IR drop Prediction Based on Machine Learning and Pattern Reduction

Yong-Fong Chang[1], Yung-Chih Chen[2], Yu-Chen Cheng[1], Shu-Hong Lin[2], Che-Hsu Lin[2], Chun-Yuan Chen[1], Yu-Hsuan Chen[1], Yu-Che Lee[1], Jia-Wei Lin[3], Hsun-Wei Pao[3], Shih-Chieh Chang[1], Yi-Ting Li[1], and Chun-Yao Wang[1]

[1]National Tsing Hua University, Taiwan, ROC
[2]National Taiwan University of Science and Technology, Taiwan, ROC
[3]MediaTek, Taiwan, ROC

***Abstract*—With the advances in semiconductor technology, the sizes of transistors are getting smaller, which has led to an increasingly severe impact of IR drop. Consequently, this trend has amplified the significance of IR drop analysis within the realm of chip design. However, analyzing IR drop is resource-intensive and time-consuming, since numerous simulation patterns are required to verify the power integrity of circuits. Additionally, with every engineering change order (ECO) step, a reevaluation is necessary. In this paper, we propose a machine learning-based method to predict IR drop levels and present an algorithm for reducing simulation patterns, which could reduce the time and computing resources required for IR drop analysis within the ECO flow. Experimental results show that our approach can reduce the number of patterns by approximately 50%, thereby decreasing the analysis time while maintaining accuracy.**

I. Introduction

With the advances in semiconductor technology, the transistor dimensions are getting smaller, which has led to a reduction in size and increased portability of electronic products in our daily lives. However, this size reduction has brought some challenges in IC design, for example, the electric field effect, the increasing complexity of semiconductor manufacturing, etc. *IR drop* issue is also a critical problem to be dealt with before design signoff [2].

IR drop refers to the voltage drop that occurs when current flows through the power delivery network. As semiconductor manufacturing processes advance, the narrowing of metal wire widths leads to increased wire resistance. Consequently, the problem of voltage drop becomes more severe.

IR drop harms the performance and reliability of the circuit. Excessive IR drop can lead to a downgrade in the operating voltage of the circuit, resulting in slower circuit operations. Furthermore, IR drop raises the chip's temperature due to increased resistance-generated heat, leading to reduced circuit performance and reliability. Hence, IR drop signoff has become an increasingly critical step in the design flow, especially for high-end products using advanced manufacturing processes.

There are two approaches for analyzing a design's IR drop levels: static analysis [1] and dynamic analysis [10]. Static analysis involves using static circuit analysis to determine voltage drop in the circuit. It calculates the current and resistance in each wire to determine the voltage drop. Since static analysis does not involve intensive computation, it costs less CPU time, but the IR drop analyzed from static analysis cannot represent the actual IR drop in the design.

On the other hand, dynamic analysis of IR drop involves using dynamic circuit analysis to determine voltage drop. This method simulates the circuit with patterns over time and analyzes the voltage drop that occurs under the different operations of the circuit. Hence, it is more computation-intensive and yields more precise estimations [7]. Although commercial tools for IR drop analysis, e.g., Redhawk-SC [13] and Voltus [14], are available, they are time-consuming. Having an efficient approach to IR drop analysis with accurate results is desired.

Vector-based dynamic IR drop analysis usually requires significant time and computing resources due to numerous simulation patterns [7]. In IR drop analysis, a pattern refers to a sequence of input vectors used to simulate a circuit's behavior over a specified time frame. A typical simulation pattern has a duration of 300ns and is divided into multiple 10ns slices. Both 300ns patterns and 30ns slices are essential for our analysis because they represent different time slots for IR drop assessment. Typically, numerous 300ns patterns need to be analyzed, and analyzing a 30ns slice may take up to 6.5 hours with commercial tools. Additionally, in the current design flow, after completing an Engineering Change Order (ECO) task, we must reanalyze all the IR drop patterns. This underscores the critical importance of IR drop signoff.

To reduce the time required for IR drop analysis, some previous works [4]–[9], [11], [12] estimated IR drop based on machine learning (ML) techniques. Experimental results revealed that using ML models to predict IR drop is highly feasible.

In this paper, we build an ML model to predict the IR drop in a circuit. Furthermore, we identify key characteristics within simulation patterns and introduce an algorithm aimed at minimizing the number of patterns necessitating simulation. Our approach empowers ECO tasks to streamline the analysis by focusing on the chosen patterns, which could decrease the time required for IR drop signoff.

The contributions of this work are twofold: 1.) We propose

a novel IR drop signoff methodology that reduces the number of patterns required for IR drop analysis, thereby reducing the overall effort of the signoff process. 2.) This is the first work that considers the package effect on IR drop analysis. The proposed new features related to the package elevates the accuracy of the predicted results.

The rest of the paper is organized as follows. Section II introduces the background related to IR drop and ML models. Section III reviews the previous works on IR drop analysis. Section IV presents the proposed approach. Section V shows the experimental results. Finally, Section VI concludes this work.

II. Background

A. IR drop

IR drop refers to the voltage drop that occurs in the power delivery network of an integrated circuit (IC) due to the resistance of interconnects and active devices.

When current (I) flows through an interconnect or a device, the resistance (R) causes a voltage drop (V), as described by Ohm's Law (V = IR). This voltage drop results in a decrease in the operation voltage of transistors in the circuit, which may cause functional failure, timing violation, or reliability degradation of a design.

IR drop can be classified into two types: static and dynamic. The IR drop analyzed by static analysis is called static IR drop. Static IR drop occurs in a steady state when the current flows through interconnects under a stable active device. Static IR drop focuses on the impact of the design's power delivery network, power domain partitioning, and wiring. Dynamic IR drop is analyzed by dynamic analysis. Dynamic IR drop occurs when the large current flows through the power network due to the high switching activity of the cell. This transient current induces additional voltage drops and leads to dynamic IR drop. High switching activity and the corresponding large transient current usually cause severe dynamic IR drop. Hence, dynamic IR drop has a significant impact on timing and reliability of circuits. Dynamic IR drop focuses on power management, timing optimization, and the effect of power supply noise. Hence, in this work, we focus on dynamic IR drop.

B. ML model

eXtreme Gradient Boosting (XGBoost) [3] is a machine learning model for supervised learning problems, e.g, classification and regression. It is an ensemble learning method that combines the results of multiple decision trees. XGBoost uses a gradient-boosting framework, which means that it iteratively builds models to reach the final model. It starts with a simple model and then adds more complex models to correct the errors made by the previous models. XGBoost provides a way to assess the importance of different features in making predictions, which is valuable for feature selection. Compared to other types of models, XGBoost can build models and make predictions more quickly. In this work, we use XGBoost as our ML model.

III. Related works

In this section, we review some related works on IR drop prediction and address their issues for exploring potential improvements to enhance the model's accuracy.

In the previous works [4]–[9], [11], [12], the authors did not consider the package effect on IR drop prediction. In fact, the package of IC is an important factor affecting IR drop. After conducting the same preliminary experiments associated with the commercial tool, we observed a notable increase in IR drop when accounting for the package effect, as compared to when it was not taken into consideration. The main reason for this phenomenon would be the resistance and inductance of the package. Therefore, we will incorporate this information into the features of our model in this work.

Resistance is definitely an important factor for IR drop. [12] assumed that the power delivery network (PDN) is uniform such that the resistance is uniformly distributed throughout the design. Therefore, resistance calculation is not necessary. [4], [5], [8], [9] calculates Euclidean distance from instances to power pads to estimate resistance. [7] [11] [6] consider effective resistance as a feature in their models, but the calculated resistance is not precise enough. In this paper, we calculate the effective resistance with a commercial tool to obtain a more accurate resistance. In addition to the effective resistance, we also consider the resistance of the least-resistance path (RLRP) as a feature of our model, providing comprehensive information about resistance in a design.

Although ML approaches significantly reduce the time spent on analyzing IR drop, an error exists in between the predicted IR drop and the golden IR drop. Therefore, we cannot entirely rely on the predicted IR drop. However, we can exploit the predicted IR drop to filter out the slices that may cause critical instances such that the impact of error can be mitigated. [5] used the predicted IR drop to select some slices, which are called critical slices, that may cause critical instances. Nonetheless, for front-end designers, each pattern has its specific importance, and selecting some slices arbitrarily from each pattern is meaningless for just reducing the time cost. Therefore, we propose an algorithm to select fewer patterns, thereby the time cost required for running IR analysis in each ECO process is greatly reduced.

IV. Proposed Approach

The overall flow of the proposed approach is shown in Fig. 1. When a design is completed, designers release patterns for design sign off. We randomly select some patterns as training data. First, we use commercial tools, e.g., Voltus, to extract features from these patterns. Then, we train our model, which is a two-level XGBoost model, to improve the accuracy of IR drop prediction.

Once we obtain the predicted IR drop from the model, we can perform the pattern selection algorithm. The combination of the golden IR drop and the predicted IR drop is the input of the pattern selection algorithm. Then, we preserve patterns that cause critical IR drops, called critical patterns, and discard the other patterns.

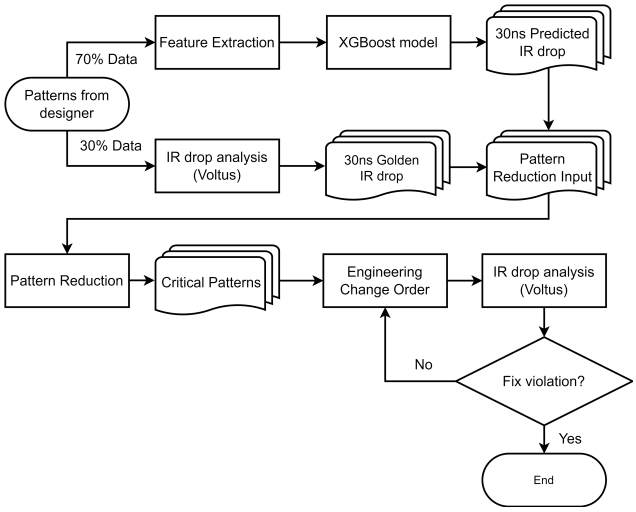

Fig. 1: The overall flow of proposed approach.

A. Feature Extraction

Our model considers two categories of features, **instance-based features** and **tile-based features**, with 56 features in total. Table I lists the features used in our model.

Instance-based feature: We use two commercial tools to generate instance-level information from designs. We use the automatic placement & routing (APR) tool to obtain the physical location information of cells (#1). Additionally, we use Voltus to get the values of effective resistance (#2) and RLRP (#3). Both effective resistance and RLRP have three features, which are the resistance at the power supply terminal, the resistance at the ground terminal, and both. We get bumps' resistance (#4) and inductance (#5) from the package information files. Like the effective resistance, each bump resistance has three features since each instance has a bump at the power supply terminal and a bump at the ground terminal. Besides, we use power information and the toggle rate obtained from Voltus as our features.

There are three types of power consumption in devices, i.e., internal power (P_i, #6), switching power (P_s, #7), and leakage power (P_l, #8). Internal power refers to the power consumed by the active components of a circuit during operation. Switching power, also known as dynamic power, is consumed during the signal transition of circuits. Leakage power is consumed in a circuit in the standby or idle state. It comes from the small leakage currents in transistors. As transistor sizes continue to shrink, leakage power has significantly contributed to overall power consumption.

In addition to these three components of power information generated from Voltus, we also refer to the work [5] and add four power features: total power (P_t, #9), scaled power (P_{Scaled}, #10), overlapped-switching power (#11), and overlapped-scaled power (#12), into our model. The formulas for total power and scaled power are shown in Table I. Formula

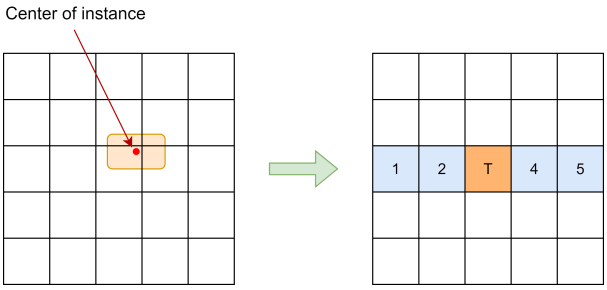

Fig. 2: The target tile of each instance.

(1) is the formula of scaled power in [5].

$$P_{Scaled} = (P_i + P_s) \times \tau + P_l \quad (1)$$

We modify this formula to reflect the actual situation as shown in Formula 2.

$$P_{Scaled} = P_i \times \tau_i + P_s \times \tau_o + P_l \quad (2)$$

Overlapped power is obtained by summing up the power of the nine surrounding tiles of an instance. Using this, the model can be trained to learn the local effect of IR drop.

The toggle rate (#13, #14) is the rate that a signal changes its state from low to high or high to low. It quantifies the frequency of transitions occurring in a circuit and is measured as the average number of signal transitions per clock cycle. The toggle rate consists of the input toggle rate and output toggle rate.

Tile-based feature: Tile-based features are converted from instance-based features. The tile-based features are inspired by the density map feature in [2]. We improved the feature for representing the information more effectively. The features that require conversion are five types of power ($P_i, P_s, P_l, P_t, P_{Scaled}$) and toggle rate. We have to determine each instance's target tile before performing the conversion. In this work, the selection of the target tile is based on the center point of each instance like Fig. 2.

Each instance includes information from five neighboring tiles as shown in Fig. 2, allowing the model to learn the local effect of IR drop. For the conversion, we sum up the powers and toggle rates of all the instances in the tile. If there is an instance across multiple tiles, the value is calculated according to the cross-area ratio. Fig. 3 is an example of conversion,

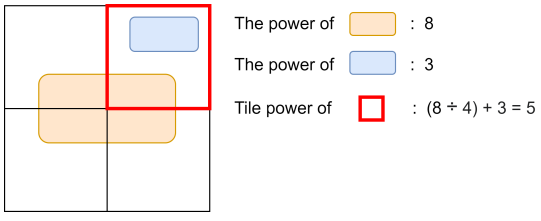

Fig. 3: The example of tile-based feature conversion.

In summary, we use 56 features, including 21 instance-based features and 35 tile-based features. Since we utilize

TABLE I: The features for training our model.

#	Feature Set	Extraction Method	Remark
Physical Features			
1	Cell physical location (X, Y)	Extracted from APR tool	
2	Effective resistance (Reff)	Extracted from Voltus	
3	Shortest path resistance (RLRP)		
4	Bump resistance	Extracted from package information files	
5	Bump inductance		
Power Features			
6	Internal power	Extracted from Voltus	
7	Switching power		
8	Leakage power		
9	Total power	$Internal\ power + Switching\ power + Leakage\ power$	
10	Scaled power	$Internal\ power \times \tau_i + Switching\ power \times \tau_o + Leakage\ power$	
11	Overlapped-switching power	Summation of neighboring cells' switching power	
12	Overlapped-scaled power	Summation of neighboring cells' scaled power	
Timing Features			
13	Input toggle rate (τ_i)	Toggle rate of the input pin	
14	Output toggle rate (τ_o)	Toggle rate of the output pin	

Tile processing:
- Accumulate up the power of all the instances within the tile.
- For the instance that crosses different tiles, its power is calculated based on the area ratio.

Overlap power processing:
- Accumulate up the power within the neighboring 3x3 tiles.

Tile + Instance-based features
Instance-based features

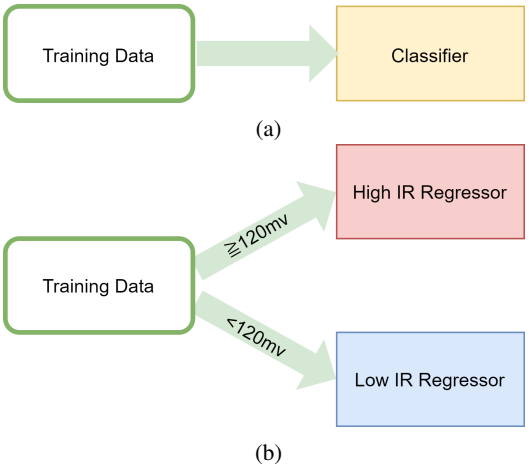

Fig. 4: Training phase of the proposed two-level model. (a)Training classifier. (b)Training regressors.

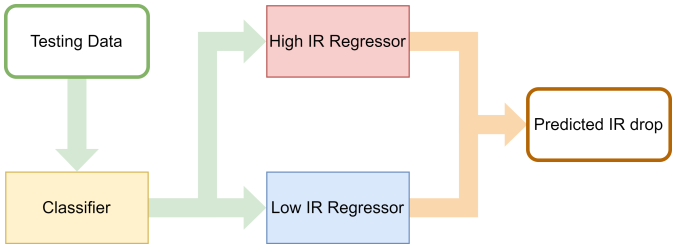

Fig. 5: Inference phase of the proposed two-level model.

the XGBoost model, which operates in a tabular manner for predictions, the output is also an instance-based IR drop.

B. XGBoost Model

We build a two-level XGBoost model for training as depicted in Fig. 4. We first train a classifier to distinguish between high and low IR drop instances. In the second level of the model, we train two regressors, one for predicting high IR drop, and the other is for predicting low IR drop. We define the threshold for high IR drop as 120mV, which is about 15% of the supply voltage.

In the inference phase, as shown in Fig. 5, the testing data is first passed through the classifier to classify the instances into respective groups. Then, the corresponding regressor is selected to predict the accurate IR drop.

C. Pattern Reduction

In the industrial flow, the simulation patterns provided by the front-end designers are assigned to each sign off team. However, only a few of these patterns are IR-critical patterns that can cause high IR drop. The purpose of pattern reduction is to identify these IR-critical patterns.

Fig. 6 shows the flow of the algorithm. The training data we use is extracted from the 30ns slices, and thus the predicted IR drop output from the model corresponds to the 30ns slices. We combine the 30ns labeled IR drop and the 30ns predicted IR drop, then use them as inputs for pattern reduction.

First, we calculate the critical instances coverage score for each non-selected slice. The critical instances refer to those instances in which the IR drop exceeds the threshold. The scoring algorithm unifies the critical instances of the slice with the selected slices. Next, we use a greedy approach to select the slices with the highest scores as candidates. Among these candidates, we choose the slice that has the most relevance to the selected slices, and add them to the selected slices. We also update the coverage score accordingly. We repeat the process until the coverage of critical instances reaches 90%.

Table II shows an example, in which there are three patterns and a total of 7 slices, and the candidate count is set to 3. In the first iteration, the three candidates with the highest coverage scores are "1^{st}_030", "3^{rd}_030", and "1^{st}_060". Thus, we choose "1^{st}_030" in this iteration.

In the second iteration, based on the updated scores, the top three candidates are "3^{rd}_030", "2^{nd}_030", and "1^{st}_060". Although "3^{rd}_030" has the highest score, "1^{st}_060" is prioritized since it belongs to the same pattern as the previously selected slice. Then, the overall score is updated to 70.

In the third iteration, the highest scores are "3^{rd}_030", "3^{rd}_060", and "2^{nd}_030". None of these three candidates is related to the selected slices. Therefore, we choose "3^{rd}_030", which has the highest score. After these iterations, the score reaches 95, surpassing the threshold of 90, and the algorithm

TABLE II: An example for pattern reduction.

Total Coverage Score : 0/100 Chosen Slices : none

Slice	Critical Instance	Update Score	Candidate	Choose?
1^{st}_030	50	50	1	√
1^{st}_060	40	40	3	
1^{st}_090	30	30		
2^{nd}_030	25	25		
2^{nd}_060	35	35		
3^{rd}_030	45	45	2	
3^{rd}_060	35	35		

(a) First iteration.

Total Coverage Score : 50/100 Chosen Slices : 1^{st}_030

Slice	Critical Instance	Update Score	Candidate	Choose?
1^{st}_030	50	-	-	√
1^{st}_060	40	70	3	√
1^{st}_090	30	60		
2^{nd}_030	25	70	2	
2^{nd}_060	35	65		
3^{rd}_030	45	75	1	
3^{rd}_060	35	65		

(b) Second iteration.

Total Coverage Score : 70/100 Chosen Slices : 1^{st}_030, 1^{st}_060

Slice	Critical Instance	Update Score	Candidate	Choose?
1^{st}_030	50	-	-	√
1^{st}_060	40	-	-	√
1^{st}_090	30	75		
2^{nd}_030	25	85	3	
2^{nd}_060	35	80		
3^{rd}_030	45	95	1	√
3^{rd}_060	35	90	2	

(c) Third iteration.

Total Coverage Score : 95/100 Chosen Slices : 1^{st}_030, 1^{st}_060, 3^{rd}_030

Slice	Critical Instance	Update Score	Candidate	Choose?
1^{st}_030	50	-	-	√
1^{st}_060	40	-	-	√
1^{st}_090	30			
2^{nd}_030	25			
2^{nd}_060	35			
3^{rd}_030	45	-	-	√
3^{rd}_060	35			

Output Critical Pattern : 1, 3

(d) Final result.

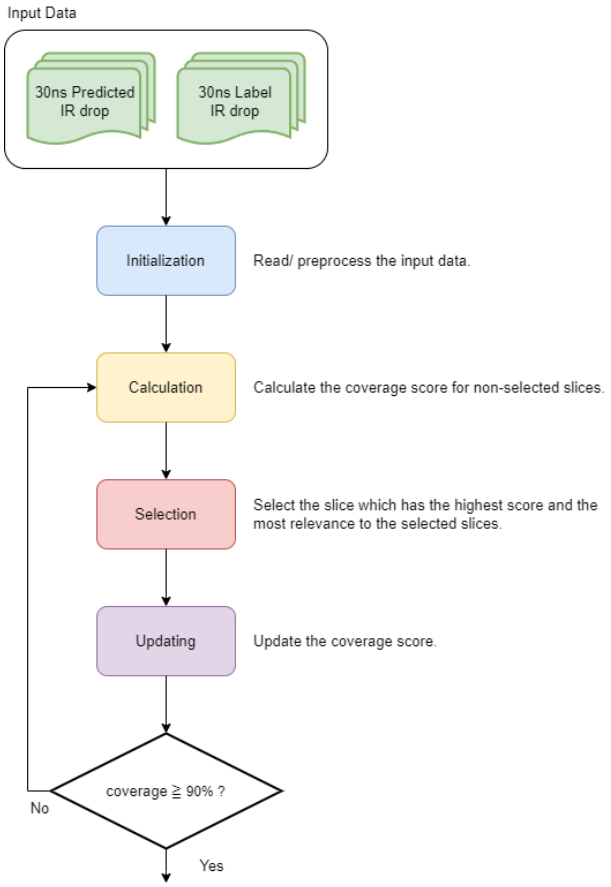

Fig. 6: The flow of pattern reduction.

TABLE III: The profile of design.

Design	$\|Instance\|$	$\|Pattern\|$	$\|TotalSlice\|$
Design 1	3508819	9	76

is terminated.

Finally, the selected slices are "1^{st}_030", "1^{st}_060", and "3^{rd}_030". Thus, the chosen patterns are the 1^{st} pattern and the 3^{rd} pattern.

V. Experimental Results

We conducted the experiments on an industrial design with 9 simulation patterns, which can be divided into 76 30ns slices. The profile of the design is shown in Table III. The supply voltage is 0.9V and the process technology is TSMC 5nm process. The length and width of a tile are twice the maximum instance length and width. The model is trained by using golden instance IR drop labels obtained from Voltus. To evaluate the performance, we used *Root Mean Square Error* (RMSE) and *Mean Absolute Error* (MAE), which are defined in Formula (3) and (4). In these formula, $\hat{y_i}$ is the golden IR drop of i^{th} instance, and y_i is the predicted IR drop of i^{th} instance. N is the total number of instances.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y_i})^2}{N}} \tag{3}$$

$$MAE = \frac{\sum_{i=1}^{N}|\hat{y_i} - y_i|}{N} \tag{4}$$

$$CC = \frac{\sum_{i=1}^{N}[\hat{y_i} - mean(\hat{y})]\,[y_i - mean(y)]}{\sqrt{\sum_{i=1}^{N}[\hat{y_i} - maen(\hat{y})]^2[y_i - mean(y)]^2}} \tag{5}$$

The correlation coefficient (CC) is represented by Formula (5) and takes on values between -1 and 1. A value of 1 indicates

a perfect positive correlation, -1 indicates a perfect negative correlation, and 0 indicates no correlation.

A. IR Drop Prediction Results

The results are shown in Table IV. We used 30% of the slices for training and the other 70% for testing. To validate whether our model can successfully predict the IR drop, we attempted different combinations of training and testing data. For Set A, we randomly select 23 slices out of the 76 slices as the training data. For Set B, we randomly select 3 patterns out of the 9 patterns, and use the slices of the selected patterns as the training data. Finally, for Set C, we select 23 slices that have a better balance of high IR drop and low IR drop as the training data. For the best case, the MAE is 3.954mV and the RMSE is 5.462mV. The CC is 0.95, which means that our predicted results are similar to the label.

TABLE IV: The result of IR drop prediction by XGBoost.

Design 1		MAE	RMSE	MaxE	MinE
Set A	Normal	4.983	6.336	97.185	-139.944
	Best	4.970	6.265	88.279	-63.153
Set B	Normal	4.356	5.940	105.683	-145.045
	Best	4.344	5.854	93.665	-67.404
Set C	Normal	3.968	5.556	120.747	-139.504
	Best	3.954	5.462	93.486	-67.441

For the CPU time, we only spent about 1300 seconds for training, and about 2300 seconds for inference. The total CPU time is about 1 hour.

B. Pattern Reduction Results

To evaluate the effectiveness of pattern reduction, we establish the following criteria. We only select up to half of all patterns, and the critical instance coverage of the selected patterns must be greater than 90%. We use the Brute-force algorithm to find the golden critical pattern. By calculating the critical instance coverage for each combination of patterns, we select the one with the highest coverage as the golden pattern.

Table V shows the results of pattern reduction. We adjust the number of the candidates (N) in the algorithm for the experiments. When the threshold of critical IR drop is 120mV, the golden critical patterns are (1, 2, 3, 7) or (1, 2, 3). Our experimental results show that our algorithm can precisely identify the same critical patterns. Specifically, when we set the value of N to 5 and 7, our algorithm chooses three patterns, (1, 2, 3), which is the same as the golden critical patterns. When we set the value of N to 1 and 3, our algorithm chooses four patterns, (1, 2, 3, 7), which is the same as the golden critical patterns.

To show the scalability of this method, we modified the threshold of critical IR drop. When we set the threshold to 100mV and 80mV, our algorithm can choose the same patterns as the golden critical patterns.

For the CPU time of our pattern reduction algorithm, we need about 600 seconds; but for the Brute-force algorithm, it cost 2 hours to identify the critical patterns. Hence, our algorithm can efficiently and effectively identify critical patterns for IR drop analysis.

TABLE V: The result of IR drop prediction by XGBoost.

Threshold	Golden Patterns	$\|Candidate\|$	$\|Slice\| \geq 90\%$	Chosen Patterns
120mV	(1, 2, 3, 7)/ (1, 2, 3)	1	11	1,2,3,7
		3	11	1,2,3,7
		5	11	1,2,3
		7	11	1,2,3
100mV	(1, 2, 3, 7)/ (1, 2, 3)	1	11	1,2,3,7
		3	11	1,2,3,7
		5	11	1,2,3
		7	11	1,2,3
80mV	(1, 2, 3, 7)/ (1, 2, 3)	1	10	1,2,3
		3	10	1,2,3
		5	10	1,2,3
		7	10	1,2,3

VI. Conclusion

In this paper, we propose the first IR drop analysis approach considering package, which improves the accuracy of IR drop prediction. Additionally, we propose an algorithm to identify critical patterns, which significantly reduces the time for IR drop analysis.

References

[1] S. Abazyan and N. Mamikonyan, "Static ir drop estimation on the power network," in *Open Access Library Journal*, vol. 7, no. 01, pp. 1–7, 2020.

[2] J.-X. Chen, S.-T. Liu, Y.-T. Wu, M.-T. Wu, C.-M. Li, N. Chang, Y.-S. Li, and W.-T. Chuang, "Vector-based dynamic ir-drop prediction using machine learning," in *Proc. Asia and South Pacific Design Automation Conference*, pp. 202–207, 2022.

[3] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proc. acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

[4] V. A. Chhabria, V. Ahuja, A. Prabhu, N. Patil, P. Jain, and S. S. Sapatnekar, "Thermal and ir drop analysis using convolutional encoder-decoder networks," in *Proc. Asia and South Pacific Design Automation Conference*, pp. 690–696, 2021.

[5] V. A. Chhabria, Y. Zhang, H. Ren, B. Keller, B. Khailany, and S. S. Sapatnekar, "Mavirec: Ml-aided vectored ir-drop estimation and classification," in *Proc. Design, Automation & Test in Europe Conference & Exhibition*, pp. 1825–1828, 2021.

[6] C.-T. Ho and A. B. Kahng, "Incpird: Fast learning-based prediction of incremental ir drop," in *Proc. IEEE/ACM International Conference on Computer-Aided Design*, pp. 1–8, 2019.

[7] Y. Kwon, G. Jung, D. Hyun, and Y. Shin, "Dynamic ir drop prediction using image-to-image translation neural network," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1–5, 2021.

[8] S.-Y. Lin, Y.-C. Fang, Y.-C. Li, Y.-C. Liu, T.-S. Yang, S.-C. Lin, C.-M. Li, and E. J.-W. Fang, "Ir drop prediction of eco-revised circuits using machine learning," in *Proc. IEEE VLSI Test Symposium*, pp. 1–6, 2018.

[9] S. N. Mozaffari, B. Bhaskaran, K. Narayanun, A. Abdollahian, V. Pagalone, S. Sarangi, and J. E. Colburn, "An efficient supervised learning method to predict power supply noise during at-speed test," in *Proc. IEEE International Test Conference*, pp. 1–10, 2019.

[10] S. Nithin, G. Shanmugam, and S. Chandrasekar, "Dynamic voltage (ir) drop analysis and design closure: Issues and challenges," in *Proc. International Symposium on Quality Electronic Design*, pp. 611–617, 2010.

[11] C.-H. Pao, A.-Y. Su, and Y.-M. Lee, "Xgbir: An xgboost-based ir drop predictor for power delivery network," in *Proc. Design, Automation & Test in Europe Conference & Exhibition*, pp. 1307–1310, 2020.

[12] Z. Xie, H. Ren, B. Khailany, Y. Sheng, S. Santosh, J. Hu, and Y. Chen, "Powernet: Transferable dynamic ir drop estimation via maximum convolutional neural network," in *Proc. Asia and South Pacific Design Automation Conference*, pp. 13–18, 2020.

[13] *RedHawk User Manual*, Ansys, 2018.

[14] *Voltus User Manual*, Cadence.