Fast IR-Drop Prediction of Analog Circuits Using Recurrent Synchronized GCN and Y-Net Model

Seunggyu Lee*, Daijoon Hyun†§, Younggwang Jung*, Gangmin Cho*, Youngsoo Shin*
*School of Electrical Engineering, KAIST, Daejeon 34141, Korea
†Department of Semiconductor Systems Engineering, Sejong University, Seoul 05006, Korea

***Abstract*—IR-drop analysis of analog circuits is a challenge because the current waveforms of target transistors, with connection to VDD or VSS, are extracted through transistor-level simulation, and the analysis itself, in particular dynamic one, is computationally expensive. We introduce two ML models for high-speed analysis. (1) Recurrent synchronized graph convolutional network (RS-GCN) is used for quick prediction of current waveforms. Each subcircuit is modeled with recurrent-GCN, in which recurrent connection is for the analysis in discrete time series. Recurrent-GCNs are synchronized to take account of common connections including VDD, VSS, and the inputs and outputs of subcircuits. Experiments show that RS-GCN takes only 0.85% of SPICE runtime, while prediction error is 14% on average. (2) Y-Net is applied for actual IR-drop analysis of small layout partition, one by one. Pad location and PDN resistance are provided as one 2D input of Y-Net; they are encoded and go through GCNs to account for neighbor layout partitions. Current map, derived from RS-GCN, becomes the second input. Final IR-drop map is extracted from the decoder. Experiments demonstrate that Y-Net, in conjunction with RS-GCN for current extraction, takes 2.5% of runtime from popular commercial solution with 15% prediction inaccuracy.**

***Index Terms*—** IR-drop analysis, analog circuit, RS-GCN, Y-Net.

I. Introduction

Voltage drop caused by power distribution network (PDN) distorts signals and thus degrades the performance of analog circuits; for instance, IR-drop of 5% reduces the bandwidth of phase-locked loop circuit by 25% [1]. It becomes more important because IR-drop increases with technology scaling down due to the higher resistance of copper interconnect and the higher current density [2].

IR-drop analysis for analog circuits is very time-consuming. A transistor-level simulation of the entire circuit is preceded to characterize the current waveforms of transistors connected to power or ground, called target transistors. PDN is then modeled as a resistive network with current sources representing the current waveforms (or their average values) obtained by the simulation. The voltage drops at the target transistors are calculated using modified nodal analysis [3], where considering the actual switching of function cells may take a lot of time. This entire process takes 16 hours for a circuit with 57k transistors.

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS2022-00207425, Data-Driven Design Technology for AI Semiconductor). The EDA tool was supported by the IC Design Education Center (IDEC), Korea.

§Corresponding author

There have been prior works related to fast IR-drop analysis for analog circuits. First, fast transistor-level simulation can be used to characterize the current waveforms. It employs various acceleration methods, such as table-based transistor modeling [4], parasitic reduction [5], matrix partitioning [6], and hierarchical simulation [7]. But, despite an accuracy loss of more than 15%, the speed-up is only about 7 times. Next, several machine learning (ML) models have been introduced to predict IR-drop in digital circuits. In [8], XG-boost predicts the voltage drop at a node by roughly calculating and considering the effective resistance between the node, voltage sources, and current sources in a local. U-Net receives the image maps of flowing current, effective resistance to voltage sources, and pad location for the local area and outputs an IR-drop map for that area [9]. These methods are not suitable for analog circuits using irregular PDN, where IR-drop can be affected by distant components.

In this paper, we address fast IR-drop prediction of analog circuits using two ML models. First, recurrent synchronized graph convolutional network (RS-GCN) is introduced to predict the current waveforms at target transistors. GCN is employed to consider the effects of connected components in a subcircuit, and it is recurrently configured to predict the current values over simulation time, called recurrent-GCN. The common connections including VDD, VSS, and the inputs and outputs of subcircuits are considered by synchronizing recurrent-GCNs. Second, Y-Net model is applied to each layout partition for IR-drop analysis, where the current waveforms obtained through RS-GCN are input in the form of a two-dimensional map. The other input maps for pad location and PDN resistance go through an encoder and multiple GCNs to take into account the effective resistances from target transistors to pads; this is much faster than calculating them one by one. To consider the impact between layout partitions, graph convolutions are performed on the features of adjacent partitions, and IR-drop map is finally obtained from a decoder.

Our main contributions are summarized as follows.

- RS-GCN for current waveform prediction, which considers the continuity of current over time through recurrent-GCN and the current flowing between subcircuits using synchronize connection.
- Y-Net for IR-drop analysis, which takes account of the impact of distant areas on IR-drop and extracts the features of effective resistance through graph convolutions.

979-8-3503-4859-0/DATE24/© 2024 EDAA

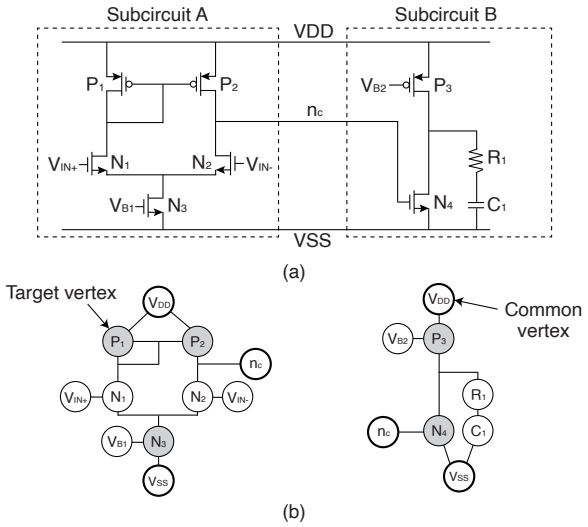

Fig. 1. (a) An example circuit with two subcircuits, and (b) graph model of subcircuits.

The remainder of this paper is organized as follows. Section II introduces RS-GCN, and its application to test circuits is demonstrated. In Section III, we present Y-Net model, and the experimental results are provided. Conclusions are drawn in Section IV.

II. Current Waveform Prediction Using RS-GCN

In cell-based digital circuits, the current waveform of each cell or even a block may be modeled beforehand or be extracted from a library. This does not hold in analog circuits. The current waveform of each transistor should be obtained through transistor-level simulation before actual IR-drop analysis, which is impractical. Thus, a fast prediction of current waveform for target transistors is performed through RS-GCN.

A. Graph Modeling and Input Matrices

Fig. 1 shows an example of graph modeling for a circuit with two subcircuits, in which vertices correspond to circuit components such as transistors, resistors, capacitors, VDD, VSS, and bias voltages. In addition, a connection vertex is introduced to indicate how and where subgraphs should be connected (see n_c). The transistors connected to VDD or VSS are called target transistors and modeled by target vertices (gray circles), while other transistors are marked as non-target (white circles). The connection between the components is represented by an edge. Note that there are common vertices (bold circles) between the two subgraphs, such as VDD, VSS, and n_c in Fig. 1(b).

Each vertex is associated with a feature vector of 10 elements, representing length (l_n) and width (w_n) of nMOS, those of pMOS, power voltage, ground voltage, bias voltage, resistance, capacitance, and a binary number indicating whether it is a connection vertex. In the feature vector, only the elements related to the vertex have non-zero values, and the rest are set to 0; a vertex of nMOS, for example, has a feature vector with all elements set to 0 except l_n and w_n. For a subcircuit with N components, N feature vectors form

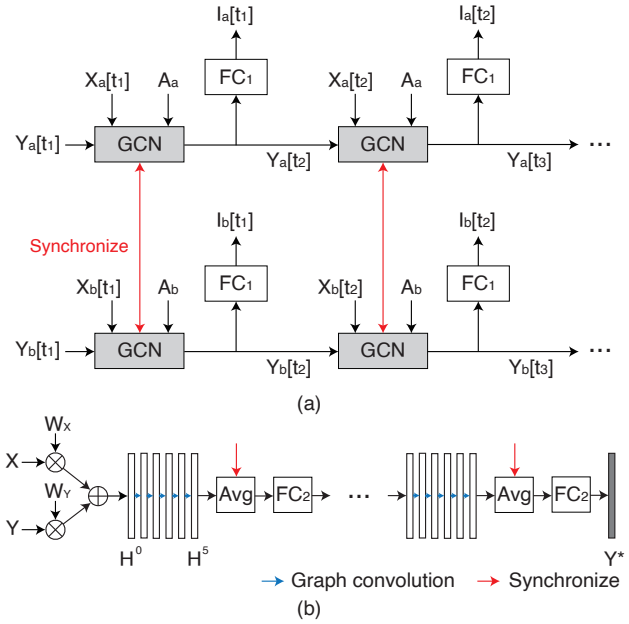

Fig. 2. (a) RS-GCN architecture and (b) the structure of GCN block.

a feature matrix $\mathbf{X} \in \mathbb{R}^{N\times 10}$, and the structure of the graph is represented by adjacency matrix $\mathbf{A}$, where rows and columns indicate the vertices in the graph, and the entries (i, j) and (j, i) are set to 1 if there is an edge between vertices i and j, 0 otherwise.

B. RS-GCN Model

Fig. 2(a) illustrates the architecture of RS-GCN for an example circuit with two subcircuits. A recurrent-GCN, which consists of multiple layers based on the time steps, is constructed for each subcircuit. In a layer, $\mathbf{X}$, $\mathbf{A}$, and recurrent input $\mathbf{Y}$ are fed into GCN block, where $\mathbf{Y}$ in the first layer is set to zero matrix. The GCN output is passed as a recurrent input to the next layer, and at the same time, 10 feature values of each target vertex go into a fully-connected (FC) layer FC_1 to get the current value at a time step. The current value at the next time step can be obtained from the output of FC_1 in the next layer. In this way, the current waveform is finally obtained from the current values at the entire time steps.

Fig. 2(b) shows the structure of GCN block. The input feature $\mathbf{H}^0$ of the first graph convolutional layer (GCL) is generated by

$$\mathbf{H}^0 = \mathbf{X}\,\mathbf{W_x} + \mathbf{Y}\,\mathbf{W_y}, \tag{1}$$

where $\mathbf{W_x}$ and $\mathbf{W_y}$ are the weight matrices for $\mathbf{X}$ and $\mathbf{Y}$, respectively. By graph convolution, the hidden state of $l+1$ layer $\mathbf{H}^{l+1}$ is determined by

$$\mathbf{H}^{l+1} = \sigma(\hat{\mathbf{D}}^{-\frac{1}{2}}\,\hat{\mathbf{A}}\,\hat{\mathbf{D}}^{-\frac{1}{2}}\,\mathbf{H}^l\,\mathbf{W}_h^l), \tag{2}$$

where $\hat{\mathbf{A}}$ is $\mathbf{A}$ plus identity matrix $\mathbf{I}$. $\hat{\mathbf{A}}$ is normalized by $\hat{\mathbf{D}}^{-\frac{1}{2}}\,\hat{\mathbf{A}}\,\hat{\mathbf{D}}^{-\frac{1}{2}}$ so that the entries of each row sum to one, where $\hat{\mathbf{D}}$ corresponds to a diagonal vertex degree matrix of $\hat{\mathbf{A}}$. $\sigma(\cdot)$ is a LeakyReLU activation, and $\mathbf{W}_h^l$ is a hidden weight matrix of l layer. After 5 graph convolutions, the common vertex features are averaged for each element with the features passed through

TABLE I
CURRENT WAVEFORM ERROR AND RUNTIME OF RS-GCN AND FAST-SPICE, WITH SPICE AS A REFERENCE

Circuits	#TRs (k)	SPICE	Fast-SPICE		RS-GCN	
		Time	Error	Time	Error	Time
sqrt	1.2	2601s	10%	587s	13%	18s
dac	2.6	3517s	12%	712s	14%	26s
voter	4.2	4475s	16%	852s	15%	35s
mem_pl	7.4	6502s	16%	941s	12%	66s
smult	11.5	8891s	19%	1042s	14%	83s
ram2k	13.8	11946s	22%	1452s	14%	101s
chip2	18.8	16299s	21%	2029s	15%	132s
Avg.			17%	×1/7	14%	×1/117

the synchronize connections and linearly transformed by FC_2. The output replaces the common vertex features in the matrix of size $N \times 10$, which enters into GCL again. The process is repeated 3 times.

C. Model Training

Total 20k sample designs are generated from 7 circuits, where 70% of samples are used for training and the rest for validation. To this end, for each circuit, operating voltage, bias voltages, the lengths and widths of transistors, the sizes of capacitors and resistors, and input voltage waveforms are each scaled by one of the pre-determined ratios. The reference current waveforms of target transistors are obtained by commercial tool [10].

We limit the maximum number of epochs to 500 and apply early stopping with the patience of 5 epochs [11], where Adam optimizer [12] is used with a learning rate of 0.001. The weight values of RS-GCN are optimized such that a loss function L is minimized. L is formulated by

$$L = \frac{1}{T}\frac{1}{N}\sum_{t=1}^{T}\sum_{i=1}^{N}(I_i[t] - \hat{I}_i[t])^2, \tag{3}$$

where N is the number of target transistors in the circuit and T is the number of time steps. $I_i[t]$ and $\hat{I}_i[t]$ denote the predicted current value and the actual current value of target transistor i at time step t, respectively. We assume that the derivative of the average function for the synchronization signal is zero. As a result, RS-GCN is treated as multiple separate recurrent-GCNs (imagine Fig. 2(a) without synchronize). In a recurrent-GCN, weight values are shared across all time sequences. Thus, the backpropagation through time algorithm [13] is adopted to propagate error signals backward through time. The gradients obtained by propagating to several recurrent-GCNs are then averaged to update the weight values.

D. Experimental Results

Experiments are carried out to assess RS-GCN model. A set of test circuits is compiled from MCNC benchmarks [14], and is listed in Table I in order of circuit complexity. Note that these circuits have not been used to train the model. The reference current waveforms at target transistors are extracted through transistor-level simulation (SPICE) with industrial 28-nm process design kit [10], and the associated runtime is given in the third column.

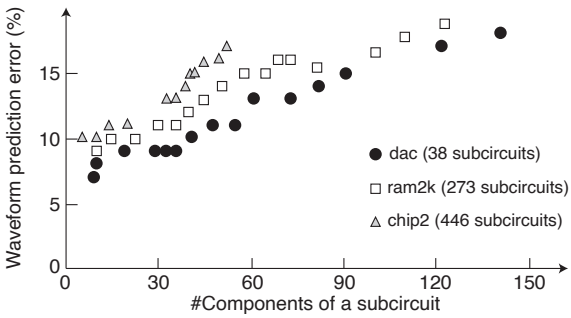

Fig. 3. Current waveform prediction error per subcircuit.

1) Prediction Accuracy: RS-GCN as well as fast transistor-level simulation (fast-SPICE) [15] are compared to SPICE in error and runtime. Waveform error is the percentage mismatch (in area-wise) with SPICE waveform as a reference; the error value in Table I is the average over all the waveforms. RS-GCN exhibits 14% error on average of all test circuits. Notably this is smaller than 17% error from fast-SPICE. In addition, fast-SPICE shows larger errors as the circuit size increases due to more approximations from node reduction and matrix partitioning [16]. The accuracy of RS-GCN is not directly affected by the circuit size, but is affected by the number of subcircuits and their complexities.

Fig. 3 shows the errors of waveform prediction according to the number of subcircuit components. In three circuits, dac, ram2k, and chip2, the error increases as the number of components increases. For large subcircuits, the impact of components on the current waveform may not be sufficiently considered by a fixed number of GCLs. Also, the error increases as the number of subcircuits increases. For subcircuits with 40 components, the average error of chip2 is 3% and 5% greater than those of ram2k and dac, respectively. The error increase comes from making a large number of feature vectors at VDD or VSS into one vector by synchronization. In RS-GCN, synchronization and recurrent structure are very important in terms of prediction accuracy. On average of all test circuits, removing synchronize connections increases the error by 7.5%; replacing recurrent-GCN with a single GCN increases the error by 4.4%.

2) Runtime: RS-GCN prediction is much faster than SPICE as well as fast-SPICE. Specifically, on average of all test circuits, its runtime is 1/117 of SPICE runtime while the runtime of fast-SPICE is only 1/7 as presented in Table I. Graph convolutions and the operations in FC layers take 56% and 33% of the total RS-GCN runtime, respectively. The runtime is proportional to circuit size because the number of operations is dependent on the number of vertices in subcircuits. For instance, the operations of graph convolution are mostly performed on the product of input feature $\mathbf{H}^l$ and weight matrix $\mathbf{W}_h^l$, in which the row size of $\mathbf{H}^l$ is the number of vertices in a subcircuit.

III. IR-DROP PREDICTION USING Y-NET

In analog circuits, the empty spaces in metal layers are filled with PDN straps as much as possible to prevent IR-drop vio-

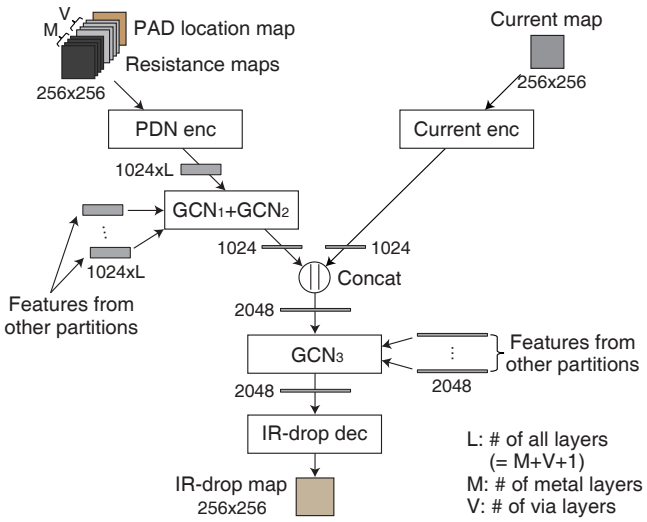

Fig. 4. Y-Net architecture for IR-drop prediction.

lations [17]. As a result, PDN is formed very irregularly, and IR-drop is affected by the features even at distant locations. However, the previous works [8], [9] only consider the input features in a local area, and the IR-drop predictions are not accurate for analog circuits. Thus, we propose Y-Net which predicts IR-drop considering the impact between distant areas.

A. Input Maps

Y-Net receives three types of input maps: pad location map, resistance map, and current map. They are extracted from each partition of corresponding layer, where a map consists of 256 pixels $\times$ 256 pixels and a pixel corresponds to a grid of 0.9 um $\times$ 0.9 um in the layout.

- **Pad location map**: the locations of power or ground pads are specified in the map in such a way that the pixels corresponding to the centers of pads contain 1 and the other pixels are assigned 0.
- **Resistance map**: the resistance value of the physical component (metal or via) constructing PDN is assigned to each pixel in proportion to the grid area overlapped by the component. This map is generated for metal and via layers.
- **Current map**: the amount of current flowing through a target transistor is assigned to each pixel in proportion to the grid area overlapping the active region of the transistor.

B. Y-Net Architecture

Fig. 4 shows the architecture of Y-Net, which consists of two encoders, three GCNs, and a decoder.

1) Encoder: A pad location map and resistance maps individually go through PDN encoder while a current map is given to current encoder. PDN encoder and current encoder have the same structure as illustrated in Fig. 5(a), but their weights are trained differently. The encoders consist of 6 convolutional layers with 3×3 kernels and 2×2 stride. Each convolution doubles the number of channels and reduces the width and height of the map by half, which is followed by

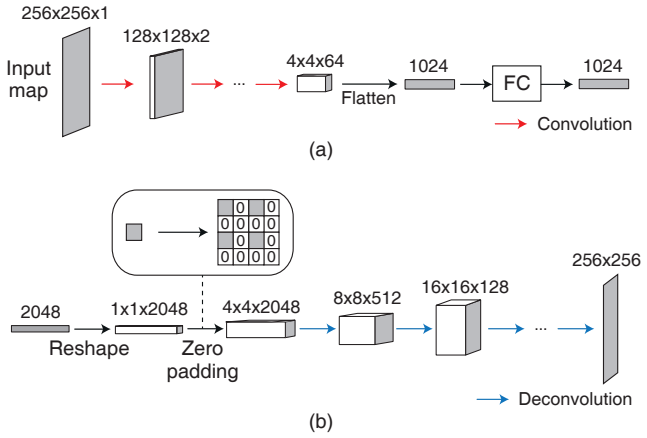

Fig. 5. Structures of (a) encoder and (b) decoder.

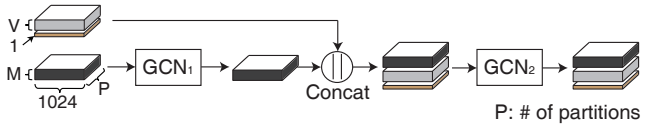

Fig. 6. Single stage of serial GCN.

batch normalization and LeakyReLU activation. The output of convolutional layers is flattened and goes through an FC layer to generate a feature vector with 1024 values.

2) Serial GCN (GCN_1+GCN_2): Fig. 6 illustrates a single stage of serial GCN which consists of GCN_1 and GCN_2 in series; this is repeated 3 times in serial GCN. The outputs of PDN encoder for all partitions form the stacked matrices of size $1024\times L\times P$, where L is the number of all PDN layers and P is the number of partitions. Only the features for metal layers are provided to GCN_1, and its outputs are concatenated with the features of pad and via layers to be fed into GCN_2. Metal layer features among the outputs of GCN_2 go into GCN_1 of the second stage, and the same process is repeated.

Both GCNs consist of 5 GCLs, and graph convolution is performed as in Eq. (2) for each layer with weight matrices of size 1024×1024. The graph modelings of GCN_1 and GCN_2 are illustrated in Fig. 7(b) and (c), respectively. For GCN_1, each partition of metal layers is modeled as a vertex, and the vertices of partitions adjacent in the routing direction are connected by an undirected edge. In the adjacency matrix, two entries (i, j) and (j, i) are set to 1 if the vertices i and j are connected, 0 otherwise. The input of GCN_1 is reshaped to the feature matrix of size $MP\times$1024, where M is the number of metal layers. For GCN_2, the partitions of all PDN layers are modeled by vertices and a directed edge connects a vertex to another one that is placed at the same location of the lower layer. In the adjacency matrix, an entry (i, j) is set to 1 if an edge connects vertices i and j. The feature matrix of size $LP\times$1024 consists of the features of all PDN layers.

3) GCN_3: From the output of serial GCN, only M1 layer features for the partition are selected, and they are concatenated with current features to generate a vector with 2048 values. It is fed into GCN_3 together with concatenated feature

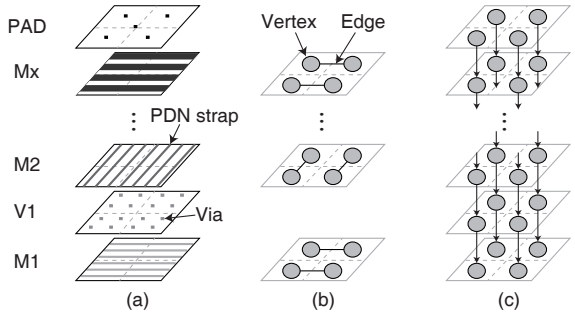

Fig. 7. Graph modeling for GCN_1 and GCN_2 in Y-Net: (a) the example of given PDN layers and graph modelings for (b) GCN_1 and (c) GCN_2.

vectors from other partitions, which constructs a feature matrix of size $P\times2048$. In graph modeling, each partition is modeled as a vertex, and the vertices of partitions that are adjacent in the layout are connected by an edge. The adjacency matrix of size $P\times P$ represents the connection of vertices in the graph; if vertices i and j are connected, two entries (i, j) and (j, i) are set to 1. GCN_3 consists of 5 GCLs, where each graph convolution is performed the same as Eq. (2) with the weight matrices of size 2048×2048. The graph convolutions are repeatedly performed to consider the impact between distant partitions on IR-drop.

4) Decoder: The output of GCN_3 is divided into P feature vectors of size 2048, and each vector is individually fed into IR-drop decoder. Fig. 5(b) shows the structure of decoder. The vector is reshaped to $1\times1\times2048$, and zeros are added to generate $4\times4\times2048$ maps, where a pixel value is copied to four pixels in 4×4 map as illustrated in the figure. It then goes through 6 deconvolutional layers with 4×4 kernels and 2×2 stride, and batch normalization and LeakyReLU activation are performed after every deconvolution. This deconvolution increases the map size twice for every layer and gradually decreases the number of channels according to the number of kernels. The decoder outputs an IR-drop map for each partition, and those maps are combined to generate one IR-drop map of the entire circuit layout.

C. Model Training

The sample designs, used for RS-GCN in Section II-C, are physically implemented for training Y-Net, and the reference IR-drop values at the power pins of each design are obtained from a commercial tool [18]. We use the mean squared error of IR-drop prediction as a loss function, which is represented by

$$Loss = \frac{1}{P}\frac{1}{N}\sum_{j=1}^{P}\sum_{i=1}^{N}(V_i^j[t] - \hat{V}_i^j[t])^2, \tag{4}$$

where N is the number of pixels in output map, $V_i^j[t]$ is the predicted IR-drop of pixel i on partition j at time step t, and $\hat{V}_i^j[t]$ is the actual IR-drop. The weights of Y-Net are updated every 16 samples to minimize the loss function. To this end, the backpropagation algorithm is used to calculate the gradient of the loss function with respect to the weights, where the derivatives of the features from other partitions are set to zeros.

TABLE II
COMPARISON OF VARIOUS IR-DROP ANALYSIS METHODS USING ML MODELS, SUCH AS XG-BOOST [8], U-NET [9], AND Y-NET

Circuits	XG-boost [8]		U-Net [9]		Y-Net	
	MAPE	Time	MAPE	Time	MAPE	Time
sqrt	30%	12s	20%	121s	11%	35s
dac	30%	21s	20%	267s	12%	55s
voter	29%	32s	21%	446s	12%	77s
mem_pl	31%	55s	20%	711s	11%	145s
smult	31%	78s	20%	965s	11%	184s
ram2k	32%	101s	20%	1372s	12%	268s
chip2	32%	132s	20%	2350s	12%	471s
Avg.	31%	×1/32	20%	×1/2	11%	×1/11

Y-Net is trained with Adam optimizer at a learning rate of 0.001. The maximum number of epochs is set to 500, and early stopping is used with the patience of 5 epochs.

D. Experimental Results

Experiments are carried out on the analog circuits employed in Section II-D. Regular PDN synthesis is performed independently with signal routing, and they are merged in such a way that the straps near metal routings are removed to avoid design rule violations [17]. For these designs, IR-drop analysis is performed through Y-Net over 10 ns period with 1 ps time step, which is the same condition as the current waveform simulation. Reference IR-drop values are obtained from a commercial solution (Totem) [18].

1) Assessment of Y-Net: In Table II, the accuracy and runtime of Y-Net are compared with those of the previous works, XG-boost [8] and U-Net [9]. The three models use the same reference current waveforms as input and their predicted IR-drops are compared with the reference IR-drop values. XG-boost uses a simple approximation of the effective resistance for feature extraction, which greatly reduces runtime, but at the cost of 31% loss of accuracy as listed in columns 2-3. On the other hand, U-Net shows better accuracy with the MAPE of 20%, which comes from the utilization of U-Net model and the accurate calculation of effective resistance. In this work, the resistive network of PDN is simplified to a circuit with equivalent resistances between M1 power pins and V1 power vias and those between the power vias and power pads. The effective resistance is then calculated in the equivalent circuit. Even though the total amount of computation is reduced, the runtime is not significantly reduced because test circuits contain a lot of V1 power vias.

The proposed model, Y-Net, achieves the MAPE of 11%, 9% smaller than U-Net, as shown in column 6. This is because, unlike other methods, Y-Net takes into account the effect on IR-drop between distant areas through GCN_3. If GCN_3 is removed from Y-Net, the MAPE increases to 19%. The effectiveness of serial GCN is also identified by replacing it with other models; the average error goes up by 9% if we use an FC instead while replacing serial GCN with a single GCN increases the error by 5%. Y-Net performs accurate predictions even when PDN is highly irregular. We create irregular PDNs for a circuit (b18) by randomly removing PDN straps, and the three ML models are used to predict IR-drop for those PDNs.

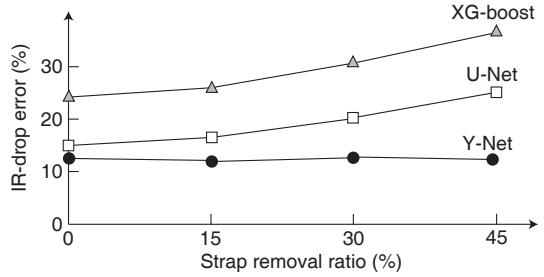

Fig. 8. The IR-drop errors of XG-boost, U-Net, and Y-Net according to the randomly removed strap ratio, i.e. the irregularity of PDN.

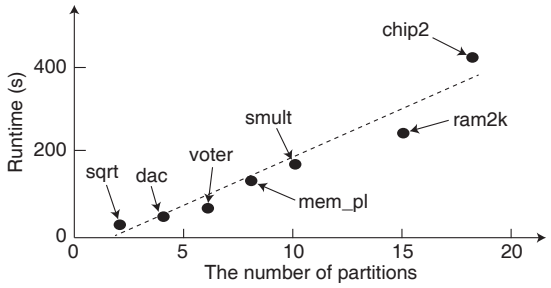

Fig. 9. Correlation between the runtime of Y-Net and the number of partitions.

As a result, as shown in Fig. 8, the prediction errors of XG-boost and U-Net increase as PDN becomes more irregular, while Y-Net always maintains similar accuracy.

The runtime of Y-Net is reduced by a factor of 11 (see column 7), which is achieved by embedding the effective resistance feature through serial GCN rather than directly calculating it. The runtime of Y-Net is largely affected by the number of partitions. The number of rows of input feature matrix for GCN is proportional to the number of partitions, and for each partition, encoding and decoding operations are carried out. Accordingly, the linear relationship between the runtime of Y-Net and the number of partitions is shown in Fig. 9.

2) Y-Net with RS-GCN: Table III shows the MAPE and the runtime of Y-Net with RS-GCN (RS-GCN+Y-Net) in columns 5-6, and they are compared with the results of a commercial solution, Totem with fast-SPICE (Fast-SPICE+Totem), listed in columns 3-4. The MAPEs of the two methods are calculated by the comparison with reference IR-drop values (SPICE+Totem). On average for all test circuits, RS-GCN+Y-Net shows an error of 15%, which is almost the same with Fast-SPICE+Totem; in case of maximum error, RS-GCN+Y-Net is better at 17% than Fast-SPICE+Totem at 19%. In terms of runtime, RS-GCN+Y-Net is 40 times faster than the reference method, and exhibits 14 times speed-up compared to Fast-SPICE+Totem. On average, in the prediction of RS-GCN+Y-Net, feature map extraction and IR-drop prediction of Y-Net take 21% and 49% of the total runtime, respectively. The remaining runtime is spent on the graph modeling of subcircuits and current waveform prediction through RS-GCN.

IV. Conclusion

We have addressed fast IR-drop analysis for analog circuits. RS-GCN is introduced to extract the current waveforms at the transistors connected to power or ground. PDN is modeled as a resistive network with the current sources derived from RS-GCN, and IR-drop analysis is then performed using Y-Net. Experimental results show that RS-GCN is 17× faster than fast-SPICE with 3% better accuracy. Y-Net, in conjunction with RS-GCN for current extraction, takes 2.5% of runtime from popular commercial solution with 15% prediction inaccuracy.

TABLE III
Comparison of IR-drop analysis between commercial tools (Fast-SPICE+Totem) and the proposed models (RS-GCN+Y-Net)

Circuits	SPICE +Totem	Fast-SPICE +Totem		RS-GCN +Y-Net	
	Time	MAPE	Time	MAPE	Time
sqrt	49m	10%	15m	15%	1m
dac	68m	12%	22m	15%	1m
voter	93m	14%	33m	16%	2m
mem_pl	138m	15%	46m	13%	4m
smult	188m	17%	57m	15%	4m
ram2k	255m	19%	80m	15%	6m
chip2	346m	18%	108m	16%	10m
Avg.		15%	×1/3	15%	×1/40

References

[1] M.-H. Chou and S.-I. Liu, "A type-I PLL with foreground loop bandwidth calibration," *IEEE Trans. on Circuits and Systems II*, vol. 68, no. 4, pp. 1103–1107, Apr. 2021.

[2] A. Ajami, K. Banerjee, A. Mehrotra, and M. Pedram, "Analysis of IR-drop scaling with implications for deep submicron P/G network designs," in *Proc. Int. Symp. on Quality Electronic Design*, Mar. 2003, pp. 35–40.

[3] C.-W. Ho, A. Ruehli, and P. Brennan, "The modified nodal approach to network analysis," *IEEE Trans. on Circuits and Syst.*, vol. 22, pp. 504–509, Jun. 1975.

[4] A. Rofougaran and A. A. Abidi, "A table lookup FET model for accurate analog circuit simulation," *IEEE Trans. on Comput.-Aided Design Integr. Circuits Syst.*, vol. 12, no. 2, pp. 324–335, Feb. 1993.

[5] J. Rommes and W. H. A. Schilders, "Efficient methods for large resistor networks," *IEEE Trans. on Comput.-Aided Design Integr. Circuits Syst.*, vol. 29, no. 1, pp. 28–39, Jan. 2010.

[6] K. J. Kerns, M. Bhattacharya, S. Rudnaya, and K. Gullapalli, "Automatic, hierarchy-independent partitioning method for transistor-level circuit simulation," U.S. Patent 0 030 665, Jan. 29, 2009.

[7] K. J. Kerns and Z. Peng, "SPICE optimized for arrays," U.S. Patent 7 324 363, Jan. 29, 2008.

[8] 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*, Mar. 2020, pp. 1307–1310.

[9] Y. Kwon, G. Jung, D. Hyun, and Y. Shin, "Dynamic IR drop prediction using image-to-image translation neural network," in *Proc. Int. Symp. on Circuits and Systems*, May 2021, pp. 1–5.

[10] "HSPICE User Guide," *Synopsys*, Sep. 2008.

[11] G. Montavon, G. Orr, and K.-R. Müller, *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2012, pp. 55–69.

[12] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," *arXiv:1412.6980 [cs.LG]*, Dec. 2014.

[13] T. Mikolov *et al.*, "Extensions of recurrent neural network language model," in *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, May 2011, pp. 5528–5531.

[14] MCNC benchmarks, Accessed: Dec. 22, 2022. [Online]. Available: http://www.intusoft.com/benchmarks.htm

[15] "FineSim User Guide," *Synopsys*, Jul. 2012.

[16] P. Li, L. M. Silveira, and P. Feldmann, *Simulation and Verification of Electronic and Biological Systems*. Berlin, Germany: Springer, 2011, pp. 23–42.

[17] G. Balsdon, "Filling vacant areas of an integrated circuit design," U.S. Patent 8 458 636, Jun. 4, 2013.

[18] "Totem User Guide," *Ansys*, Apr. 2018.