit simulations, that allowed the design space quickly without sacrificing the fidelity.

Reuter et al. [7] described a ML-based approach to build compact model design for reconfigurable FETs. Their findings demonstrated that ML is not only limited to prediction of performance but can also be used for developing device-level compact models which can be directly integrated into the circuit simulators.

Jacinto et al. [12] investigated the machine learning techniques to study prediction for electrical behavior and used CMOS inverter for his study. In this investigation multiple ML techniques were evaluated to predict the I-V characteristics and the timing of CMOS inverter.

2.3 Sensitivity Analysis in CMOS Design

Singh and Kalra [8] proposed a ML-based approach for reliability analysis of Negative Bias Temperature Instability (NBTI) designs for integrated circuits. Their work used the ML techniques to analyze the sensitivity of circuit reliability of different designs and process parameters, thus providing the dominant factors which affects circuit aging and degradation.

Malarkodi et al. [10] introduced a support vector machine (SVM) based framework for accurate on-chip power analysis in CMOS VLSI circuits. Their method focused mainly on identifying the parameters which affecr the power consumption and also to predict the power values for various operating conditions.

Zhang et al. [11] developed a methodolgy for SET analysis in the pass transistor logic. Their results gave idea that ML models can be helpful in predicting the sensitivity of circuits to radiation-induced

disturbances. Without the need for intensive Monte Carlo simulation, rapid reliability analysis can be enabled.

Dehghani et al. [13] presented a machine learning-based framework with a methodology SHAP (SHapley Additive exPlanations) sensitivity analysis. Their work was applied to different domains, the approach was to interpret sensitivity analysis using SHAP values to provide a methodology which is applicable to CMOS circuit parameter analysis.

2.4 Summary of Literature Review

The reviewed literature indicates that Machine and Deep learning methods can be very promising tools to predict the performance of circuit parameters and conducting sensitivity analysis. However, several research gaps are still observed:

1. Major of the existing research focuses on either delay prediction or sensitivity analysis differently, rather than exploring both in a single methodology.

2. Lack of research is there on comprehensive comparison for different ML/DL techniques mainly for prediction of propagation delay of the standard CMOS gates.

3. Very few studies provide comparisons of detailed quantitative of computational speedup improved with ML/DL methods versus the SPICE simulations.

4. The role of engineered features such as the time constant RC and width ratios in improving the prediction accuracy has not been researched.

This thesis aims to bridge these gaps by developing a deep learning based model for prediction of delay of standard CMOS logic gates, and comparing it between different techniques, further performing sensitivity analysis and also demonstrating the speedup achieved my ML/DL models over the SPICE simulations.

CHAPTER 3

METHODOLOGY

This chapter describes the overall approach to develop the Deep Learning based model to predict propagation delay. Each step of the process is covered which includes the selection of circuits, simulations, then dataset creation, data preprocessing, feature engineering, baseline model development, design of Deep learning architecture, training the process and the metrics which is used to evaluate the performance.

3.1 Circuit Selection

This research on fundamental CMOS logic. These gates are selected for the following reasons:

1. **Basic Building Blocks**: The inverter, NAND, and NOR gates are the foundation of all the digital circuits. Any Boolean function can be realized using NAND and NOR.

2. **Fundamental Electrical Characteristics**: These gates capture the essential switching characteristics of CMOS circuits which includes the rise/fall time, propagation delay which are common to all the CMOS logic designs.

3. **Practical Relevance:** These gates are widely used to make complex digital system like adders, MUX, decoders, FF and also the memory units. At this level the accurate delay estimation is very crucial for the timing analysis of large circuits.

4. **Diverse Topologies:** The inverter (one NMOS and one PMOS), NAND, and NOR represent main transistor configuration in CMOS logic, that offer a comprehensive basis for building and evaluating the prediction model.

3.2 Technology Node and Simulation Setup

A 45 nm CMOS technology node is selected for performing the propagation delay analysis. The selection of the 45 nm node is based on its widespread adoption in modern VLSI design for highspeed digital circuits with low power consumption and high integration density.

Parameter	Level / Value
Simulation Environment	SPICE-based Circuit Simulator
Technology Node	45 nm CMOS Technology
Supply Voltage (VDD)	1.0 V
Temperature Condition	27°C
Process Corner	TT (Typical-Typical)
Additional Process Corners (Optional)	FF, SS

Table 1: Simulation Environment and Operating Conditions for CMOS Delay Analysis

The propagation delay measurement methodology employs the standard 50% input voltage to 50% output voltage transition method. The delay is calculated as the time difference between when the input touches 50% of the voltage level and when the output touches 50% of the voltage transition level. It is used for consistent measurement of both the rise and fall delay in CMOS logic circuits.

3.3 Parameter Variation Strategy

A CMOS-based parameter variation strategy is employed where key circuit parameters are varied systematically to study their effect on switching behavior. The key parameters and the variation ranges are shown in Table 2.

Parameter	Typical Range of Variation
Wn (NMOS Width)	100 nm – 500 nm
Wp (PMOS Width)	200 nm – 800 nm
L (Channel Length)	45 nm – 90 nm
VDD (Supply Voltage)	0.8 V – 1.2 V
CL (Load Capacitance)	1 fF – 20 fF
Temperature	0°C – 100°C

Table 2: Key Parameters and Their Role in Propagation Delay Analysis

Latin Hypercube Sampling (LHS) is used to ensure unbiased coverage of the parameter space. Unlike simple random sampling or grid-based sampling, LHS divides the range of each parameter to equal intervals and then it is ensured that each of the interval is sampled only once. This approach allows us for good coverage of the whole range of input variables, which captures the nonlinear relationships more effectively, thus reducing the total simulations. As a result, LHS provides a more efficient and systematic way tof exploring high dimensionsl parameter spaces compared to traditional full factorial sampling methods.

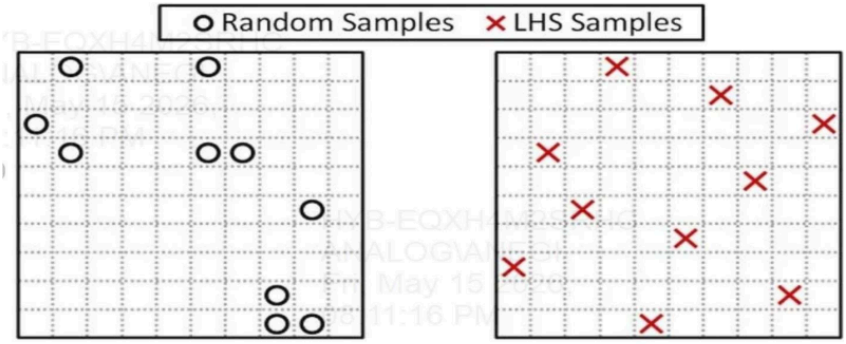


Fig. 3.1:Random Sampling vs Latin Hypercube Sampling

3.4 Dataset Generation

The parameter sweep method is implemented using a SPICE-based simulator, which is used to sweep the parameters which are varied across the defined ranges. This will enable us to generate a comprehensive dataser by observing the circuit behavior under different. The major input parameters selected for the simulation are:

- Width of NMOS (Wn)
- Width of the PMOS transistor (Wp)
- Channel length (L)
- Supply voltage (VDD)
- Load capacitance (CL)
- Temperature

By varying these parameters, many simulations are created which represent different operating conditions of the CMOS logic circuits. Propagation delay is determined using the standard 50% voltage transition method.

The dataset includes input parameters like Wn, Wp, L, VDD, CL, and temperature. The output variables consists of rise delay (tPLH), fall delay (tPHL), and average propagation delay (tpd). Around 2500 simulation samples are generated, ensuring proper coverage of the nonlinear relationships between the input parameters and propagation delay.

3.5 Data Preprocessing

To ensure data quality and reliability, dataset undergoes through preprocessing before training:

1. **Invalid Data Removal**: Data points are identified and removed from unstable SPICE simulations (e.g., convergence failures, unrealistic delay values).

2. **Outlier Filtering:** Outliers are identified and then they are removed with the use of Interquartile Range (IQR) method. Points of data lying below $Q1 - 1.5\times IQR$ or above $Q3 + 1.5\times IQR$ are taken as outliers and are excluded from the dataset.

3. Feature Normalization: Normalizing of all input features are done using Min-Max. This ensures equal contribution of every feature of model training and prevents features with large values from dominating the process.

The value of a feature is normalized is calculated using the below equation:

$$X\_norm = (X - X\_min) / (X\_max - X\_min)$$

Fig. 3.2: Data Preprocessing Pipeline

3.6 Feature Engineering

Feature engineering is done to improve the accuracy of DL models . It is done by introducing an additional derived features that captures a better underlying behavior of CMOS circuits.

Transistor Width Ratio (Rw): The transistor width ratio gives the relative strength of the PMOS and NMOS devices. It is important to determin the switching behavior of CMOS gates:

$$Rw = Wp / Wn$$

RC Time Constant Product (τ): The propagation delat in CMOS circuits can be approximated by using the RC delay model. RC gives the product of resistor and capacitor which is he time constant that is associated with charging and discharging of the load capacitance.

$$\tau = Reff \times CL$$

where τ gives the estimated time constant which is related to delay, effective channel resistance is given by Reff, and CL is the load capacitance.

Effective Drive Strength (Deff): It is a simplified measure that gives the drive capability of the network that is used to represent how effectively it can source /sin the current.

$$Deff = (Wn + Wp) / L$$

3.7 Dataset Splitting and Cross-Validation Three

parts are there for dataset splitting:

Training Set (70%): Used to learn the parameters of model.

Validation Set (15%): To tune the hyperparameters and for monitoring the overfitting.

Test Set (15%): To evaluate the model performance.

Assuming that the dataset contains N samples:

For ensuring the reliability of the results, a 5 fold cross validation is applied. In the method dataset will be divided into k subsets which are same, then the model will be trained k times. While training the model uses a different fold as the test set and the remaining k-1 is taken as the taining set.

N_fold = N / k,

Ww have used k = 5 in this study. This approach is used in which every data sample is used for both training and testing across different iterations, which improves the robustness and lead to a reliable dela prediction model.

3.8 Baseline Model Development

Several ML models are developed before implementing the deep learning model. The following baseline models are implemented:

Linear Regression (LR): Linear Regression gives us the propagation delay based on a weighted sum of the input features:

$$y = \beta_0 + \Sigma(\beta_i \times x_i) \text{ for } i = 1 \text{ to } n$$

where y represents is the predicted delay, x_i represents the input parameters (like Wn, Wp, L, VDD, CL, Temperature), β_0 is the intercept, and β_i are the coefficients. Linear regression may not properly capture the nonlinear dependencies in the circuit.

Polynomial Regression (PR): It is the extension of LR by including polynomial terms in the features of input:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \ldots + \beta_k x^k$$

where k is the degree of the polynomial. This model captures the nonlinear relationships.

Random Forest (RF): In Random Forest many decision trees are made and then the average prediction of each treeis taken.

$$\hat{y} = (1/T) \times \Sigma(y_t) \text{ for } t = 1 \text{ to } T$$

where $\hat{y}$ is the predicted delay, T is the total number of trees, and y_t is the prediction from the t-th decision. Random forest is capable of capturing complex non linear patters and reboust to overfitting.

Support Vector Regression (SVR): SVR transforms the input into a high dimension space using the kernel functions and it determines an optimal regression hyperplane which defined epsilom. It is good for non liners problmens but is intensive for large datasets.

Gradient Boosting (GB): Weak learners are sequentially ensembles in Gradient Boosting, where each new model attempts to correct the errors that were made by the previous ones. It requires carefuk tuning but achives high prediction accuracy. If it is not properly regularized it can overfit.

3.9 Deep Learning Model Architecture

Following the evaluation of baseline models, a deep learning regression model is developed to effectively address the nonlinear relationships between CMOS circuit parameters and propagation delay. The model tha is proposed is based on the feedforward Artificial Neural Network (ANN) architecture, specifically a Multilayer Perceptron (MLP).

Input Layer:

The input layer gets the normalized circuit parameters and engineered features: Wn, Wp, L, VDD, CL, Temperature, width ratio (Rw), and RC product (τ). The input vector is expressed as:

$X = [x_1, x_2, x_3, \ldots, x_n]$

where x_i represents the i-th input feature.

Hidden Layers:

The network contains multiple hidden layers. Each neuron in a hidden layer computes a weighted sum of its inputs plus a bias term, followed by a nonlinear activation function:

$$z_j = \Sigma(w_{ij} \times x_i) + b_j \text{ for } i = 1 \text{ to } n$$

$$a_j = f(z_j)$$

Activation Function -

ReLU: The Rectified Linear Unit (ReLU) activation function

$$f(z) = max(0, z)$$

ReLU helps the network learn nonlinear relationships efficiently while ignoring the gradient problem affecting the sigmoid and tanh activation functions during the backpropagation training process.

Output Layer:

The output layer produces the predicted propagation delay value. For each gate type, the model predicts the rise delay (tPLH), fall delay (tPHL), and average propagation delay (tpd).

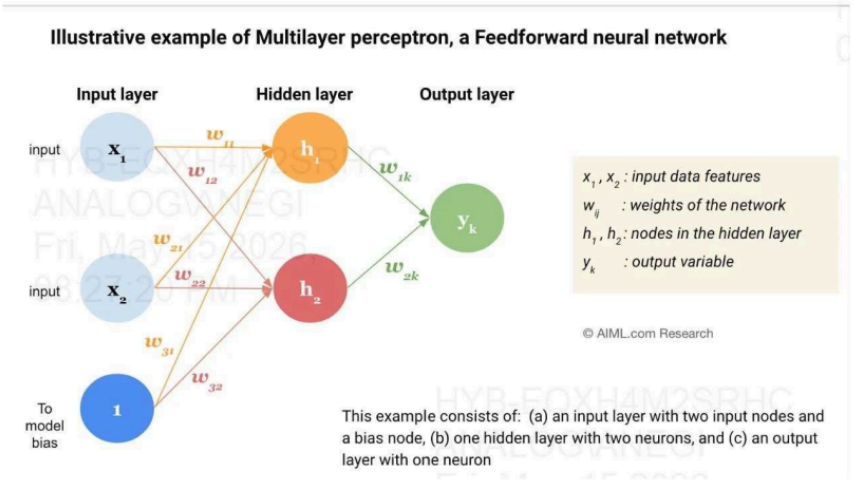


Fig. 3.3: Architecture of Artificial Neural Network (MLP)

3.10 Model Training Procedure

The model training process involves the following steps:

1. **Weight Initialization:** Network weights are randomly initialized.
2. **Loss Computation:** The Mean Squared Error loss function is the difference between predicted and actual delay values.

3. **Weight Update:** The model weights are updated by using the Adam optimizer to further minimize the loss and improve the accuracy.

The MSE loss function is defined as:

$$MSE = (1/N) \times \Sigma(y_i - \hat{y}_i)^2 \text{ for } i = 1 \text{ to } N$$

Parameter	Value / Method
Weight Initialization	Random Initialization
Forward Propagation	Enabled
Loss Function	Mean Squared Error (MSE)
Backpropagation	Gradient-based
Optimizer	Adam
Number of Epochs	100
Batch Size	30
Learning Rate	0.0005
Validation Monitoring	Validation Loss Tracking
Early Stopping	Enabled

Table 3: Deep Learning Model Training Parameters

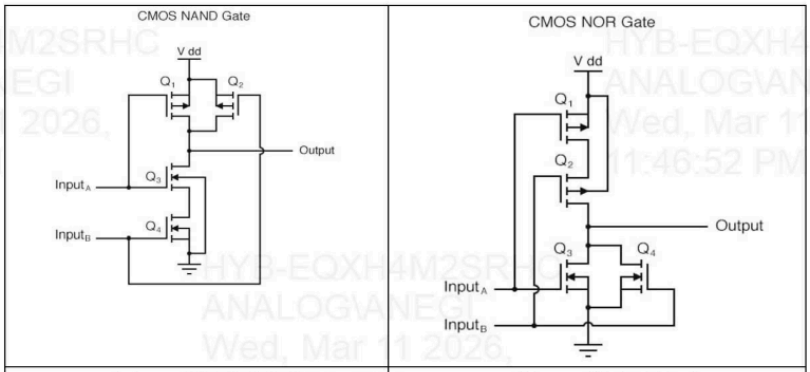
During training, the performance of model is continuously evaluated on the set that is validated. Early stopping is applied to stop the training once the validation no longer improves, it helps to avoid overfitting to ensure better generalization of the model.

Metric	Description
Mean Squared Error (MSE)	Average squared prediction error — measures the average of the squares of the differences between predicted and actual values
Root Mean Squared Error (RMSE)	Square root of MSE — provides error in the same units as the target variable
Mean Absolute Error (MAE)	Average absolute prediction error — measures the average of the absolute differences between predicted and actual values
Coefficient of Determination (R^2)	Proportion of variance in the dependent variable explained by the model — $R^2 = 1$ indicates perfect prediction
Percentage Error (PE)	Error relative to the actual value, expressed as a percentage

Table 4: Different metric and description

CHAPTER-4

RESULTS



(a) CMOS 2-Input NAND Gate

(b) CMOS 2-Input NOR Gate

The five baseline models —

All the models are trained and then tested on the same dataset.

The results shows different R2 score and percentage error for all models:

Linear Regression shows an R^2 score of 0.86 and a percentage error of 9.2%, which shows that linear assumption is unable to capture the non linear relationship.

Polynomial Regression shows an R^2 of 0.90 and 7.8% error. It suggests that polynomial can capture the complexnon linear relationship between input and output, which Linear regression in unable to do.

Random Forest shows an R^2 of 0.93 with 5.9% error. The performance is further improved here due to the ensemble structure which is better in capturing the non linear interactions.

Support Vector Regression shows an R^2 of 0.91 and 6.5% error, which means it can handle non linear interaction very well.

Gradient Boosting shows the best results with an R^2 of 0.94 and 5.2% error. Here the error is reduced and more R2.

4.2 Deep Learning Model Performance

The proposed MLP-based deep learning model gives better results across all the metrics:

MSE is 0.0012 which is the lowest among all the models.

RMSE is 0.034 which shows minimal deviation and high accuracy.

MAE is 0.025 which means low error.

R2 is 0.97 which is highest among all.

This shows that the performance is overall improved as compared to other ML models. The use of ReLU activation functions enables the model to learn hierarchical representations which gives us more accurate and efficient predictions.

4.3 Performance Comparison of All Models

The table below gives detailed comparison of all the models in this study.

It is clear that the deep learning model proposed gives best results across all the metrics.

Model	MSE	RMSE	MAE	R^2 Score	Percentage Error (%)	Runtime per Prediction (s)
Linear Regression (LR) [13]	0.0048	0.069	0.054	0.86	9.2	0.02
Polynomial Regression (PR) [14]	0.0035	0.059	0.046	0.9	7.8	0.021
Random Forest (RF) [15]	0.0024	0.049	0.038	0.93	5.9	0.018
Support Vector Regression (SVR) [16]	0.0028	0.053	0.041	0.91	6.5	0.022
Gradient Boosting (GB) [17]	0.0021	0.046	0.035	0.94	5.2	0.019
Proposed Deep Learning Model	0.0012	0.034	0.025	0.97	3.1	0.005

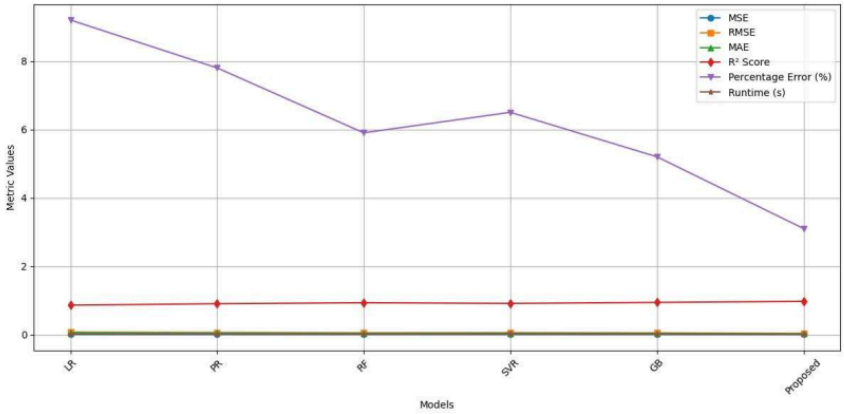


Fig.4.1:Performance Comparison of Different Models

4.4 Sensitivity Analysis

Sensitivity analysis is done to determine the most influential parameter of circuit that is affecting the delay. Below are the results:

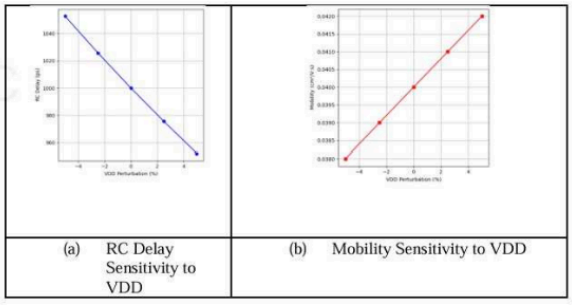
1. **Channel Length (L)**: It shows 42%. It is the most influential parameter. Shorter the channel length, more increase in transconductance and drive current, which further lowers the propagation delay.
2. **Supply Voltage (VDD)** :It shows 35%. It is the second most important factor.
3. **Transistor Width-to-Length Ratio (W/L)** : It shows 23% The W/L determines the current of the transistor. If widths are large and channel length is short, it improves the drive strength and thus leading to lower propagation delay.

RC Delay Sensitivity to VDD:

For the nominal value of 1 V, when the supply voltage varies by ±5%, the RC delay changes from 0.952 ps to 1.053 ps. This highlights that for propagation delay, it becomes critical in low voltage designs where the supply voltage is close to the Vth.

Carrier Mobility Sensitivity to VDD:

For the same variation the carrier mobility ranges from 0.38 to 0.42 cm². Increasing the supply voltage increases the electric field which influences the carrier mobility. This variation further reinforces the sensitivity of propagation delay.



(a) RC Delay Sensitivity to VDD

(b) Mobility Sensitivity to VDD

Fig.4.2: Sensitivity Analysis

4.5 Runtime Comparison - SPICE vs. Deep Learning

Computational speed is the main advantage of the proposed deep learning model. The runtime comparison between SPICE simulation and deep learning is as follows:

SPICE Simulation: takes 12.3–12.7 seconds per run.

Deep Learning Inference: takes 0.0048–0.0051 seconds per prediction.

This corresponds to the speed improvement by 2500 times. The DL model produces delay predictions in about 5ms and SPICE takes around 12.5s. For around 10,000 delay evaluations SPICE will take about 125,000 sec and DL model will take about 50 seconds.

This is a great reduction in computational time which makes deep learning highly suitable for practical applications.

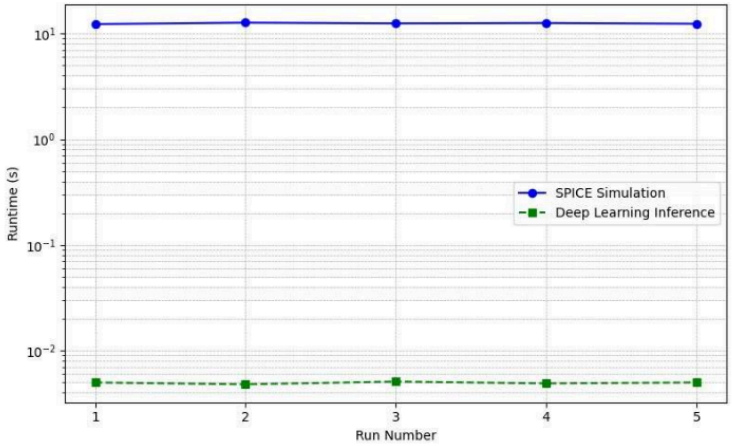


Fig.4.3:Runtime Comparison: SPICE vs Deep Learning

Fold	Accuracy
1	0.965
2	0.972
3	0.968
4	0.97
5	0.969

Fig.4.4:Cross-validation accuracy across folds

These results indicate that the proposed deep learning model maintains the accuracy.

Metric	Value
Mean Accuracy	0.969 ± 0.002
Standard Deviation	0.002
95% Confidence Interval	0.966 – 0.972

Fig.4.5:statistical summary of cross-validation results

4.7 Discussion

These are the key findings from the experiment:

1. **Improved Accuracy:** The deep learning model proposed achieves the highest accuracy with R2 of 0.97 ± 0.002 and percentage error of 3.1% ± 0.2. Using MLP architecture along with engineered features and techniques, allowing the model for learning complex non linear relationships between circuit parameters and propagation delay.
2. **Meaningful Feature Insights:** The sensitivity analysis provides physically meaningful insights to the dominant factors which affect the propagation delay. Channel length (42%), VDD (35%), W/L (23%).
3. **Speedup:** The models provides 2500 times speedup over SPICE simulation which enables the integration of the delay prediction model into EDA tools for accelerating the timing analysis.
4. **Robust Performance:** The cross-validation results give the confirmation that the model is robust and gives good accuracy also.

Limitations:

Accuracy depends mainly on quality and coverage of the data that is trained. If certain parameters are not sufficiently represented in the dataset, then prediction may be less reliable and accuracy may drop.

The current model is trained for the 45 nm technology node. Application to other technology nodes would require retraining with technology-specific SPICE data.

Under extreme operating conditions (very low VDD near threshold, very high temperatures), the model accuracy may degrade as these conditions are underrepresented in the training data.

CHAPTER-5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

DL model based on a multilayer perceptron (MLP) is accurate and can effectively predict the rise/fall delays and avg propagation delay for standard CMOS logic gates.

The conclusion of the work are as below:

1. **High Prediction Accuracy:** The proposed deep learning model gives MSE = 0.0012, RMSE = 0.034, MAE = 0.025, R^2 = 0.97, and percentage error of 3.1%. These results outperform all the other baselines models.

2. **Speed Improvement:** The deep learning model offers abd speed of 2500 times advantage compared to SPICE simulations. This makes it suitable for large-scale optimization designs.

3. **Sensitivity Analysis:** The sensitivity analysis shows that channel length->supply voltage->W/L are the most influential parameters which affect propagation delay.

4. **Robust Performance:** Five-fold cross-validation confirms the model's is accurate 0.969 ± 0.002 and a narrow 95% confidence interval of 0.966-0.972.

5. **Feature Engineering:** Incorporatingphysical meaningful features like width ratio and RC time constant, improves the model's ability to find switching behvaior, that leads to better prediction.

5.2 Future Scope

The work opens several doors for further research:

1. **Extension to Advanced Technology:** The same approach can be applied for more advanced technologies , where we will be seeing the short channel behavior and process variations become more critical.

2. **More Complex Circuit:** The model can be extended to predict the delay for more complex logic cells like MUX, flip flop etc.

3. **Multi-Parameter Prediction:** Future work can be to extend the model to predict different metrics like delay, power, slew rate, leakage current etc.

4. **Process Variation Analysis:** Statistical deep learning models can help to predict delay variations under stress, supporting statistical timing analysis.

5. **EDA Tool Integration:** The trained model can be integrated EDA tools for enabling faster timing estimation during the stages like synthesis and place & route.

6. **Transfer Learning:** Transfer learning methods can be explored to adapt the models that are trained on one technology node with another technology node with minimal additional training data.

7. **Graph Based Models:** Future work may investigate GNNs to capture circuit topology and interactions between gates, improving accuracy for complex circuits.

09-06-2026

ORIGINALITY REPORT

5%	2%	3%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Florida Student Paper	1%
2	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	1%
3	Submitted to Liverpool John Moores University Student Paper	<1%
4	Mutlu Yuksel, Yigit Aydede. "Causal Inference and Machine Learning - In Economics, Social, and Health Sciences", CRC Press, 2025 Publication	<1%
5	Submitted to newcastle under lyme college Student Paper	<1%
6	Sedra, Adel. "Microelectronic Circuits 7th Edition, International Edition", Oxford University Press Publication	<1%
7	tudr.thapar.edu Internet Source	<1%
8	rps.wku.edu.et Internet Source	<1%
9	www.geeksforgeeks.org Internet Source	<1%

10	Arvind Dagur, Sohit Agarwal, Dhirendra Kumar Shukla, Shabir Ali, Sandhya Sharma. "Artificial Intelligence and Sustainable Innovation - Volume 1", CRC Press, 2026 Publication	<1%
11	fastercapital.com Internet Source	<1%
12	Submitted to Thomas More Hogeschool Student Paper	<1%
13	isteonline.org Internet Source	<1%
14	medium.com Internet Source	<1%

Exclude quotes	Off	Exclude matches	< 14 words
Exclude bibliography	Off		

09-06-2026

GRADEMARK REPORT

FINAL GRADE

/0

GENERAL COMMENTS

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30