Vector-based Dynamic IR-drop Prediction Using Machine Learning

Jia-Xian Chen[1], Shi-Tang Liu[1], Yu-Tsung Wu[1], Mu-Ting Wu[1], Chieo-Mo Li[1], Norman Chang[2], Ying-Shiun Li[2], Wen-Tze Chuang[2]
[1]*Graduate Institute of Electronics Engineering*
National Taiwan University, Taiwan
[2] *Ansys Inc.*
{jason12399, rbufghj9713, elijah5wu}@gmail.com, {r08943094, cmli}@ntu.edu.tw
{Norman.Chang, Ying-Shiun.Li, WenTze.Chuang}@ansys.com

2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC) | 978-1-6654-2135-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ASP-DAC52403.2022.9712489

Abstract —
Vector-based dynamic IR-drop analysis of the entire vector set is infeasible due to long runtime. In this paper, we use machine learning to perform vector-based IR drop prediction for all logic cells in the circuit. We extract important features, such as toggle counts and arrival time, directly from the logic simulation waveform so that we can perform vector-based IR-drop prediction quickly. We also propose a feature engineering method, *density map*, to increase correlation by 0.1. Our method is scalable because the feature dimension is fixed (72), independent of design size and cell library. Our experiments show that the mean absolute error of the predictor is less than 3% of the nominal supply voltage. We achieve more than 495 speedups compared to a popular commercial tool. Our machine learning prediction can be used to identify IR-drop risky vectors from the entire test vector set, which is infeasible using traditional IR-drop analysis.

Keywords— Dynamic IR-drop, machine learning

I. INTRODUCTION

As the process scales down, IR-drop has become an important issue for design signoff. Excessive IR-drop degrades the circuit performance and even leads to functional failure. This is especially a serious problem because some ATPG test vectors cause excessive IR-drop and result in significant yield loss. IR-drop signoff for modern design is therefore required before tape-out. Because there are too many vectors to analyze, designers have no choice but to either perform vector-based analysis on a few vectors or perform *vectorless* analysis. However, vectorless analysis can be pessimistic, and there are no good rules to select vectors for vector-based analysis. According to our observation, IR-drop hotspots occur at different locations for different vectors. Fig. 1 shows two IR-drop maps of the same design (b19). We can see that two vectors result in quite different IR-drop hotspots. Therefore, a fast vector-based IR-drop predictor is highly needed.

In this paper, we propose a machine learning model to predict dynamic IR-drop for every cell. Given a circuit design, we train our machine learning model with a few vectors (less than 20 in our case). Then, we can predict dynamic IR-drop very quickly for the other vectors. In this way, we can perform vector-based IR-drop analysis at a much higher speed compared to the IR-drop analysis of the commercial tool.

The proposed technique has the following advantages. First, our training time is short because our machine learning model is trained with a small portion of vectors. Second, we propose to extract important features from the logic simulation waveform, so we can obtain useful vector-based timing information for dynamic IR-drop prediction. Third, our proposed method is scalable because our feature dimension is independent of design size and cell library. Fourth, our method can predict the dynamic IR-drop for every cell, so it is suitable for different IR-drop signoff criteria. Finally, we predict both ground bounce and IR-drop together. For simplicity, in the rest of this paper, we sum up these two numbers and just call it *IR-drop*.

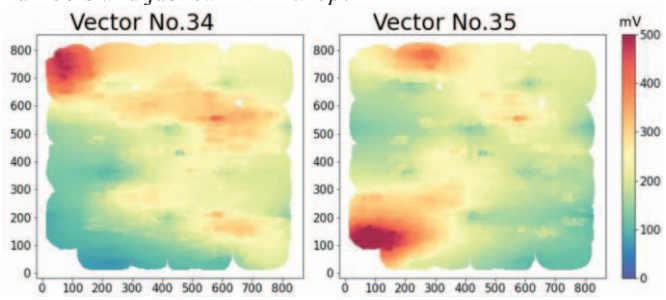

Fig. 1. Very different IR-drop maps of two vectors

We proposed two types of features: *raw features* and *density map features*. We further divide raw features into three categories: *vector-independent features* (VI), *vector-dependent waveform features* (VDW), and *vector-dependent power features* (VDP). VDW and VDP provide important information about vectors so that our model can predict IR-drop for different vectors.

To validate our proposed method, we run experiments on three circuit designs. Our experiments show that the mean absolute error is less than 3% of the nominal supply voltage. We also demonstrate an important application: to identify IR-drop risky ATPG test vectors. We can identify IR-drop risky vectors with at least 495.1 speedups compared with a popular commercial tool. Table I shows the comparison with previous works. Our goal is more difficult

than previous work, but our results and runtime are better or equal to the others.

TABLE I. Comparison with previous works

Work	Types of IR-drop predicted	Predict Target	Max Absolute Error (mV)	Speedup Ratio
[Yamato 12]	Average dynamic IR-drop	Cell	8.9	300
[Dhotre 17]	Dynamic IR-drop	Hot spot	-	-
[Fang 18]	Dynamic IR-drop (vectorless)	Cell	3	300
[Mozaffari 19]	Dynamic IR-drop	Sensor	7.3	-
[Xie 20]	Dynamic IR-drop	Tile	-	30
Ours	Dynamic IR-drop (vector-based)	Cell	26 (multiple vectors), 3 (single vector)	495-778

The rest of the paper is organized as follows. Section II reviews previous researches on IR-drop predictors. Section III reveals proposed techniques, and section IV shows experimental results. Finally, Section V provides our conclusion of this paper.

II. Background

Linear regression was used to predict the average dynamic IR-drop over the launch clock of each cell [1]. They selected three vectors and perform power analysis. Then, they used the average power to build a linear model for every cell to predict the average dynamic IR-drop. Their work showed a good correlation between average dynamic IR-drop and average power, but building a linear model for every cell is time-consuming.

Unsupervised learning was used to identify hot spots of IR-drop [2]. They used layout and power information to create a self-defined index. Then, they clustered cells by the self-defined index. Their work can identify the high IR-drop region but cannot calculate the real IR-drop value of each cell.

Convolutional neural networks and *XGBoost* were used to predict the worst IR-drop of each cell [3]. They used timing, power, and physical information as features. They proposed *neighbor cell features* to include information about the simultaneous switching of neighbor cells. Their work showed good prediction accuracy for *vectorless* IR-drop analysis, but their machine learning model did not consider the effect of different vectors.

Natural language processing-based models were used to identify the dynamic IR-drop of a sensor [4]. After they trained the machine learning model, they can predict the IR-dorp for several similar GPU circuits. However, they can only predict the IR-drop of one sensor on the chip.

A modified convolutional neural network was used to predict the dynamic IR-drop of a tile [5]. They used power information as features. They can predict the IR-drop of a tile at a specific time but cannot predict the IR-drop value of each cell.

In this paper, we use a commercial IR-drop analyzer, *Redhawk-SC* [6], as our golden reference. To perform IR-drop analysis, we provide Redhawk-SC the following data: technology data, design data, and library data. Technology data contain technology node descriptions about the metal layer stack. Design data include a layout file (*DEF*), a pad location file, and a timing file (FSDB). The DEF file contains a physical description of instances, power networks, and other circuit elements. Pad location file contains the physical locations of power and ground sources. The FSDB file records the waveform of each cell. Library data include a *LEF* file and a *liberty* file. The LEF file contains physical descriptions of library cells. The liberty file contains various characteristics of the libraries, which provide the type, functionality, power, and timing of each cell.

In this work, the IR-drop of a cell is defined as the *maximum dynamic V_{DD}-drop* plus *ground bounce* during the *capture cycle* for a given vector. The capture cycle time is set to be 100 ns.

III. Proposed Techniques

A. Proposed Flow

Our proposed flow is shown in Fig. 2. We train a machine learning model in the training phase. In the *Vector selection* step, we select representative vectors as the training vector set. In the *dynamic IR-drop analysis* step, we obtain the accurate IR-drop for every cell with a commercial tool, Redhawk-SC, when a representative vector is applied. As a result, we obtain IR-drop values as labels for training. In the *Raw feature extraction* step, we extract IR-drop-related features from the circuit-related files. In the *density map features creation* step, we create density map features from the raw features for training. Finally, we train our machine learning model with raw features, density map features, and labels.

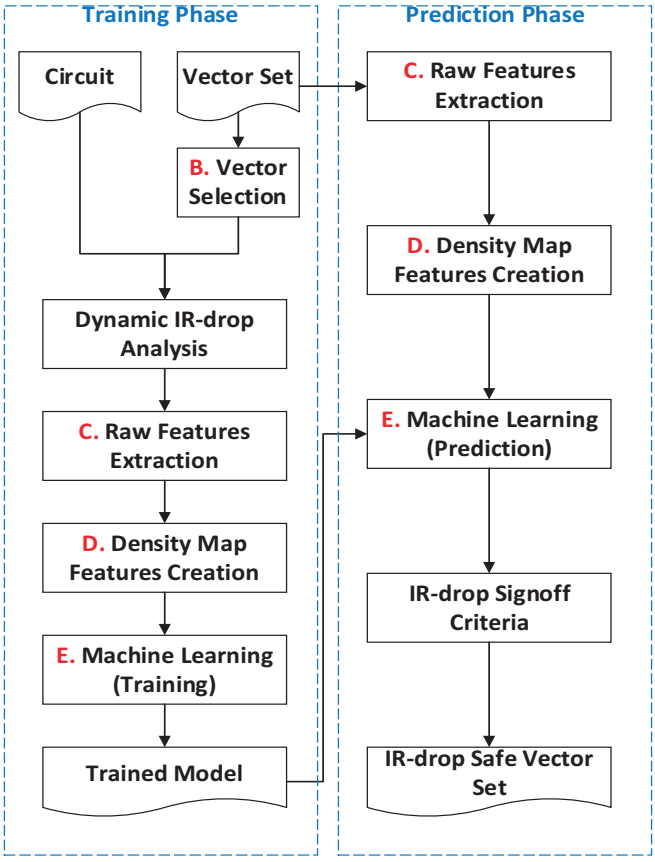

Fig. 2. Our proposed flow

We identify the IR-drop risky vectors with the trained machine learning model in the prediction phase. We perform the same raw feature extraction and create density map features as we do in the training phase for each vector to obtain the features for prediction. We predict the IR-drop for every

cell using the trained model. In the *IR-drop signoff criteria* step, we can identify IR-drop risky vectors based on the predicted IR-drop. After we remove IR-drop risky vectors, we can get the *IR-drop safe vector set*.

B. Vector Selection

To prevent the machine learning model from extrapolation, we need to have vectors of diversified IR-drop. In this paper, we select representative vectors based on the total toggle counts during logic simulation. First, we sort vectors into bins by total toggle counts. Next, we select one representative vector from each bin. As a result, we select representative vectors with various IR-drop severity.

TABLE II. Definition of raw features

Raw Features	Description	Source*	Category**
x, y	Physical location	RH	VI
w, h	Dimension	RH	VI
R_{eff}	Effective resistance	RH	VI
SPR	Shortest path resistance	RH	VI
Cell type	Cell type	RH	VI
P_{leak}	Leakage power	RH	VI
C_{load}	Loading capacitor	RH	VI
Pi_{c1}, Pi_{c2}, Pi_r	Equivalent Pi model	RH	VI
TC_{input}	Toggle of input	FSDB	VDW
TC_{output}	Toggle of output	FSDB	VDW
$TC_{internal}$	Toggle of internal connection	FSDB	VDW
$T_{arrival}$	Minimum arrival time	FSDB	VDW
$P_{internal}$	Internal power	RH	VDP
P_{switch}	Switching power	RH	VDP
$T_{transition}$	Transition time	RH	VDP
I_{peak}	Peak current	RH	VDP

*RH: Redhawk-SC; FSDB: format of waveform file

**VI: vector-independent features; VDW: vector-dependent waveform features; VDP: vector-dependent power features

C. Raw Features Extraction

We extract features related to dynamic IR-drop from a variety of sources. These features are the basic information of a cell, and we call them *raw features*. Raw features are listed in TABLE II. The second column shows the description of raw features. x and y are the physical locations of the target cell. w and h are the dimensions of the target cell. Effective resistance is the equivalent resistance that contributes to real power. R_{eff} is the effective resistance from the power pad to the target cell. *SPR* is the shortest path resistance from the power pad to the target cell. SPR is a quicker method to estimate the resistance than R_{eff}. *Cell type* is a number that encodes the type of cell in the library. We use label encoding, so the feature dimension does not increase with the number of cell types. P_{leak} is the leakage power dissipated by the target cell. Leakage power is the static power consumption regardless of logic switching. C_{load} is the loading capacitor on the output of the target cell. Pi_{c1}, Pi_{c2}, Pi_r are the equivalent pi model of the target cell. Redhawk-SC translates a cell into an equivalent pi model. Then, Redhawk-SC can calculate Elmore delay and perform IR-drop analysis. TC_{input} is the total toggle counts of inputs of the target cell. TC_{output} is the total toggle counts of outputs of the target cell. $TC_{internal}$ is the total toggle counts of the internal connection of the target cell. $T_{arrival}$ is the minimum arrival time in the capture cycle. $P_{internal}$ is the internal power dissipated by the target cell. P_{switch} is the switching power dissipated by the target cell. $T_{transition}$ is the transition time when the cell toggles. I_{peak} is the peak current during simulation time.

The source from which we extract raw features affects the runtime and performance of the IR-drop predictor. The third column of TABLE II shows the source of raw features. After performing power analysis, we can extract x, y, w, h, *SPR*, R_{eff}, *Cell type*, P_{leak}, C_{load}, Pi_{c1}, Pi_{c2}, Pi_r, $P_{internal}$, P_{switch}, $T_{transition}$, and I_{peak} from Redhawk-SC. After we perform the logic simulation, we get the waveform file, FSDB. Then, we can extract TC_{input}, TC_{output}, $TC_{internal}$, and $T_{arrival}$ from FSDB.

Based on the runtime, we separate raw features into three categories, including vector-independent features (VI), vector-dependent waveform features (VDW), and vector-dependent power features (VDP). The categories of features are listed in the fourth column of TABLE II. x, y, w, h, *SPR*, R_{eff}, *Cell type*, C_{load}, Pi_{c1}, Pi_{c2}, and Pi_r do not change with input vectors, so they are VI. P_{leak} does not depend on input transition, so it is almost constant for a logic cell. We classify P_{leak} as VI. The same circuit design shares the same VI. We extract VI only once in the training phase, so they do not affect our prediction time. $P_{internal}$, P_{switch}, $T_{transition}$, and I_{peak} are VDP because these features are extracted from power analysis. TC_{input}, TC_{output}, $TC_{internal}$, and $T_{arrival}$ are VDW because these features are extracted from the waveform file. Totally, VDP and VDW extract time increases with the number of vectors. Both VDP and VDW affect our prediction time. VDW extraction is faster than VDP extraction because power analysis takes a long time.

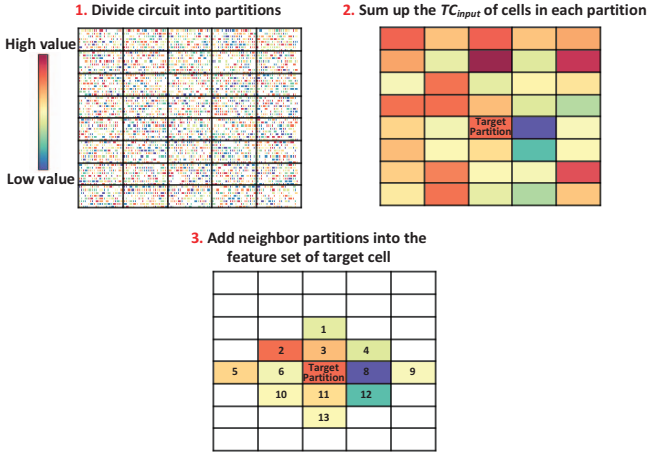

Fig. 3. Density map features

D. Density Map Features Creation

Besides raw features, we also propose to use *density map features*, so our machine learning model has local information around a target cell [3]. Because IR-drop is a local effect, density map features provide important information for the IR-drop prediction of the target cell. We show how to create density map features for a target cell in Fig. 3. First, we divide the whole circuit into partitions. Because IR-drop effects are limited in range, the width and height of the partition are fixed for the same technology node. According to our experiment in Fig. 4, the prediction results are optimal when the width of a partition is set to eight times the largest cell width, and the height of the partition is eight times standard cell height. Fig. 3-1 shows the cell placement of a circuit and its partitions. Take total toggle counts of inputs, TC_{input}, as an example. Each dot is a cell, and the color of

each dot represents the TC_{input} of a cell. Second, we sum up TC_{input} values of all cells in each partition and create a *density map* (Fig. 3-2). A *target partition* is a partition that contains the target cell. A neighbor partition is a partition of North/East/West/South to the target partition. Third, we append TC_{input} values of 13 neighbor partitions to the feature set of the target cell (Fig. 3-3).

In this paper, we select *SPR*, R_{eff}, TC_{input}, TC_{output}, $TC_{internal}$, $P_{internal}$, P_{switch}, and I_{peak} to create density map features. Totally, our feature dimension is 124 (20 raw features plus 8 x 13 density map features). Please note that the feature dimension only depends on the number of raw features and density map features. Our feature dimension is not changed with the size of the circuit design, so our proposed technique is scalable to large designs.

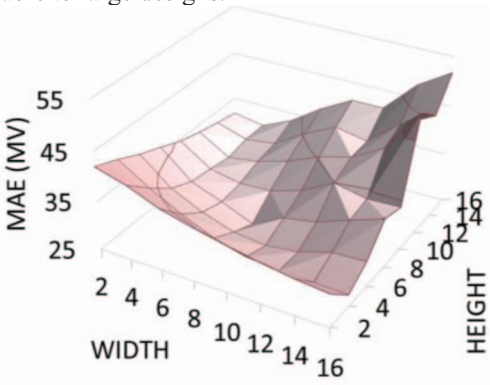

Fig. 4. MAE for different partition sizes

E. Machine learning

In our work, we choose *XGBoost* as our machine learning model [7]. XGBoost is a tree-based machine learning model which has achieved good results in many machine learning challenges. Here are the reasons why we choose XGBoost. First, XGBoost is extremely fast. The training time of the tree-based machine learning model is shorter than the training time of neural networks. What's more, XGBoost supports parallel computations, which makes its training time shorter than the training time of the other tree-based machine learning model. Second, XGBoost is explainable. We can extract the feature importance from the tree-based machine learning model. Then, we can understand how the machine learning model makes a prediction.

We illustrate how XGBoost works below. We are given a data set $\mathcal{D} = \{(x_i, y_i)\}\ (|\mathcal{D}| = \mathrm{n}, x_i \in \mathbb{R}^{\mathrm{m}}, y_i \in \mathbb{R})$, where x_i is input features of the i_{th} cell. m is the feature dimension and N is the total number of cells. y_i is the golden IR-drop value of i_{th} cell calculated by dynamic IR-drop analyzer. XGBoost ensembles K additive functions to get the predicted IR-drop value $\hat{y}_i$.

$$\hat{y}_i = \phi(x_i) = \sum_k^K f_k(x_i), f_k \in \mathcal{F}, \tag{1}$$

where F is the space of additive functions. f_k is a tree structure. Our goal is to minimize the following regularized objective function.

$$\mathcal{L}(\phi) = \sum_i \ell(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{2}$$

$$, \text{where } \Omega(\mathrm{f}) = \gamma \mathrm{T} + \frac{1}{2}\lambda\|w\|^2 \tag{3}$$

ℓ is a differentiable loss function that measures the error between predicted IR-drop value $\hat{y}_i$ and golden IR-drop value y_i. In this work, we use *mean square error* as our loss function,

$$\frac{1}{N}\sum_{i=1}^{N}\|\hat{y}_i - y_i\|^2. \tag{4}$$

Ω is the regularization term. T is the number of leaves in the tree. w is the leaf weights. XGBoost uses these parameters to punish the complexity of the model.

XGBoost is trained in an additive manner. XGBoost adds a new tree f_t to minimize the following objective.

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} \ell(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \tag{5}$$

$\hat{y}_i^{(t-1)}$ is the predicted IR-drop value of the i_{th} cell at the $t\text{-}1_{th}$ iteration. XGBoost uses *second-order Taylor expansion* to approximate the objective function in Eq. (5), so XGBoost is capable of dealing with large datasets.

IV. Experimental Results

A. Environmental setup & evaluation metric

The experiments have been executed on Intel Xeon E5-2620 v3 and Intel Core i7-9700KF with GeForce RTX 2080. Three designs (*MEMC*, *b19*, and *leon3mp*) have been evaluated by our proposed method. MEMC is a design to perform motion estimation and motion compensation, which is a design created by ourselves. b19 and leon3mp are benchmark circuits from *ITC'99* and *Gaisler Research*, respectively. Profiles of these three circuits are shown in TABLE III. They are synthesized in NanGate 45-nm open-cell library [8], and the nominal V_{DD} of these designs is 0.95V. The second column shows the number of cells in each design. The third columns show the number of vectors generated by the ATPG tool, *TetraMAX*. The fourth and fifth columns show the average and max IR-drop of all cells for the first 100 vectors. For these designs, the golden IR-drop values are provided by *Redhawk-SC*. In our experiments, golden IR-drop values are the summation of IR-drop and ground bounce. The sixth column shows the runtime of IR-drop analysis for the capture cycle of a vector using Redhawk-SC.

TABLE III. Profile of circuit designs

Circuit Design	# cells	# vectors	Mean IR-drop (mV)	Max IR-drop (mV)	Runtime of IR-drop analysis (s)
MEMC	223,829	187	248.67	402.72	7,224
b19	347,049	1,953	183.87	565.34	4,785
leon3mp	1,049,484	3,558	219.03	467.68	24,210

To evaluate the performance of our method, we use *Normalized Root Mean Square Error* (RMSE), which is defined in equations (6) and (7). In these equations, y_i is the golden IR-drop of the i_{th} cell, and $\hat{y}_i$ is the predicted IR-drop of the i_{th} cell. N is the number of training data.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}} \tag{6}$$

$$\text{NRMSE} = \frac{RMSE}{mean(y)} * 100\% \quad (7)$$

The *Pearson product-moment correlation coefficient* (CC) is defined in equation (8). CC is a number between -1 and 1. If predicted IR-drop and golden IR-drop are the same, CC is 1.

$$\text{CC} = \frac{\sum_{i=1}^{N}[y_i - mean(y)][\hat{y}_i - mean(\hat{y})]}{\sqrt{\sum_{i=1}^{N}[y_i - mean(y)]^2[\hat{y}_i - mean(\hat{y})]^2}} \quad (8)$$

Mean Absolute Error (*MAE*) and *Max Error* (*MaxE*) are defined in equations (9) and (10). We use MaxE to show the maximum error (golden IR – predict IR). Positive MaxE means *underprediction,* which is more serious than *overprediction* because we may fail to find serious IR-drop cells.

$$\text{MAE} = \frac{\sum_{i=1}^{N}\|\hat{y}_i - y_i\|}{N} \quad (9)$$

$$\text{MaxE} = \max(y_i - \hat{y}_i), i = 1 \; to \; N \quad (10)$$

B. Vector-based Dynamic IR-drop Predictor

The training data and validation data are obtained from different vectors to validate whether our model successfully predicts the IR-drop of unseen vectors. According to our experiment in Fig. 5, as the number of training vectors increases, our MAE decreases until the number is greater than ten. Because one IR-drop vector can produce millions of labeled data, just a few vectors are enough to train our model. We use data from 18 vectors obtained in vector selection for training and the data from other randomly chosen 82 vectors for validation.

We compare the performance of raw features with density map features for three circuit designs. The results are shown in TABLE IV. The column, *Raw*, shows the performance of using only raw features. The column, *Raw+Density*, shows the performance of using both raw features and density map features. We can see that the CC of Raw+Density is higher than Raw. The NRMSE of Raw+Density is lower than Raw. Although MaxE is high, they occur mostly in low IR-drop cells, so they won't affect our risky vector identification (see section D). This experiment shows that density map features are very important features that help to improve our prediction accuracy.

TABLE. IV Performance of raw features and density map features

	MEMC		b19		leon3mp	
	Raw*	Raw +Density**	Raw*	Raw +Density**	Raw*	Raw +Density**
MAE(mV)	26.37	19.47	43.06	25.91	13.65	12.29
MaxE(mV)	344.93	326.49	456.26	195.69	360.52	298.20
CC	0.45	0.66	0.34	0.79	0.73	0.83
NRMSE(%)	18.36	15.25	29.35	18.97	12.55	10.32

*Raw: raw features only

**Raw+Density: *raw features and density map features*

In the second experiment, we compare the performance of different feature sets. The results are shown in TABLE V. For each circuit design, CC of VI is the lowest among all feature sets because VI has features of the circuit but doesn't have features of vectors. Then, we add the features about vector, VDW, and WDP. For b19, VI+VDW has better performance than VI+VDP. For MEMC and leon3mp, VI+VDP has better performance than VI+VDW. Both performances of VI+VDP and VI+VDW are better than VI for all circuit designs. Surprisingly, experimental results reveal that VI+VDW+VDP cannot achieve the best performance among all feature sets. One possible reason could be the *curse of dimensionality* [9]. VDW and VDP provide similar information for the machine learning model. Using both VDW and VDP increases the feature dimension but does not provide more information than using one of VDW and VDP. Based on the result in TABLE V, training machine learning with VI+VDW or VI+VDP can achieve good performance.

In Fig. 6, we use the XGBoost model with the feature set VI+VDW to predict all the cells of one unseen vector. The left part is the golden IR-drop map, and the right part is our predicted IR-dorp map. These two IR-drop maps are nearly the same by our eyes. The MAE of this vector is 10.57mV. The MaXE of this vector is 81.06mV.

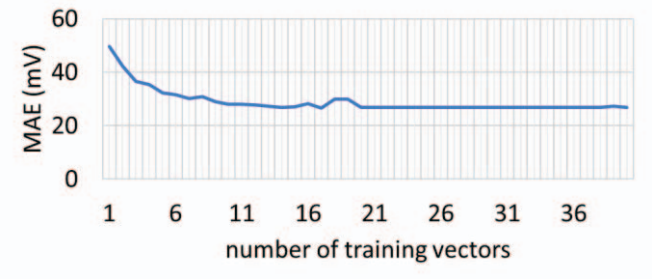

Fig. 5. Compare models trained with different numbers of vectors

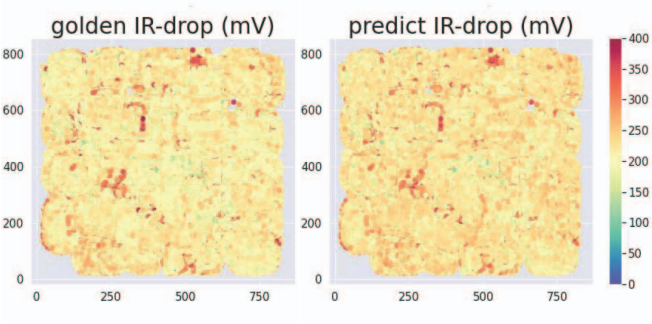

Fig. 6. Golden IR-drop map v.s predicted IR-drop map

for an unseen vector of b19

C. Runtime Comparison

TABLE VI shows the training time of our work. For VI+VDW of MEMC, we spend 130,032 seconds collecting training labels. Then, we extract VI and VDW with 1,608 and 109 seconds, respectively. Preprocessing needs 154 seconds, and training needs 126 seconds. The total training time is 132,029 seconds. Collection training labels takes a large portion of training time.

TABLE VII shows the prediction time of our work. The values are normalized per 100K cells for one vector. For MEMC, b19, and leon3map, their VI+VDP speedup ratios are 6.1, 17.7, and 4.7, respectively. By comparison, the VI+VDW speedup ratios are 495.1, 778, 600.6, respectively. Because VDP needs more run time to perform power analysis, prediction with VDP is much slower. In summary, our proposed VI+VDW features are very fast with good prediction accuracy. By adding the proposed waveform features, VDW, we successfully improve the speedup ratio.

D. IR-drop Risky Vector Identification

Our machine learning prediction can be used to identify *IR-drop risky vectors* from the entire test vector set, which is infeasible using traditional IR-drop analysis. IR-drop risky vectors are those vectors whose average IR-drop of 5% worst

cells is large than 310 mV. *IR-safe vectors* are those vectors that are not IR-risky. To make sure we catch all IR-drop risky vectors, we did two extra modifications to our predicted IR-drop: 1) perform a linear transformation, and 2) add one MAE as *guardband* (MEMC = 9.61mV, b19 = 23mV, leon3mp = 10.07mV). We show confusion matrices in Fig.7. For MEMC, there are 10 IR-drop risky vectors, and we identify all of them. We did not under-predict any IR-drop risky vectors. The upper right corner is zero. For b19 and leon3mp, there are five and two underpredictions, respectively. We can just redo test generation to replace these vectors in a relatively short time.

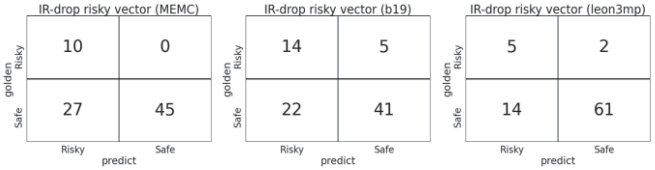

Fig. 7. IR-drop risky vector identification

V. CONCLUSION

We proposed a vector-based machine learning model to predict the dynamic IR-drop for all logic cells in the circuit. Our machine learning model considers important features extracted from the logic simulation waveform so we can make an IR-drop prediction quickly for all vectors. We also propose density map features to include the information of cells near the target cell. Our experiments show that our mean absolute error is less than 3% of the nominal supply voltage. We achieve more than 495 speedups compared to a popular commercial tool. We show one application of our machine learning model in the last experiment. Our machine learning model can identify most IR-drop risky vectors. For our benchmark circuits (1 million gates), IR-drop for all vectors ($3{,}500^{+}$ vectors), which takes at least one year for traditional analysis, can be predicted with our machine learning model in one day.

TABLE V. Performance of IR-drop predictor for different feature sets

Design / Metrics	MEMC				b19				leon3mp			
	VI	VI+VDW	VI+VDP	VI+VDW +VDP	VI	VI+VDW	VI+VDP	VI+VDW +VDP	VI	VI+VDW	VI+VDP	VI+VDW +VDP
MAE(mV)	22.84	19.47	15.77	15.76	34.69	25.91	30.61	30.19	9.56	12.29	14.39	13.06
MaxE(mV)	338.41	326.49	262.41	357.04	279.86	195.69	249.48	287.52	626.55	298.2	219.81	263.09
CC	0.47	0.66	0.81	0.78	0.55	0.79	0.71	0.71	0.76	0.83	0.87	0.82
NRMSE(%)	17.72	15.25	12.17	12.91	25.61	18.97	21.24	21.34	11.8	10.32	9.71	10.31
Feature Dimension	20	72	72	124	20	72	72	124	20	72	72	124

TABLE VI. Training time for different feature sets (18*N* training data)

Stage	MEMC				b19				leon3mp			
	VI	VI+VDW	VI+VDP	VI+VDW +VDP	VI	VI+VDW	VI+VDP	VI+VDW +VDP	VI	VI+VDW	VI+VDP	VI+VDW +VDP
IR-drop analysis for training labels(s)	130,032	130,032	130,032	130,032	86,130	86,130	86,130	86,130	435,780	435,780	435,780	435,780
VI extraction(s)	1,608	1,608	1,608	1,608	2,951	2,951	2,951	2,951	5,913	5,913	5,913	5,913
VDW extraction(s)	0	109	0	109	0	24	0	24	0	256	0	256
Preprocess(s)	25	154	129	125	10	86	79	83	57	470	470	470
Training(s)	58	126	146	151	85	96	150	171	746	509	925	943
Total trining runtime(s)	131,723	132,029	131,915	132,025	89,176	89,287	89,310	89,359	442,496	442,928	443,088	443,362

TABLE VII. Prediction time for different feature sets (100K cells)

Stage	MEMC				b19				leon3mp			
	VI	VI+VDW	VI+VDP	VI+VDW +VDP	VI	VI+VDW	VI+VDP	VI+VDW +VDP	VI	VI+VDW	VI+VDP	VI+VDW +VDP
VDW extraction(s)	0.0	2.7	0.0	2.7	0.0	0.4	0.0	0.4	0.0	1.4	0.0	1.4
VDP extraction(s)	0.0	0.0	529.4	529.4	0.0	0.0	76.4	76.4	0.0	0.0	489.5	489.5
Preprocess+Prediction(s)	0.6	3.8	3.2	3.1	0.2	1.4	1.3	1.3	0.3	2.5	2.5	2.5
Total prediction time(s)	0.6	6.5	532.6	535.2	0.2	1.8	77.6	78.1	0.3	3.8	492.0	493.3
IR-drop analysis(s)	3,227.5				1,378.8				2,306.8			
Speed-up ratio	5,234.8	495.1	6.1	6.0	8,861.1	778.0	17.8	17.7	7,685.7	600.6	4.7	4.7

REFERENCES

[1] Yamato, Yuta, et al. "A fast and accurate per-cell dynamic IR-drop estimation method for at-speed scan test pattern validation." 2012 IEEE International Test Conference. IEEE, 2012.

[2] Dhotre, Harshad, Stephan Eggersglüß, and Rolf Drechsler. "Identification of efficient clustering techniques for test power activity on the layout." 2017 IEEE 26th Asian Test Symposium (ATS). IEEE, 2017.

[3] Fang, Yen-Chun, et al. "Machine-learning-based dynamic IR drop prediction for ECO." Proceedings of the International Conference on Computer-Aided Design. 2018.

[4] Mozaffari, Seyed Nima, et al. "An Efficient Supervised Learning Method to Predict Power Supply Noise During At-speed Test." 2019 IEEE International Test Conference (ITC). IEEE, 2019.

[5] Xie, Zhiyao, et al. "PowerNet: Transferable dynamic IR drop estimation via maximum convolutional neural network." 2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC). IEEE, 2020.

[6] Ansys RedHawk-SC User Manual, 2018

[7] Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016.

[8] "NanGate FreePDK45 open cell library," Jan. 2016, [Online]. Available: http://www.nangate.com/?page_id=2325

[9] Köppen, Mario. "The curse of dimensionality." 5th Online World Conference on Soft Computing in Industrial Applications (WSC5). Vol. 1. 2000.