Simultaneous Conjugate Gradient and iAFF-UNet for Accurate IR Drop Calculation

He Liu[1], Yipei Xu[2], Simin Tao[3], Zhipeng Huang[3], Biwei Xie[4,3], Xingquan Li[3,5*] and Wei Gao[1]
[1]School of Electronic and Computer Engineering, Peking University, Shenzhen, China
[2]Electronic and Information Engineering, Tongji University, Shanghai, China
[3]Pengcheng Laboratory, Shenzhen, China
[4]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[5]School of Mathematics and Statistics, Minnan Normal University, Zhangzhou, China
Email: liuh@stu.pku.edu.cn, xuyipei@tongji.edu.cn, {taosm, huangzhp}@pcl.ac.cn, xiebiwei@ict.ac.cn, fzulxq@gmail.com, gaowei262@pku.edu.cn

***Abstract*—IR drop analysis has become a computationally challenging problem with the shrinking of advanced process nodes. Solving the IR drop problem is time-consuming and an accurate and fast IR drop calculator is crucial for shortening the design cycle. In this work, we introduce an innovative IR drop calculation framework based on the conjugate gradient method and iAFFUNet network. iAFFUNet incorporates the UNet structure with the iterative attention feature fusion (iAFF) blocks to refine conventional approaches of feature concatenation and fusion. iAFF blocks employ multi-scale channel attention modules to enhance feature representation. Furthermore, we leverage intermediate results from the conjugate gradient method as augmented features and utilize graph attention networks for initial value calculation, thereby expediting the iteration process. Alternatively, the matrix operation process can be further accelerated using GPU optimization. During the training phase, we adopt a transfer learning strategy by fine-tuning limited real circuit datasets based on a pre-trained model obtained from training with a substantial amount of synthetic circuit datasets. Experimental results on the ICCAD 2023 contest real hidden testcases under the Nangate 45nm process node show that our model achieves an average improvement of 48.7% and 53.9% in MAE compared to the contest's champion and the second place, respectively. Additionally, our model achieves a 39.8% reduction in CPU runtime compared to the champion of the contest.**

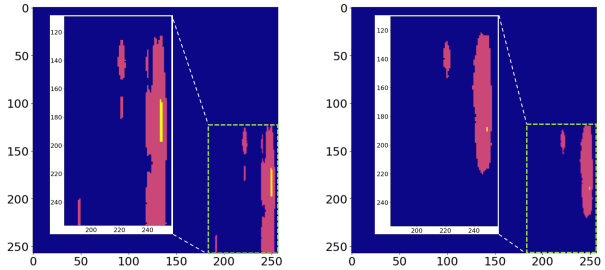
***Index Terms*—IR drop calculation, iAFF, UNet, graph attention network, conjugate gradient**

I. Introduction

In high-performance and low-power integrated circuit (IC) designs, IR drop analysis is becoming increasingly vital in the backend physical design of the chips, directly impacting the chip's performance and reliability [1]. IR drop can cause local voltage drops within the semiconductor chip, thereby affecting the switching speed of transistors. This may reduce the chip's computational speed, consequently impacting the overall performance. IR drop can also lead to timing violations, resulting in chip overheating, which exacerbates the aging process or potentially damages the IC design [2]. In advanced industrial designs, executing a complete full-chip IR drop analysis may require several hours [3]. To expedite IR drop analysis, a fast and accurate IR drop solver is required.

The power delivery network (PDN) can be represented as a configuration comprising voltage sources, current sources, and resistors, with the network's conductive paths acting as a series of resistors. Traditionally, determining the voltage at each node in PDN involves solving a set of linear equations represented as $GV = J$, where G stands for the conductance matrix, V is the vector of unknown voltages, and J is the current vector. In static IR drop simulations involving millions of PDN nodes, solving the equations is highly time-consuming. To address the limitations of traditional circuit simulation in terms of time and memory complexity, several previous works have made some effort. The approach in [4] simplifies the complex power grid into a coarser structure and then maps the solution back to the original grid, significantly enhancing the efficiency and speed of both DC and transient analysis for modern VLSI circuits. The authors of [5] employ a linear complexity random walk technique to analyze the power grids of integrated circuits. The work in [6] presents several efficient node-based and row-based power grid DC analysis methods, improving the overall analysis speed and efficiency.

With the advancement of machine learning, many research efforts have introduced ML-based IR drop estimators for predicting IR drop. The study [7] introduces a novel method for predicting IR drop in power grids using XGBoost. Which optimizes the feature extraction process and utilizes the locality of the power grid, the model maintains high computational speed



(a) Ground Truth (b) Machine Learning

Fig. 1. The comparison between the predicted hotspot map generated by a purely machine learning-based model and the corresponding ground truth for testcase14. The yellow and red areas represent the top 10% and the 10%-50% IR drop regions, respectively. (a) The ground truth hotspot. (b) The machine learning-based model.

when handling large-scale power grid designs. The method in [8] utilizes superposition and partitioning for effective electrical feature extraction and combines it with XGBoost for accurate IR drop calculation across various design modifications without rerunning comprehensive simulations. PowerNet [9] is a novel dynamic IR drop estimation technique based on the convolution neural network (CNN) model, enabling better transferability across different designs. The authors in [10] construct the IR drop prediction work as an image-to-image translation task, using the IREDGe network with the image-based features to predict the full-chip static IR drop.

The heuristic algorithms and numerical optimization methods previously discussed consume substantial time for large-scale PDN networks. While machine learning-based IR drop calculators have effectively reduced estimation times, achieving high accuracy remains challenging. For the purely ML-based IR drop calculation method, Fig. 1 demonstrates its low accuracy in IR drop hotspot detection. To address the time-consuming nature of the numerical methods and the low accuracy of the ML-based methods, we introduce a novel IR drop calculation framework, which integrates neural networks with numerical computation. The introduced neural network draws from the UNet model and integrates the iAFF [11] modules (iAFFUNet) to improve feature representation and fusion. We adopt a numerical optimization approach to generate intermediate IR drop maps as enhanced features. The following are the main contributions of this work.

- We introduce an effective simultaneous conjugate gradient and iAFFUNet (**CG-iAFFUNet**) IR drop calculation framework, which combines the advantage of the numerical optimization method and the deep learning model.
- We present an iAFF-UNet model, which integrates the iterative attention fusion blocks into the conventional UNet model. This architecture enables flexible handling of multi-scale features, effectively capturing both large-scale structures and small-scale details.
- We utilize the inexact conjugate gradient (CG) iteratively to pinpoint the turning point of IR drop calculation. Enhancing the efficiency of iterative CG, a graph attention network (GAT) is constructed to estimate a desirable initial solution for CG. Moreover, matrix operations in the iterative process are accelerated by utilizing the GPU.
- We use the Kolmogorov-Smirnov (KS) test to analyze the different distributions between synthetic data and limited real data. Transfer learning is applied to fine-tune models trained on synthetic data with limited real data, aiming to enhance generalizability on unseen real circuit data.
- Compared to the champion of the ICCAD 2023 contest, experimental results show that our proposed model achieves a 48.7% improvement in terms of MAE and a 39.8% reduction in runtime, striking a favorable balance between precision and runtime metrics.

II. Our IR Drop Calculation Framework

The modified nodal analysis method and Kirchhoff's current law (KCL) are circuit analysis techniques that are frequently used in calculating IR drop. These KCL equations form a system of linear equations that can be represented in matrix form as $GV = J$, where V is the vector of unknown node voltages. For each node, the diagonal elements G_{ii} indicate the total conductance connected to node i, and the off-diagonal elements G_{ij} $(i \neq j)$ represent the conductance between nodes i and j. The conductance matrix G can be represented as:

$$G = \begin{bmatrix} \sum_{j\neq 1} G_{1j} & -G_{12} & \cdots & -G_{1n} \\ -G_{21} & \sum_{j\neq 2} G_{2j} & \cdots & -G_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -G_{n1} & -G_{n2} & \cdots & \sum_{j\neq n} G_{nj} \end{bmatrix}$$

The current vector J is derived from the external currents flowing into the nodes of the circuit. For a circuit with n nodes, the current vector can be expressed as: $J^T = [J_1, J_2, \ldots, J_n]$. Here, J^T denotes the transpose of J, with each element J_i representing the net external current flowing into node i.

Calculating large matrices for IR drop analysis poses substantial challenges in computational runtime and memory consumption. It is widely acknowledged that numerical calculation methods are time-consuming, and machine learning models often exhibit low accuracy and poor generalization. To address the limitations of a single approach, we propose a hybrid method that leverages both numerical calculations and machine learning. For many numerical iterative methods, a practical approach is to leverage the initial several iterations to swiftly attain a reasonably accurate solution (referred to as a turning point). Upon surpassing this point where convergence slows, a precise machine-learning model can seamlessly take over to efficiently complete the iteration process. As shown in Fig. 2, to shorten the iterative method's runtime, we divide the entire iteration into two stages. After identifying a turning point through statistical analysis, where runtime and accuracy balance, the initial phase of iteration utilizes the CG algorithm, followed by the adoption of machine learning techniques for regression and computation. By combining an inexact iteration algorithm with machine learning models, we strike a balance between time efficiency and precision. In addition, to further reduce the iteration step of CG, a desirable initial solution should be properly selected.

In this work, we propose an accurate CG-iAFFUNet IR drop calculation framework, illustrated in Fig. 3, which primarily comprises two stages. In Stage I, we utilize a GAT model to generate a desirable initial solution for CG iteration. Then, an inexact CG algorithm is employed to iteratively pinpoint a turning point, and further utilize an an interpolation algorithm to generate a smooth and inexact IR drop map as an enhanced feature. In Stage II, we concatenate the enhanced feature of the inexact IR drop map with the initial features and feed them into the proposed iAFFUNet model, which enables swift and accurate computation of IR drop results for testcases.

III. **Stage I**: Numerical Optimization for IR Drop

A. *initGAT: Initial Solution Learning with GAT*

In addressing systems of linear equations, the choice of initial values for iterative methods profoundly influences the

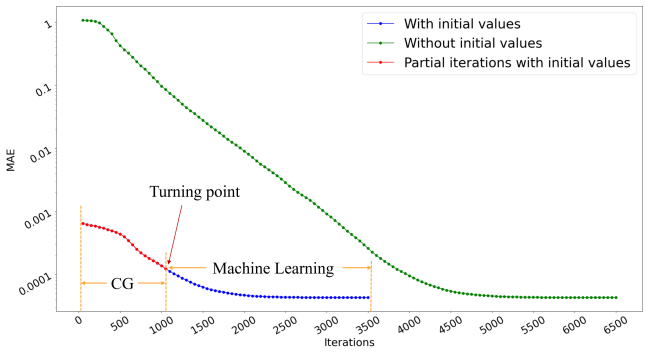


Fig. 2. Comparison of the performance of the CG algorithm with and without initial values. Green curve: CG algorithm with the initial values set to zero. Red and blue curve: CG algorithm with initial values estimated by proposed initGAT method.

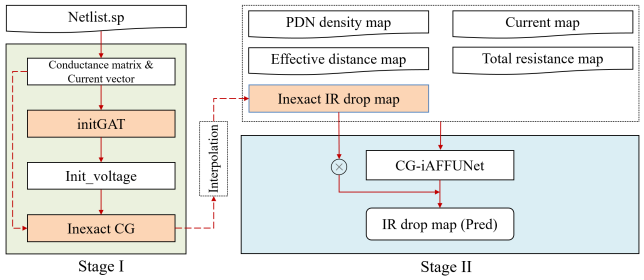


Fig. 3. Overall framework of the proposed CG-iAFFUNet for IR drop calculation. **Stage I**: The process of constructing enhanced inexact IR drop maps with CG algorithm. **Stage II**: The proposed model inference procedure.

convergence speed and the precision of the ultimate solution. As shown in Fig. 2, when initial values closely approximate the actual solution, the iterative method converges to the precise result with increased rapidity and efficiency.

The architecture of the proposed method for estimating initial voltages for iterative solvers is depicted in Fig. 4. In the PDN systems, the conductance matrix G is a sparse matrix, which facilitates its transformation into a graph representation. In this graph, each node corresponds to a row of matrix G. If the matrix has dimensions of $n \times n$, then the corresponding graph is composed of n nodes. If an element G_{ij} in matrix G is non-zero (and $i \neq j$), then there exists an edge between nodes i and j in the graph, where the value of this edge represents the conductance between two nodes in the PDN network.

The items in the current vector J correspond to the attributes of the nodes in the graph. Features are extracted with the graph attention network (GAT) [12] and GraphSAGE [13] blocks. The output of the multilayer perceptron (MLP) comprises the estimated voltage values corresponding to each node.

B. Inexact CG for IR Drop Map Calculation

The CG algorithm is an efficient solver for linear equations that are symmetric and positive-definite, making it well-suited for power delivery network analyses. Achieving higher precision through complete iteration will demand more iterations and extended time. We control the solution precision by adjusting the tolerance parameter tol, iterating to a certain

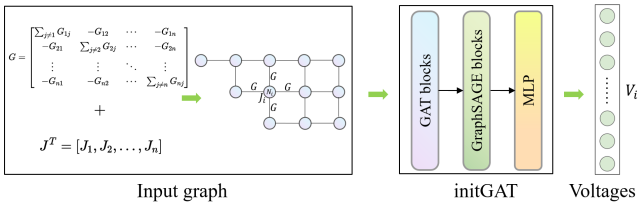


Fig. 4. The structure of constructing the input graph and estimating the initial solution using the proposed initGAT.

accuracy within a shorter period. List "inexact_CG" in Table I refers to the MAE of the inexact IR drop maps generated by the CG algorithm. Although these results may lack high precision, they serve as useful guides for IR drop calculation.

The detailed description of the proposed conjugate gradient method is shown in Algorithm 1. Lines 1-2 define the inputs and the outputs of the algorithm. Line 3 converts the input conductance matrix A into Compressed Sparse Row (CSR) format, which is efficient for storing and performing computations on sparse matrices. Line 4 converts vector b into a GPU array format to leverage GPU acceleration. In Lines 5-10: this part focuses on initializing the vector x_0. If x_0 is not provided, x is initialized as a zero vector of size N (size of matrix A). If x_0 is provided, to accelerate the iteration process, it is predicted using a model $x_0 = initGAT(PDNGraph)$, and then converted into a GPU array format. Lines 11-12 computes the initial residual $\mathbf{r}$ as $\mathbf{b}$ - $A \cdot \mathbf{x}$ and sets the initial search direction $\mathbf{p}$ to $\mathbf{r}$. Line 13 computes the dot product r_dot_r of the residual $\mathbf{r}$, executed on the GPU for hardware acceleration. Lines 14-26 perform up to maximum iterations. In each iteration, updates $\mathbf{x}$ and $\mathbf{r}$, calculates new values for α and β, and adjusts the search direction $\mathbf{p}$ with these values. If the new residual dot product is less than the tolerance tol, the convergence criterion is met, and the iteration loop is exited.

A preliminary estimate $x_0 = initGAT(PDNGraph)$ that is nearer to the actual solution can substantially decrease the number of iterations needed to achieve a specified tolerance level. With a reduced initial residual, each subsequent iteration can more effectively converge toward the ultimate solution. Particularly for large-scale issues, where computational resources and time are valuable, enhancing the initial estimate to minimize iterations can significantly boost efficiency. Secondly, the CG algorithm incorporates basic operations such as the dot product, which comprises multiple independent tasks ideally suited for parallel processing on numerous GPU cores. For sparse matrices, a GPU-optimized library such as CuPy is utilized to transfer matrix multiplication tasks to GPU devices, significantly enhancing the computational speed. Whether to transfer the above process to GPU for hardware acceleration is optional. In this paper, when comparing runtime with other works, the runtime used is consistently from the CPU version.

IV. **Stage II**: Machine Learning Based IR Drop

In this section, we introduce the selected features and the iAFFUNet neural network structure for IR drop calculation.

Algorithm 1 Conjugate Gradient algorithm with predicted initial values and GPU acceleration (optional)

Input: Conductance matrix A, current vector b, initial guess x_0, tolerance tol , maximum iterations max_iter
Output: Voltage vector x, number of iterations
$A \leftarrow CSR(A)$ ▷ Convert A to CSR format for optimization
$b \leftarrow Array(b)$ ▷ Convert b to a GPU array
if x_0 is not provided **then**
    $x \leftarrow ZeroVector(N)$ ▷ Initialize x as a GPU zero vector
else
    $x_0 \leftarrow initGAT(graph)$ ▷ Predict the initial value x_0
    $x \leftarrow Array(x_0)$ ▷ Initialize x as a GPU zero vector
end if
$r \leftarrow b - A \cdot x$
$p \leftarrow r$
$r\_dot\_r \leftarrow dot(r, r)$ ▷ Perform dot product on GPU
for $i = 1$ to max_iter **do**
    $Ap = A \cdot p$
    $\alpha = r\_dot\_r / dot(p, Ap)$ ▷ Perform dot product on GPU
    $x \leftarrow x + \alpha \cdot p$
    $r \leftarrow r - \alpha \cdot Ap$
    $r\_dot\_r\_new \leftarrow dot(r, r)$ ▷ Perform dot product on GPU
    $\beta = r\_dot\_r\_new / r\_dot\_r$
    $p \leftarrow r + \beta \cdot p$
    $r\_dot\_r \leftarrow r\_dot\_r\_new$
    if $r\_dot\_r < tol$ **then**
        break
    end if
end for

A. Features Selection and Generation

Feature selection is vital for the effectiveness of neural networks. For IR drop calculation tasks, the input features are constructed as images, which can be summarized as follows:

Current map. Instance current is determined by the ratio of the average power to voltage for each instance in the design. The current maps provide the distribution of power consumption across different parts of the chip, the distribution of current also directly impacts the voltage distribution within the PDN.

PDN density map. The density or compactness of wiring in the PDN across different regions of the chip. The PDN will be a region-specific uniform PDN where the density within a particular area remains consistent.

Effective distance map. Calculated based on the equivalent resistance of each node to the power pad. Nodes at greater distances may cause larger IR drops due to the need for current to travel through longer paths, increasing the total resistance.

Total resistance map. Calculated based on the topology and resistive values of the PDN. Total resistance maps are directly related to the magnitude of IR drop, as larger resistances lead to greater voltage drops under the same current.

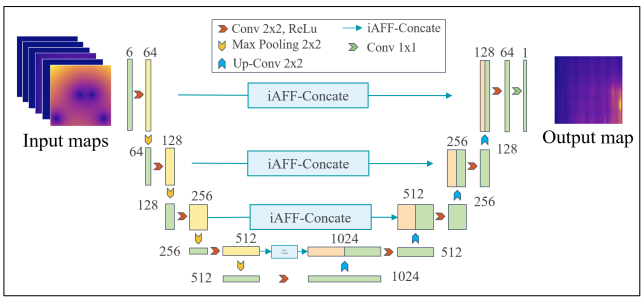


Fig. 5. The iAFFUNet structure proposed in this work.

Inexact IR drop map. This feature represents the IR drop maps calculated with the inexact CG algorithm proposed in Section III-B and the interpolation operation.

Feature crossing map. The product of the current map and the corresponding total resistance map.

Utilizing a comprehensive set of features helps the model learn deeper data representations of the PDN networks.

B. iAFFUNet for IR Drop Calculation

U-Net [14] is a symmetric convolutional neural network architecture specifically designed for image segmentation. Its key innovation is the encoder-decoder structure augmented with skip connections to ensure precise feature map fusion, thus maintaining a high accuracy. The encoder consists of multiple convolutional layers and max-pooling layers arranged alternately, aimed at reducing the spatial dimensions of the image while increasing the feature depth to capture the image's contextual information. The decoder part incrementally restores the feature maps' size via upsampling (or deconvolution) and convolutional layers. Concurrently, skip connections transfer corresponding feature maps from the encoder to conserve more detailed information. The final layer of the network, a convolutional layer, maps the output of the decoder to the same number of channels as the target output map, yielding the ultimate expected result.

The introduced iAFFUNet structure is a UNet variant. As shown in Fig. 5, we follow the classic UNet but are different from the traditional skip connections where the basic concatenate approach inadequately fuses features, missing essential information. We introduce the "iAFF-Concate" module depicted in Fig. 6 (a) to refine feature fusion techniques. We integrate the "iterative attention feature fusion (iAFF)" module and leverage the attention weights to adjust the contribution of two feature fusion mechanisms, which preserves the strong feature extraction of UNet and enhances the model's feature fusion ability. As shown in Fig. 6 (a), the two feature tensors A and B to be fused are concurrently fed into two branches. The first branch initially concatenates the two feature tensors, creating a new fused feature tensor. Subsequently, this tensor undergoes processing via a convolution layer, which facilitates the enhancement of feature extraction through convolu-

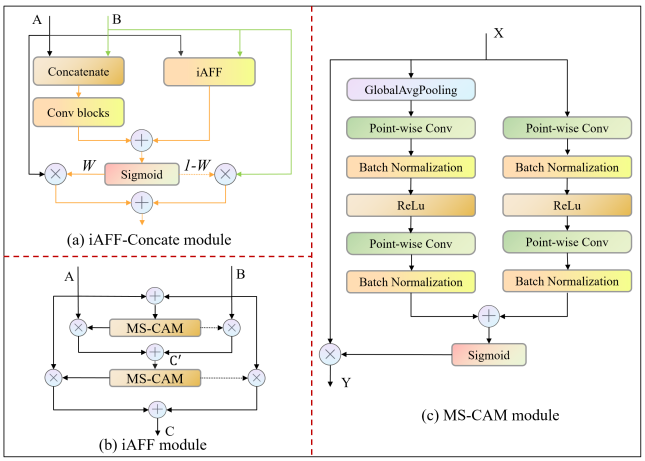


Fig. 6. The submodules of iAFFUNet structure proposed in the work. (a) iAFF-Concate module. (b) iAFF module. (c) MS-CAM module.

tional operations f_{conv}. This process can be represented as $F_1 = f_{conv}(concat(A, B))$. The other branch initially passes through the iAFF module, where features are fused via an attention mechanism, resulting in an output $F_2 = iAFF(A, B)$. This output is then added to the convolution feature map F_1 and passed through a Sigmoid activation σ to generate weights $W = \sigma(F_1 + F_2)$. Finally, the first feature tensor A is element-wise multiplied by the weights W, and this product is added to the element-wise product of the second input feature map B with the complement of the weights. The sum of these products serves as the output $F = W \cdot A + (1 - W) \cdot B$. This process involves an element-wise weighted addition operation within the network, combining direct features with attention-weighted features to produce the final output, which helps the model focus on the most beneficial features for the task. This enables more effective utilization of multi-scale feature information, thereby improving the model's accuracy.

As is depicted in Fig. 6 (b), the iAFF is constructed based on the multi-scale channel attention module (MS-CAM) [11] shown in Fig. 6 (c). MS-CAM analyzes the input feature maps at multi-scale spatial resolutions, contributing to the model capturing both global information and local details of the feature maps. The local attention mechanism focuses on capturing spatial details, for the input feature map $X \in \mathbb{R}^{C \times H \times W}$ with C channels and feature maps of size $H \times W$, local attention transforms the input feature map X through the operations:

$$f_L(X) = BN(Conv_{1\times1}(ReLU(BN(Conv_{1\times1}(X))))) \quad (1)$$

Where BN means the Batch Normalization and $ReLU$ denotes the Rectified Linear Unit. Global attention $f_G(X)$ reduces the spatial dimensions of the feature map to 1×1 by adaptive average pooling $AdaptiveAvgPool2d$ to capture global contextual information. This process involves a global pooling step initially, focusing on overall features. The outputs of local and global attention are combined and fusion weights are generated through the Sigmoid function

$$W = \sigma(f_L(X) + f_G(X)) \quad (2)$$

where σ is the Sigmoid function used to normalize fusion weights between 0 and 1. The computed channel weights are then used to reweight the input feature map to enhance or suppress specific channels $Y = W \cdot X$. In Fig. 6 (b), the iAFF module leverages two MS-CAM modules to fuse input features, and the output $\mathbb{C}'$ of the first MS-CAM module can be expressed as

$$\mathbb{C}' = W \cdot (A + B) \cdot A + (1 - W \cdot (A + B)) \cdot B \quad (3)$$

And the output $\mathbb{C}$ of the second MS-CAM are based on the $\mathbb{C}'$, it can be formulated as:

$$\mathbb{C} = W \cdot \mathbb{C}' \cdot A + (1 - W \cdot \mathbb{C}') \cdot B \quad (4)$$

In this work, we refine the traditional UNet with the "iAFF" modules, which leverage both local and global attention mechanisms for optimizing feature fusion. The local attention component of the MS-CAM module focuses on extracting detailed information, while the global attention part captures broader information from the input feature maps.

C. Transfer Learning: From Synthetic to Real Circuit

Applying deep learning in the EDA domain requires extensive and high-quality datasets for training. The acute scarcity of publicly available benchmarks is likely to precipitate overfitting and diminish the generalizability of models. In both academia and industry, to adequately protect intellectual property (IP), there is limited availability of diverse and realistic circuit data which is necessary to train machine learning models for EDA applications, although some works [15] have made great efforts to generate the datasets. In the task of IR drop prediction, the lack of rich PDN benchmarks still remains a key factor affecting prediction accuracy, especially in advanced process designs. In this work, to overcome the challenges of limited real circuits, we employ synthetic but realistic PDN benchmarks, where rich current maps are generated using BeGAN [16], a methodology based on generative adversarial networks (GANs). Additionally, PDN topologies and power pad locations are produced using the open-source software opeNPDN [17] [18].

We used 979 synthetic circuit datasets generated by BeGAN and 10 real circuit datasets as the training set, and another 10 unseen real circuit datasets as the test set. To verify the distribution of synthetic circuit data compared to real circuit data, we use the Kolmogorov-Smirnov (KS) test to calculate the KS statistic and P value between the means of the golden IR drop maps in two datasets. The Kolmogorov-Smirnov test (K-S test) is a frequently employed non-parametric statistical method. While it is predominantly employed as a one-sample test for comparing the frequency distribution of a sample with a recognized distribution, such as the Gaussian distribution, the Kolmogorov-Smirnov test can also be applied as a two-sample test. Initially, the null hypothesis assumes that the artificially generated data and the real data used for training have the same distribution. Using the KS test, we calculate a KS statistic of 0.7 and a P value of 0.012. In the second experiment, the null hypothesis assumes that the real data used for training and

the unseen real circuit data used for testing have the same distribution. The calculation yields a KS statistic of 0.2 and a P value of 0.994. Therefore, at a 5% level of significance in the first experiment, we can reject the null hypothesis, meaning that synthetic circuit datasets and real data for training have different probability distributions whereas in the second case, we can accept the null hypothesis that the limited real circuit data for training and the unseen real circuit for testing have the same distribution.

The limited availability of real circuit datasets for training may hinder the model's ability to effectively generalize to unseen data. Our model is initially trained on a large synthetic dataset to learn generalizable features that can be applicable to real circuits. Then, we use the pre-trained model as a starting point, accompanied by a reduction in the learning rate, fine-tuning is applied with a much smaller dataset of real circuits. This step is crucial for adjusting the model weights to better suit the distribution of the real circuit datasets while preserving the generalization of the pre-trained model.

V. Experiments Results

Our algorithm and model were implemented using Python and Pytorch on a Linux machine with two NVIDIA A100 GPUs, 4 Intel Xeon Platinum 8380 CPUs at 2.3 GHz, and 1T RAM. In this work, the ICCAD 2023 contest benchmark suites [3] and the open-source benchmark suites [16] under the Nangate 45nm technology node are used as the train and test datasets. In these datasets, there are only 10 real circuit datasets available for training and another 10 hidden real datasets for testing, along with a large amount of synthetically generated data with BeGAN. This transfer learning approach is used to prevent overfitting that may occur when training solely on real data and to enhance generalizability. According to the concept of transfer learning, we train a model with 979 synthetic circuit data and then fine-tune it on this pre-trained model using 10 real circuit data to obtain a model that adapts to the distribution of real data. To evaluate the efficiency of the proposed IR drop calculation framework, we compared the results with the state-of-the-art machine learning-based methods, corresponding to the champion and the second place of the ICCAD 2023 contest in terms of the mean absolute error (MAE), F1-score, and runtime. In this work, we use 90% of the maximum IR drop as a threshold, meaning that nodes within the top 10% IR drop are categorized as the positive class. Based on the process, the F1-score can be used as a metric for binary classification that evaluates the accuracy of identifying hotspot regions, the metric facilitates a balanced evaluation of the model's accuracy in terms of both precision and recall, where a higher score indicates better precision.

A. Comparison on Accuracy

Table I shows the experimental results. "MAE" denotes the mean absolute error between the predicted and the reference (golden) IR drop maps. A lower MAE indicates a closer approximation of the predicted IR drop map to the ground truth IR drop map. "F1-score" measures the accuracy in identifying hotspot regions, where a higher score indicates better precision. "Size" refers to the dimensions of the IR drop maps. In "ratio_1", the two columns represent the ratios of our model's MAE and F1-score values to those of the first place's corresponding MAE and F1-score. "ratio_2" denotes the ratios of the aforementioned metrics to those of the second place. In the evaluation against the hidden real testcases, the proposed CG-iAFFUNet framework shows a 48.7% and 53.9% improvement in MAE over the results of the first place and the second place of the ICCAD 2023 contest, respectively. For the F1-score, the framework achieves a 62.1% improvement compared with both the results of the first and the second place. Fig. 7 and Fig. 8 provide a clear and straightforward visual representation of the MAE and F1 score comparisons of unseen testcases, respectively. It can be observed that the MAE results for unseen testcases, particularly testcase13, 14, 15, and 16, are slightly higher, this is due to the different conditions under which each testcase was generated, with these particular cases possibly being produced under extreme conditions. And our model still achieves good results in these testcases. For detailed demonstration, the comparison between the predicted and actual (golden) IR drop map images for the hidden real "testcase20" are presented in Fig. 9. As Fig. 10 illustrated our model also achieves higher accuracy in identifying hotspot areas for the "testcase14". The visual evidence demonstrates that the results calculated by the proposed method exhibit a strong correlation with the ground truth IR drop maps.

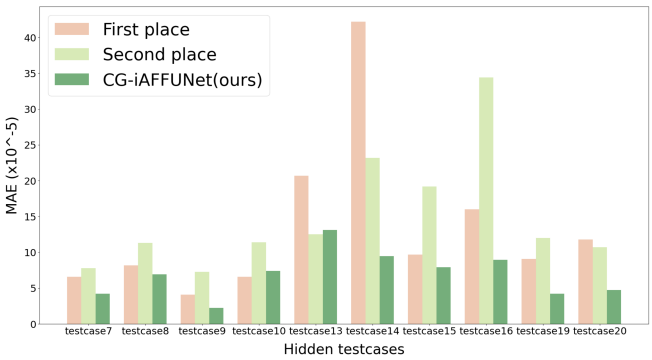


Fig. 7. Comparison of Mean Absolute Error (MAE) on ten hidden testcases.

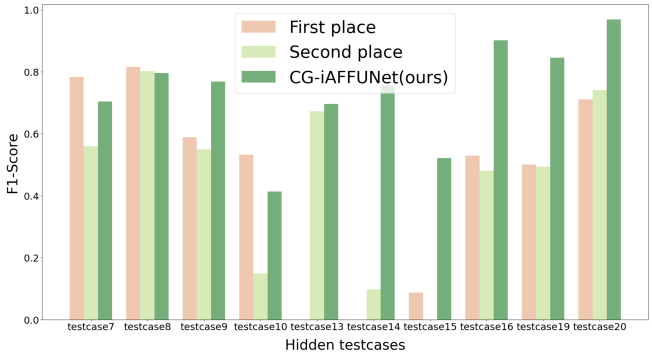


Fig. 8. Comparison of F1-score on ten hidden testcases.

TABLE I
EXPERIMENTAL RESULTS ON THE ICCAD 2023 BENCHMARKS. ↓ MEANS "LOWER IS BETTER", ↑ MEANS "HIGHER IS BETTER".

Test cases	Size	First Place			Second Place			Our CG-iAFFUNet						inexact_CG runtime (s)			
		MAE	F1 score	RT(s)	MAE	F1 score	RT(s)	inexact_CG	MAE ↓	F1 score ↑	RT(s)	Ratio_1	Ratio_2	W/o initGAT	W/ initGAT	W/ initGAT &GPU	Speed-up
testcase 7	601	6.56E-5	**0.783**	12.48	7.76E-5	0.560	2.75	1.20E-4	**4.19E-5**	0.704	7.22	0.64/0.90	0.54/1.26	3.43	0.86	0.60	3.99/5.72
testcase 8	601	8.15E-5	**0.816**	12.03	1.13E-4	0.802	2.57	1.28E-4	**6.93E-5**	0.796	7.72	0.85/0.98	0.61/0.99	3.70	0.78	0.57	4.74/6.49
testcase 9	835	4.06E-5	0.589	20.04	7.27E-5	0.550	4.52	1.16E-4	**2.25E-5**	**0.768**	11.95	0.55/1.30	0.31/1.40	7.47	1.95	0.64	3.87/11.67
testcase 10	835	**6.59E-5**	**0.532**	19.51	1.14E-4	0.149	4.23	1.56E-4	7.41E-5	0.414	11.41	1.12/0.78	0.65/2.78	7.38	1.88	0.68	3.93/10.85
testcase 13	257	2.07E-4	0	5.91	**1.25E-4**	0.673	0.77	1.53E-4	1.31E-4	**0.696**	3.21	0.63/NAN	1.04/1.03	0.62	0.3	0.66	2.07/0.94
testcase 14	257	4.22E-4	0	5.68	2.32E-4	0.098	0.79	1.96E-4	**9.45E-5**	**0.757**	3.36	0.22/NAN	0.41/7.72	0.69	0.22	0.76	3.14/0.91
testcase 15	489	9.68E-5	0.088	12.20	1.92E-4	0	2.02	1.10E-4	**7.89E-5**	**0.521**	6.24	0.81/5.92	0.41/NAN	2.17	0.82	0.60	2.65/3.62
testcase 16	489	1.60E-4	0.529	11.01	3.44E-4	0.481	1.99	1.94E-4	**8.96E-5**	**0.902**	5.76	0.56/1.71	0.26/1.88	2.20	0.8	0.63	2.75/3.49
testcase 19	870	9.05E-5	0.501	20.52	1.2E-4	0.494	4.9	8.25E-5	**4.23E-5**	**0.845**	13.75	0.47/1.69	0.35/1.71	7.92	2.67	0.73	2.97/10.85
testcase 20	870	1.18E-4	0.711	19.32	1.07E-4	0.741	4.72	9.01E-5	**4.73E-5**	**0.969**	12.90	0.40/1.36	0.44/1.31	8.23	2.26	0.70	3.64/11.76
Average	/	1.35E-4	0.455	13.87	1.50E-4	0.455	2.93	1.35E-4	**6.91E-5**	**0.737**	8.35	**0.513/1.621**	**0.461/1.621**	4.37	1.25	0.66	3.48/6.62

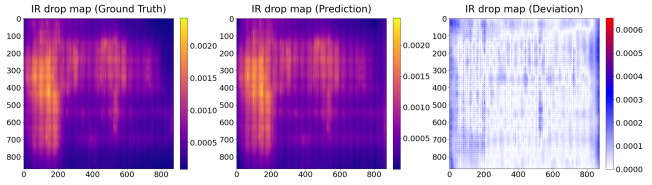
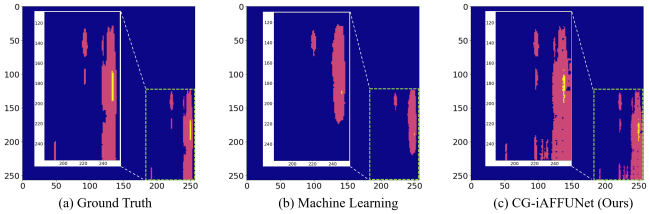


Fig. 9. The comparison between the predicted IR drop map by our model with the corresponding ground truth of the testcase20, and the deviation between the two IR drop maps.



(a) Ground Truth (b) Machine Learning (c) CG-iAFFUNet (Ours)

Fig. 10. The comparison between the predicted hotspot map by pure machine learning-based model, with the corresponding ground truth of the testcase14. The yellow and red areas represent the top 10% and the 10%-50% IR drop regions, respectively. (a) The ground truth hotspot. (b) The pure machine learning-based model. (c) Our CG-iAFFUNet.

B. Effectiveness of Transfer Learning

To validate the effectiveness of the transfer learning approach, we conducted comparative experiments. The comparative results, shown in Fig. 11, demonstrate the predictions from the pre-trained model and the results after fine-tuning the model with the additional 10 unseen real circuit testcases. Compared to the pre-trained model, the fine-tuned transferred model exhibited an average improvement of 12.8% in terms of MAE, with a maximum increase of 30%.

The comparison of MAE before and after fine-tuning reveals a significant improvement in accuracy, illustrating the effectiveness of the fine-tuning process in enhancing the model's generalization capabilities to unseen real circuit datasets. As visualized in Fig. 12, the IR drop map generated by the fine-tuned model exhibits a closer alignment with the ground truth, further evidencing the successful adaptation of the model to the specific characteristics of the real circuit datasets.

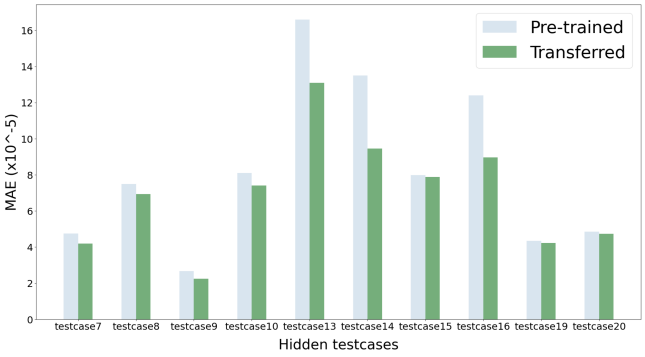
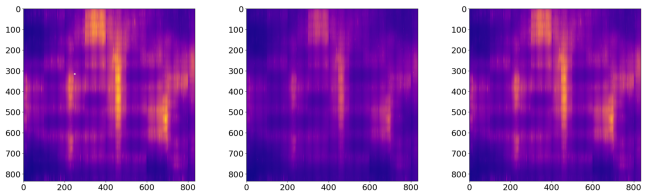


Fig. 11. Comparison of the MAE for hidden testcases between the pre-trained model and the model after transfer learning.



(a) Ground Truth (b) Pre-trained (c) Transferred

Fig. 12. Comparison of the results of applying transfer learning to testcase7. (a) Ground truth IR drop map. (b) Result of the pre-trained model. (c) Result of transferred model with fine-tuning.

C. Comparison on Runtime

In calculating IR drop, striking a trade-off between accuracy and computational speed is critical. In Table I, "RT" encompasses the total duration required for reading input files, preprocessing data, and model inference. We evaluate the runtime of the proposed IR drop calculation model against the top two competitors of the contest. Compared to the champion's performance, our model demonstrates a 39.8% reduction in CPU runtime on average, despite a 48.7% increase in terms of MAE. Although the proposed model does not have a significant advantage in runtime compared to the second place, our model boasts enhanced accuracy, effectively balancing precision with runtime efficiency.

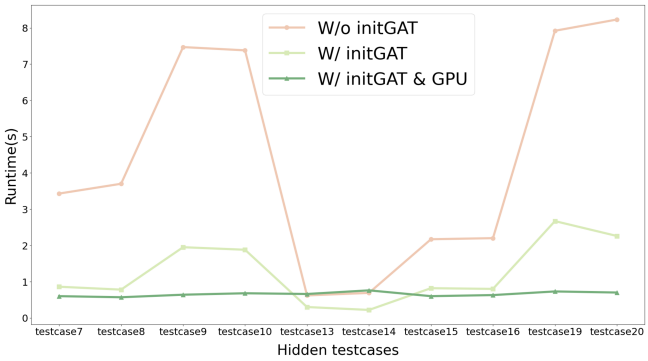


Fig. 13. Results of the ablation study on the use of inexact CG with initial values estimated by initGAT and GPU acceleration in the work.

D. Effectiveness of the initGAT and GPU Acceleration

In this work, we introduce a technique employing GAT to estimate initial values for the CG algorithm and expedite the CG method via GPU acceleration. The efficacy of initGAT is substantiated through ablation studies. "W/o initGAT" denotes the iteration time of the CG algorithm without the initGAT predicted initial values, defaulting all to zero. "W/ initGAT" signifies the time needed to achieve comparable accuracy with initGAT-predicted initial values. "W/ initGAT&GPU" indicates the execution time with both the predicted initial values and GPU acceleration. "Speed-up" shows the ratio of the "W/o initGAT" data to "W/ initGAT" and "W/ initGAT&GPU" data, respectively. After applying initGAT-predicted initial values, the CG algorithm's performance sees a maximum improvement of 4.74×. With the addition of GPU acceleration, this enhancement can reach up to 11.76×. To more vividly display the comparative effects of using initial values estimated by initGAT and GPU acceleration, Fig. 13 illustrates the runtime of inexact CG under different conditions. Furthermore, when using GPU acceleration, there is typically a fixed overhead involved. The acceleration effect becomes more pronounced as the size of the IR drop map increases.

VI. Conclusion

In this work, we propose an accurate IR drop calculation framework utilizing the conjugate gradient optimization and iAFFUNet model. iAFFUNet combines the UNet architecture with iAFF blocks, incorporating MS-CAM modules to bolster information capture across multiple scales. To improve the accuracy of iAFFUNet, we utilize a fast conjugate gradient algorithm to obtain inexact IR drop maps as enhanced features. Additionally, we boost iteration speed by leveraging graph attention networks to estimate initial values for iterative processes. During training, to avoid overfitting and enhance generalizability on unseen real testcases, we deploy the transfer learning strategy, fine-tuning a synthetically pre-trained model with limited real circuit data to ensure the proposed neural network fidelity to real circuit distributions. Experimental comparisons show the effectiveness and efficiency of the proposed method.

Acknowledgment

This work is supported in part by the Major Key Project of PCL (PCL2023A03).

References

[1] J. Lienig, S. Rothe, M. Thiele, N. Rangarajan, M. Ashraf, M. Nabeel, H. Amrouch, O. Sinanoglu, and J. Knechtel, "Toward security closure in the face of reliability effects iccad special session paper," in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 2021, pp. 1–9.

[2] K. Balaskas, G. Zervakis, H. Amrouch, J. Henkel, and K. Siozios, "Automated design approximation to overcome circuit aging," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 11, pp. 4710–4721, 2021.

[3] G. S. P. Kadagala and V. A. Chhabria, "2023 iccad cad contest problem c: Static ir drop estimation using machine learning," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. IEEE, 2023, pp. 1–5.

[4] J. N. Kozhaya, S. R. Nassif, and F. N. Najm, "A multigrid-like technique for power grid analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 10, pp. 1148–1160, 2002.

[5] H. Qian, S. R. Nassif, and S. S. Sapatnekar, "Power grid analysis using random walks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 8, pp. 1204–1224, 2005.

[6] Y. Zhong and M. D. Wong, "Fast algorithms for ir drop analysis in large power grid," in *ICCAD-2005. IEEE/ACM International Conference on Computer-Aided Design, 2005.* IEEE, 2005, pp. 351–357.

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

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

[9] 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 *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2020, pp. 13–18.

[10] 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 *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, 2021, pp. 690–696.

[11] Y. Dai, F. Gieseke, S. Oehmcke, Y. Wu, and K. Barnard, "Attentional feature fusion," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2021, pp. 3560–3569.

[12] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio *et al.*, "Graph attention networks," *stat*, vol. 1050, no. 20, pp. 10–48 550, 2017.

[13] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241.

[15] R. Liang, A. Agnesina, G. Pradipta, V. A. Chhabria, and H. Ren, "Circuitops: An ml infrastructure enabling generative ai for vlsi circuit optimization," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. IEEE, 2023, pp. 1–6.

[16] V. A. Chhabria, K. Kunal, M. Zabihi, and S. S. Sapatnekar, "Began: Power grid benchmark generation using a process-portable gan-based methodology," in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 2021, pp. 1–8.

[17] V. A. Chhabria, A. B. Kahng, M. Kim, U. Mallappa, S. S. Sapatnekar, and B. Xu, "Template-based pdn synthesis in floorplan and placement using classifier and cnn techniques," in *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2020, pp. 44–49.

[18] V. A. Chhabria and S. S. Sapatnekar, "Openpdn: a neural-network-based framework for power delivery network synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 10, pp. 3515–3528, 2021.