

CNN-Cap: Effective Convolutional Neural Network Based Capacitance Models for Full-Chip Parasitic Extraction

Dingcheng Yang, Wenjian Yu*, Yuanbo Guo, Wenjie Liang

Dept. Computer Science & Tech., BNRist, Tsinghua University, Beijing 100084, China

Email: ydc19@mails.tsinghua.edu.cn, yu-wj@tsinghua.edu.cn, guoyb17@mails.tsinghua.edu.cn, liang-wj18@tsinghua.org.cn

Abstract—Accurate capacitance extraction is becoming more important for designing integrated circuits under advanced process technology. The pattern matching based full-chip extraction methodology delivers fast computational speed, but suffers from large error, and tedious efforts on building capacitance models of the increasing structure patterns. In this work, we propose an effective method for building convolutional neural network (CNN) based capacitance models (called CNN-Cap) for two-dimensional (2-D) structures in full-chip capacitance extraction. With a novel grid-based data representation, the proposed method is able to model the pattern with a variable number of conductors, so that largely reduce the number of patterns. Based on the ability of ResNet architecture on capturing spatial information and the proposed training skills, the obtained CNN-Cap exhibits much better performance over the multilayer perception neural network based capacitance model while being more versatile. Extensive experiments on a 55nm and a 15nm process technologies have demonstrated that the error of total capacitance produced with CNN-Cap is always within 1.3% and the error of produced coupling capacitance is less than 10% in over 99.5% probability. CNN-Cap runs more than 4000X faster than 2-D field solver on a GPU server, while it consumes negligible memory compared to the look-up table based capacitance model.

Index Terms—full-chip parasitic extraction, capacitance model, pattern matching, machine learning, convolutional neural network

I. INTRODUCTION

With the continuous down scaling of process technologies, the interconnect wires in integrated circuit (IC) become smaller, closer to each other, and the integration density becomes higher. As a result, modeling effects of the interconnect parasitics (mainly including resistance and capacitance) is increasingly important and crucial for guaranteeing the performance of integrated circuits [1]–[3]. Nowadays, the parasitic modeling (called parasitic extraction) has become one of the essential steps in the design flow, which is the basis of accurate timing analysis and other performance verification [4].

As billions of transistors are placed on a chip, it is challenging to perform the parasitic extraction for tens of billions of interconnect segments. To solve the full-chip parasitic extraction problem, the pattern matching based techniques are the most widely used, such as in StarRC of Synopsys, QRC of Cadence, and other commercial tools. Another approach of parasitic extraction is based on field solver [5]–[9], which has the highest accuracy. However, due to excessive computational cost, the field-solver based approach is not suitable for the full-chip extraction problems.

The pattern matching approach divides an interconnect layout into small geometries and then calculates the capacitances of each geometry with pre-built empirical formulas or look-up tables of capacitance. A *pattern* refers to the geometries sharing similar topology or formula of capacitance. For a given process technology, a pattern library is pre-characterized by enumerating millions of sample geometries and solving the capacitances of each geometry with field solver. Then, the capacitance values are fitted into formulas associated with geometry, or stored as look-up tables. At the time of extraction, through pattern matching the capacitances of input geometries can be calculated quickly and the capacitances of nets are obtained by assembling the capacitances of these geometries.

However, the pattern matching approach is facing the following challenges. 1). The number of patterns increases with the advancement of process technology, and it becomes difficult to make the patterns covering all possible interconnect typologies in real design. 2). The look-up table based approach storing capacitance values of sample geometries consumes enormous or unaffordable memory space for achieving good accuracy, while the error of empirical formulas increases as well. 3). the pattern matching approach needs a large number of capacitance values produced by field solver, which often takes longer time for a process technology as the metal/dielectric configuration becomes complex [3]. So, there is a strong need for new capacitance modeling technique to improve the performance of pattern matching based method.

Although the process of fitting capacitance formulas for a structure pattern is a regression problem and deep neural networks (DNNs) have achieved notable successes in many classification and regression problems in recent years [10], only a few of published work are about employing DNN in the area of parasitic extraction [11]–[15]. Moreover, most of them either deal with the numerical computing in the field-solver approach [12], or do not involve the capacitance calculation for a given interconnect geometry [14]. The most relevant work to the capacitance modeling or calculation is [13], where a neural network based method is presented for several structure patterns in three-dimensional (3-D) ICs. Nevertheless, it only considers single-dielectric structures with simple multilayer perception (MLP) neural networks, and the demonstrated error on total capacitance can be larger than 10% [13]. The practicality and effectiveness of the technique in [13] is obviously not good.

Instead of directly calculating capacitances, an MLP neural network based approach was proposed to improve the pattern matching based capacitance extraction through automatic pattern classification and capacitance formula building [15].

In this work, we propose a convolutional neural network (CNN) based capacitance modeling method for improving the pattern matching based extraction methodology. The major contributions are as follows.

- A grid-based data representation and the corresponding DNN modeling approach are proposed for 2-D structure pattern with a variable number of conductors. It separates the tasks of calculating total capacitance and coupling capacitance, so as to potentially reduce the difficulty of training accurate model for capacitance extraction. Moreover, allowing a pattern to include a variable number of conductors largely reduces the number of patterns and therefore the efforts on building capacitance models.
- A CNN model called CNN-Cap, which is derived from the ResNet architecture and inherits the ability of capturing spatial information, is proposed for predicting the capacitances of the pattern with a variable number of conductors. A training approach including a loss function for more accurate coupling capacitance is proposed to make CNN-Cap suitable for modeling the pattern capacitances.

Extensive experiments with two process technologies demonstrate that the proposed CNN-Cap has much better accuracy on capacitance calculation than the counterpart model based on MLP neural network. For all the tested pattern structures, CNN-Cap is able to predict all total capacitances with less than **1.3%** error, and **over 99.5%** of coupling capacitances with error less than 10%. The sensitivity of the model's performance to the size of training data is also studied, which shows that CNN-Cap performs well with less training effort. Finally, CNN-Cap runs **4693X** faster than 2-D field solver on a GPU server, while it consumes negligible memory compared to the look-up table based capacitance model.

II. BACKGROUND

A. Full-Chip Capacitance Extraction Based on Pattern Matching and 2.5-D Extraction Method

For full-chip capacitance extraction, directly using the 3-D field solver is infeasible due to its excessive cost of memory and CPU time. To obtain good tradeoff between accuracy and efficiency, the 2.5-D extraction method with pattern matching technique is widely used. The pattern matching based extraction methodology include three major modules: 1) pattern generation, 2) capacitance model building, and 3) layout capacitance extraction [2], [9], [15]. It is illustrated as Fig. 1. For a given process technology, the work of 1) pattern generation and 2) capacitance model building are carried out just once.

The 2.5-D extraction method refers to the method considers 3-D geometric effects on capacitance with 2-D interconnect patterns through sweeping 3-D geometry of interconnects in two perpendicular directions [16], [17]. It is adopted by the OpenROAD project for parasitic extraction [18]. During the layout capacitance extraction, extraction windows are generated

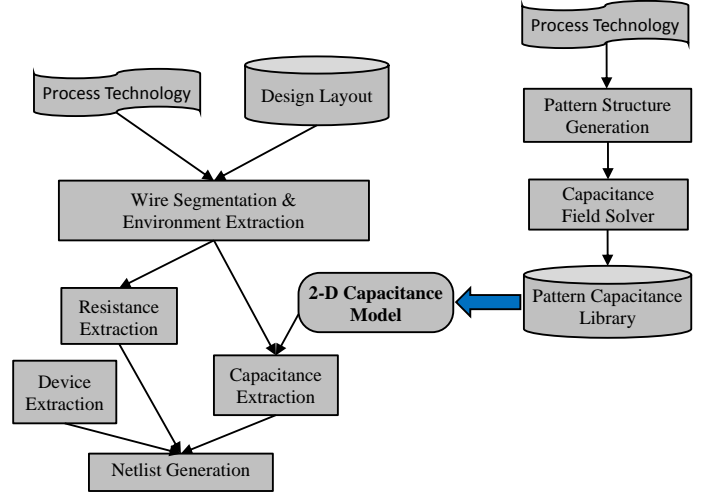


Fig. 1. The pattern matching based full-chip parasitic extraction [3].

along the interconnect line (called *master net* or *master conductor*) whose capacitances are of concern. Each window includes a segment of the master net and its neighbor conductors (called *environmental conductors*). The techniques of 2.5-D extraction and pattern matching are employed to calculate the capacitances among the conductors in the window. Take the structure shown in Fig. 2 as an example, where a wire with name *m2* crosses over a wire named *m1*. Along direction A, a 2-D cross-section view is shown in the middle of Fig. 2. Along direction B, the other 2-D cross section is shown to the right. If the capacitances in the two 2-D cross-section views are known, we can approximately compute the capacitance between *m1* and *m2* with the 3-D effects taken into consideration. Suppose the 2-D capacitance between *m1* and *m2* in the view along A is

$$C_A = C_{1f1} + C_{1o} + C_{1f2}, \quad (1)$$

where C_{1f1} and C_{1f2} are two fringe capacitances, and C_{1o} is the overlapping capacitance. Similarly,

$$C_B = C_{2f1} + C_{2o} + C_{2f2}, \quad (2)$$

for the view along B. Then,

$$C_{m1,m2} = C_A w_1 + (C_B - C_{2o}) w_2, \quad (3)$$

where w_1 and w_2 are widths of wires *m1* and *m2*, respectively.

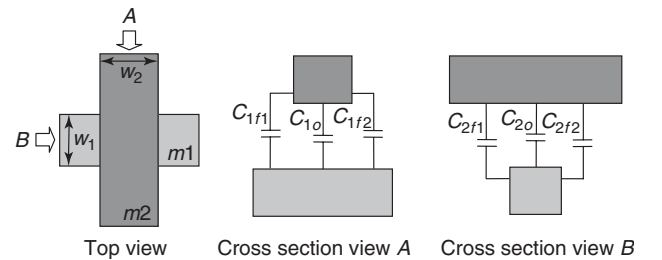


Fig. 2. 2.5-D capacitance calculation for a crossover structure [16].

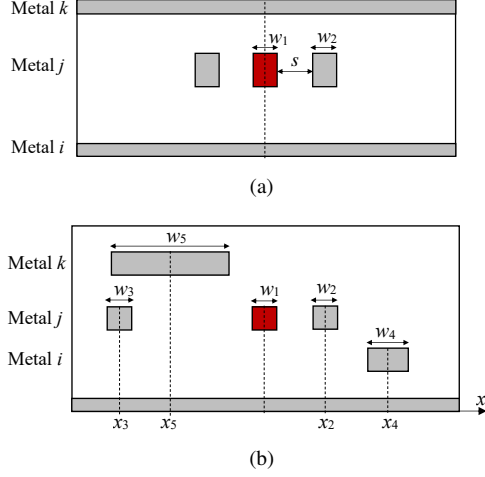


Fig. 3. Two typical 2-D interconnect structure patterns (multi-dielectric environment is not drawn). (a) Pattern-A: “sandwich” structure with three parallel wires. (b) Pattern-B: three metal layers with a fixed number of conductors.

With this method, one can only consider the capacitance models for the 2-D cross-section structures. These structures are regarded as the instances of pre-defined 2-D patterns. Two kinds of typical structure patterns are shown in Fig. 3. Pattern-A shown in Fig. 3(a) is a “sandwich” structure including two big-plane conductors and three parallel wires in between. The red block denotes the master conductor, and the whole structure is usually left-right symmetrical. Pattern-B’s structure is more general than A, where the conductors on the top and down metal layers are not required to be a big plane across the extraction window. On the layer where the master conductor lies, there are two conductors on the left and right side of the master respectively. At the very bottom, there is an extra big-plane conductor representing the substrate ground. Although not shown in Fig. 3, the multi-dielectric environment is considered in these structure.

Usually in a window, only the conductors located on the nearest metal layers above and below the master conductor are considered. Due to the proximity effect of electrostatic field, this brings little error to the capacitances of the master. Therefore, most patterns are defined by three metal layers (containing master conductor and its neighbour conductors) in a given process technology, and the numbers of conductors on each layer. A capacitance model for a pattern is built through computing the capacitances of a lot of instance structures with field solver and then building capacitance models with look-up tables or curve fitting techniques.

B. Neural Network Based Capacitance Extraction

The neural network with multiple hidden layer of neurons is often referred to as deep neural network (DNN). Let $\mathbf{x} \in \mathbb{R}^n$ be the input data and $\mathbf{y} \in \mathbb{R}^m$ be the output data, then, the DNN can be viewed as a function $f(\mathbf{x}; \boldsymbol{\theta}) : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with parameters $\boldsymbol{\theta}$. The form of the function f determines the type of DNN. The training of a DNN and the prediction with a trained DNN are illustrated in Fig. 4. During the training, a large amount of input data along with the corresponding outputs

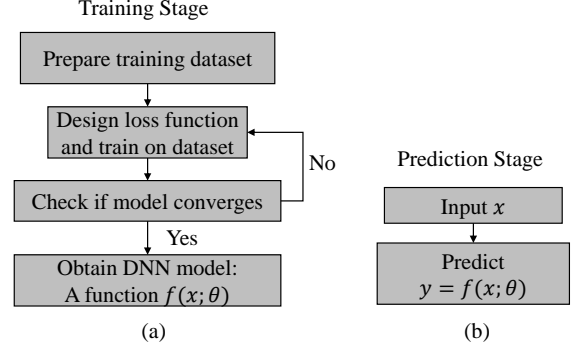


Fig. 4. The work flow of a DNN: (a) training stage, (b) prediction stage.

(called labels) are fed to the network. The network parameters $\boldsymbol{\theta}$ are optimized to minimize the difference between $f(\mathbf{x}; \boldsymbol{\theta})$ and the corresponding labels. When the difference is sufficiently small, the network is trained to become a good regressor, which can be used to do prediction (see Fig. 4(b)).

The MLP neural network is a simple yet popular neural network, as shown in Fig. 5 where only one hidden layer is drawn. Suppose there are n_l hidden layers. Let $\mathbf{h}^{(i)}$ be the variables residing at i -th hidden layer’s neurons. Those on the input layer and output layer can be denoted by $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(n_l+1)}$, respectively. The input data elements are assigned to each neuron in the input layer and feed-forward to the next layer iteratively until they reach the output layer. For the i -th layer, it means $\mathbf{h}^{(i)} = g_i(\mathbf{h}^{(i-1)})$ where g_i is a function to represent the feed-forward process. In general, g_i is a compound function of a nonlinear activation function and a linear function including weight parameters.

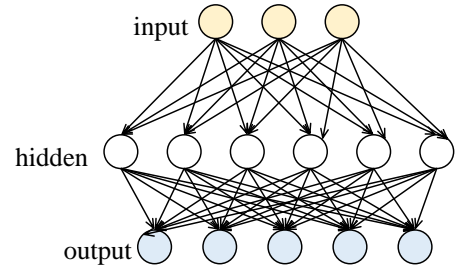


Fig. 5. An example of MLP network. The yellow, white, and blue circles stand for neurons in the input, hidden, and output layers.

The MLP network has been studied in capacitance extraction of several 3-D structures [13]. The idea is straightforward and can be applied to build the capacitance model of the 2-D patterns. For Pattern-A, there are just three width/spacing parameters for describing the structure: w_1, w_2 and s (see Fig. 3(a)). For the example of Pattern-B (see 3(b)), the parameters are widths w_1, w_2, w_3, w_4, w_5 , and location coordinates x_2, x_3, x_4, x_5 . This makes the regression model well defined; the input \mathbf{x} is just a vector including these geometry parameters, and the output \mathbf{y} is a vector including the total capacitance and coupling capacitances. With a lot of sample structures for a pattern and corresponding capacitances results from field solver, the MLP based model can be trained.

Besides training approach and model architecture, different loss function also affects the difficulty of optimization process and the performance of trained model. Usually the mean square error (MSE) is employed as the loss function for minimization:

$$MSE = \frac{1}{N} \sum_{i=1}^N \|f(\mathbf{x}^{(i)}; \boldsymbol{\theta}) - \mathbf{y}^{(i)}\|^2, \quad (4)$$

where N is the number of training examples, $\mathbf{x}^{(i)}$ indicates the i -th example, and $\mathbf{y}^{(i)}$ is the corresponding label.

III. BUILDING EFFECTIVE CAPACITANCE MODELS BASED ON CONVOLUTIONAL NEURAL NETWORK

In this section, we first propose the idea of leveraging CNN to improve the capacitance modeling of structure patterns. Then, we present a grid-based data representation, the CNN architecture and the training approach in details. Finally, we present the data generation approach and other details.

A. The Basic Idea

In all existing methods, a pattern in the 2.5-D extraction flow is determined by a combination of metal layers and the numbers of conductors on these layers. And, the geometric parameters characterizing a pattern increases with the number of conductors in the pattern. For a pattern with a large quantity of parameters, building an accurate model with either traditional approaches or the MLP based approach becomes difficult. To overcome this issue, we view the 2-D cross-section structure as a kind of image, and try to characterize capacitance related information within it with the convolutional neural network which performs very well in image processing applications. So, in this way, we can allow a pattern including a variable number of conductors and thus largely reduce the number of patterns.

This considered pattern is illustrated by Fig. 6. Its structure is more general than that of Pattern-B. The master conductor is still at the center of the middle layer. Although the conductors can lie on more than three metal layers, we assume they are just on three metal layers (with index i , j and k) in the rest of this paper. The proposed method can be easily extended to the patterns with more than three metal layers. The structure in Fig. 6 is referred to as Pattern-C.

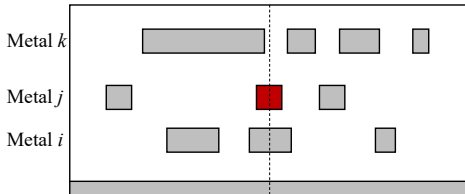


Fig. 6. A structure of three metal layers with a variable number of conductors (dielectric environment not depicted). The structure is referred to as Pattern-C.

B. Grid-Based Data Representation

Inspired by the idea of encoding the layout of interconnect wires with density-based features in the design for manufacture (DFM) research [19], we propose to evenly divide the width

of extraction window into L grid cells. Thus, the conductor placement of a metal layer can be depicted by an L -dimensional vector. The value of vector element is the density, i.e. the ratio of conductor occupying the grid cell. See the example in Fig. 7, where the horizontal placement of three conductors (labeled 1, 2 and 3), the grids, and the corresponding density map are shown. For a Pattern-C structure, the conductor placement can be described with three L -dimensional vectors. Notice this representation is not ambiguous if we guarantee that the grid size is less than the minimum spacing between wires on the metal layer.

			2				1				3				
Density map	0	0	1	1	1	0.1	0.8	1	1	0.4	0.3	1	1	0.9	0
Feature for C_{11}	0	0	1	1	1	0.1	1.8	2	2	1.4	0.3	1	1	0.9	0
Feature for C_{12}	0	0	-1	-1	-1	-0.1	1.8	2	2	1.4	0.3	1	1	0.9	0
Feature for C_{13}	0	0	1	1	1	0.1	1.8	2	2	1.4	-0.3	-1	-1	-0.9	0

Fig. 7. Grid-based data representation for a metal layer in a pattern. A case with three conductors is shown in the first row. The following rows are the density map and the data representations for the calculation of total capacitance and coupling capacitances.

This grid-based representation cannot identify the master conductor. So, we must encode more information regarding the pattern into it. Suppose the structure includes n_c conductors. Our problem is to extract one total capacitance, and $n_c - 1$ coupling capacitances. Consider the density-based representation for one metal layer: $\mathbf{d} \in \mathbb{R}^L$. We modify it to obtain an L -dimensional vector \mathbf{x} for a sub-problem of calculating total capacitance. The scheme of the modification is that, if the master conductor covers the i -th grid, we have $x_i = d_i + 1$. The obtained feature vector \mathbf{x} is shown as the third row in Fig. 7. For calculating the coupling capacitance between the master and an environmental conductor, a unique feature vector is also generated by modifying \mathbf{d} . Besides adding 1 to the elements corresponding the cells overlapped with the master, another modification is applied to identifying the environmental conductor. If the environmental conductor covers the i -th grid, we make that $x_i = -d_i$. Now, for calculating different coupling capacitance there is a different feature vector for data representation. In Fig. 7, the last two rows show the feature vectors for the sub-problems of calculating coupling capacitances C_{12} and C_{13} .

Because the values of total capacitance and coupling capacitance can differ for several orders of magnitude, the accuracy demand for them is usually different. In this work, we distinguish the problems of calculating total capacitance and of coupling capacitances, and trains two models for them respectively for a given Pattern-C. This ensures the overall accuracy of capacitance extraction. With this idea and the proposed grid-based data representation, our method for building the capacitance models can be depicted as Fig. 8. Notice that a sample structure is converted to n_c data inputted to the two models in the training stage. And, in the prediction stage, the trained model Model- C_C for coupling capacitance will be evaluated for $n_c - 1$ times to output the $n_c - 1$ coupling capacitances.

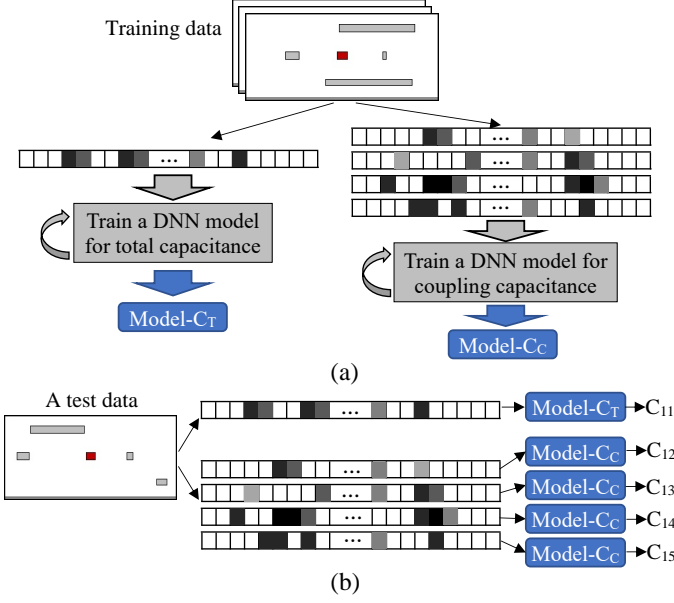


Fig. 8. The proposed method for building capacitance models for Pattern-C. (a) The training stage. (b) The prediction stage.

C. CNN Architecture and Training Approach

Because the CNN is able to capture the spatial information in the data, it is expected to perform better than MLP neural network for the problem of capacitance calculation. So, we develop the CNN base models (called CNN-Cap) for predicting the capacitances of Pattern-C structures. There are two models with same architecture used for total capacitance and coupling capacitance, respectively.

The architecture of CNN-Cap is shown in Fig. 9, which is derived from the famous ResNet architecture [20]. In our problem, the input data (pattern structure with extraction information) is vectors, instead of the matrices in image related problem. So, we just use 1-D convolutional layer (shown as colored block in Fig. 9) in CNN-Cap. Notice that the number of metal layers in the pattern is like the concept of “channel” of image data. A batch normalization and a ReLU layer are inserted after every convolutional layer. The input data will reach the last convolutional layer through stacked convolutional layers. Then, it will be pooled into a fixed-length vector by an average pooling layer. Finally, the fixed-length vector will be fed to the fully-connected layer with an output dimension of one to predict capacitance. In Fig. 9, the convolutional layer captioned “1x3 conv, 64” means it has 64 channels and a kernel size of 3. “/2” means to halve the input length (by a 1-D pooling layer or a 1-D convolutional layer with a stride of 2). The convolutional layers with the same color mean they have the same input length. Whenever a 1-D convolutional layer halves the input length, the input channel will be doubled. The key point of CNN-Cap is the shortcut connection, which takes a shallower vector h_1 and a deeper vector h_2 as input and takes $\mathcal{F}(h_1) + h_2$ as output. The function \mathcal{F} is an identity mapping in most cases (solid line). The dotted shortcut takes a 1-D 1x1 convolutional layer (equivalent to linear projection) as

the mapping function when the shape of two input vectors is mismatched. Without the shortcut connection, a deeper DNN model will have both higher testing error and training error. The higher training error means that it is difficult to train a deep plain model. The shortcut connection can preserve the low-level features all the time, which makes training easier. Thanks to the advanced modules, i.e., 1-D convolutional layer and shortcut connect, our CNN-Cap has better learning ability than the MLP neural networks. This will be revealed with experimental results.

We have tested three architectures derived from ResNet-18, ResNet-34, and ResNet-50. We found out that derived from ResNet-34 is consistently better than that from ResNet-18, and performs similarly to that from ResNet-50. So, the ResNet-34 derived architecture is chosen in this work.

For training CNN-Cap, the stochastic gradient descent (SGD) optimizer and the Adam optimizer [21] are considered. Our experimental results show the Adam optimizer makes convergence easier to reach and the trained model perform better. So, it is employed in this work.

The approach of grid search is used to find the appropriate learning rate and batch size, which affect performance of the obtained CNN-Cap model. The value range of learning rate is from 10^{-5} to 10^{-2} , and the batch size is enumerated in $\{16, 32, 64, 128\}$.

For the cost function, in addition to the MSE loss function (4), we have tried the following loss function.

$$MSRE = \frac{1}{N} \sum_{i=1}^N \left(1 - \frac{f(\mathbf{x}^{(i)}; \boldsymbol{\theta})}{\mathbf{y}^{(i)}} \right)^2, \quad (5)$$

where N is the number of training data, $\mathbf{x}^{(i)}$ indicates the i -th input data, and $\mathbf{y}^{(i)}$ is the corresponding label. The division of (5) stands for an element-wise operation. This MSRE loss function includes the relative error, so that the optimization could possibly attain a better accuracy. However, it may bring difficulty to the convergence of training process. We have done extensive experiments, and found out that the loss function

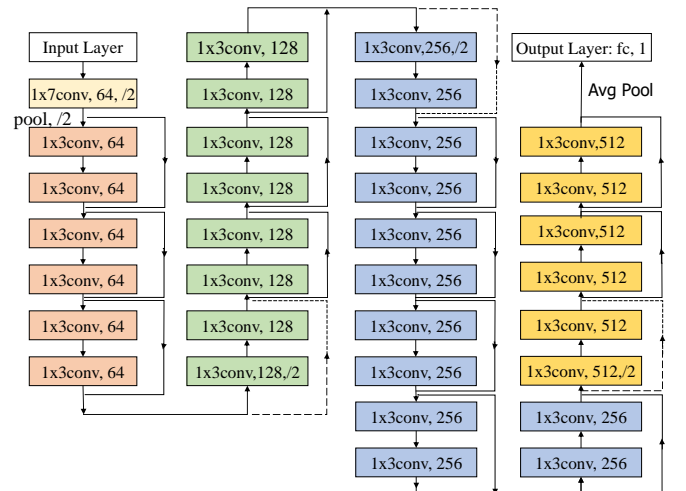


Fig. 9. The architecture of proposed CNN-Cap.

MSRE is better than MSE for the task of training the Model- C_C for coupling capacitance.

D. Dataset Generation and Other Discussion

In this work, a data is a 2-D structure of cross-section view with the proposed data representation, and the label includes the capacitances computed with a field solver.

For a given process technology and a specified triple of (i, j, k) of metal layer indices, we can generate random sample structures of Pattern-C. We ensure that the data complies with the rules of minimum width and minimum spacing of the process technology. As the structure is a cross-section view along the interconnect line, we let the width of master conductor be a smaller value with larger probability and not exceed 10 times of the minimum width. Similarly, the random data can be generated for the conventional patterns, like Pattern-A and Pattern-B. Notice we can also build the CNN-Cap model for a conventional pattern.

The whole width of the extraction window is determined with a simulation experiment to detect how far if an environmental conductor is away from the master conductor the coupling capacitance between them decreases to 1% of the total capacitance.

Fig. 10 shows the geometries of the conductors in two data generated with our approach. Fig. 10(a) stands for a data for Pattern-B and Fig. 10(b) is for Pattern-C.

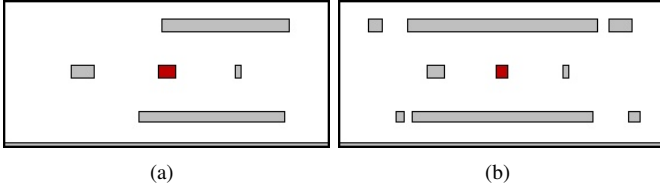


Fig. 10. Two randomly generated data for: (a) Pattern-B and (b) Pattern-C. The dielectric environment is not depicted.

The existing approach with MLP neural network cannot handle Pattern-C, because the structure involves a variable number of conductors. Instead, we can combine the MLP with the grid-based data representation. It means we connect the three vectors to a form a long vector and then input it to the MLP network. However, this hybrid model cannot have high accuracy, because the spatial information in the grid-based representation is lost. The comparison experiments in Section IV.B validate this.

IV. EXPERIMENTAL RESULTS

2-D pattern structures are generated following two process technologies. One is a 55nm process technology from industry, while the other is the 15nm process technology in FreePDK15 [22], [23]. The width of structure (extraction window) is set to $56\tilde{w}_{min}$, where \tilde{w}_{min} is the minimum width of the metal layer where the master conductor resides. For the structures, we obtain the capacitance results (as labels of data) with the golden-standard capacitance solver Raphael [24]. All tested

DNN models are implemented with PyTorch, and all experiments are carried out on a Linux server with 2 Intel Xeon Silver 4214 CPU at 2.2GHz and 8 Nvidia RTX2080Ti GPUs.

The geometric parameters for the metal layers in the two process technologies are listed in Table I.

TABLE I
THE GEOMETRIC PARAMETERS OF THE METAL LAYERS IN THE TWO TESTED PROCESS TECHNOLOGIES (IN UNIT OF μm)

Layer index	55nm Tech.			15nm Tech.		
	thickness	w_{min}	s_{min}	thickness	w_{min}	s_{min}
1	0.1	0.054	0.108	0.06	0.028	0.036
2	0.16	0.081	0.08	0.06	0.028	0.036
3	0.2	0.09	0.09	0.06	0.028	0.036
4	0.2	0.09	0.09	0.06	0.028	0.036
5	0.2	0.09	0.09	0.06	0.028	0.036
6	0.2	0.09	0.09	0.13	0.056	0.056
7	0.85	0.36	0.36	0.13	0.056	0.056
8	-	-	-	0.13	0.056	0.056
9	-	-	-	0.13	0.056	0.056
10	-	-	-	0.13	0.056	0.056

For the two technologies, we randomly choose some three-metal-layer combinations to generate the 2-D patterns. For each pattern, we generated 50000 sample structures with the approach in Section III.D to form a dataset. Each dataset is then randomly split into a training subset (with 90% of the samples) and a testing subset (with 10% of the samples). For CNN-Cap which utilizes the grid-based data, we convert a sample structure to n_c data where n_c is the number of conductors in the sample. The number of grid cells (L) is set to 1024, which ensures that the grid size is less than the half of minimum spacing at any metal layer. To avoid the interference of very small coupling capacitance, the coupling capacitance whose value is less than 1% of the corresponding total capacitance is not considered in the training stage and the prediction stage.

To tune the hyper-parameters for CNN-Cap, we randomly choose a layer combination under the 55nm technology to generate a dataset of Pattern-C. With it we tune the hyper-parameters to make the CNN-Cap's performance on the testing subset of this dataset best. The obtained hyper-parameters include batch size set to 64, learning rate set to 10^{-4} for predicting total capacitance and 10^{-5} for predicting coupling capacitance.

In the following subsections we will first evaluate the performance of CNN-Cap on Pattern-B structures, and then evaluate its performance on Pattern-C structures. Finally, we will study the effect of training subset's size and other evaluation metrics.

A. Results for Pattern-B Structures

For Pattern-B structures, the MLP-based model presented in Section II.B can be used as the baseline. We call it MLP-Cap. Similar tuning is also applied to MLP-Cap, with a dataset of Pattern-B. For the architecture, we tried different numbers of hidden layers and different numbers of neurons respectively. After some heuristic trials, we finally use an architecture with three hidden layers, and 256, 256 and 512 neurons are set in the three layers respectively. Three nonlinear activation functions: ReLU, Sigmoid and Tanh are tested. The results show that Tanh function consistently surpass the other two. The grid search

is used to find the appropriate learning rate and batch size. Finally, the batch size is set to 32, and the learning rate is set to 10^{-5} . Besides, the loss functions for MSE (4) and MSRE (5) are tested with MLP-Cap. The results show that with the MSE loss function makes MLP-Cap perform better.

In this experiment, the pattern is the same as that in Fig. 3(b) which includes 1, 3 and 1 conductors on three metal layers respectively. And, the structure can be described with 9 geometric parameters, which are the input to MLP-Cap.

The results for predicting total capacitance and coupling capacitance, for eight datasets from the both technologies, are shown in Table II and Table III, respectively. In the tables, every two rows correspond to a data set generated for a pattern. Err_{avg} means the average of the relative error's absolute value. Err_{max} means the maximum of the relative error's absolute value. Ratio($Err > 5\%$) in Table II means the ratio of the number of total capacitances with error larger than 5% to the number of all total capacitances predicted. Ratio($Err > 10\%$) in Table III means the ratio of the number of coupling capacitances with error larger than 10% to the number of all coupling capacitances predicted. Here, we regard relative error of 5% as the accuracy criterion for total capacitance, and the relative error of 10% as the accuracy criterion for coupling capacitance.

The experimental results show that CNN-Cap always performs better than MLP-Cap. In more details, both CNN-Cap and MLP-Cap can predict total capacitance with less than 1% relative error on average, while the maximum relative error of CNN-Cap is not larger than **1.3%**. However, the maximum relative error of MLP-Cap is often larger than 5%. As for predicting coupling capacitance, the maximum relative error of MLP-Cap is always larger than 10% and can be as large as 114%. On the contrary, only a very few of coupling capacitances (at most **0.1%** of all coupling capacitances) computed with CNN-Cap have relative error larger than 10%.

TABLE II
DNN MODELS' PERFORMANCE ON TOTAL CAPACITANCE FOR PATTERN-B

Tech. Node	Layers	Method	Err_{avg}	Err_{max}	Ratio($Err > 5\%$)
55nm	(2, 3, 6)	MLP-Cap	0.8%	8.1%	0.4%
55nm	(2, 3, 6)	CNN-Cap	0.2%	1.2%	0
55nm	(2, 4, 6)	MLP-Cap	1.2%	11.4%	0.7%
55nm	(2, 4, 6)	CNN-Cap	0.2%	1.0%	0
15nm	(1, 3, 5)	MLP-Cap	0.5%	3.8%	0
15nm	(1, 3, 5)	CNN-Cap	0.2%	1.3%	0
15nm	(1, 3, 8)	MLP-Cap	0.5%	5.2%	0.0%
15nm	(1, 3, 8)	CNN-Cap	0.2%	1.1%	0

TABLE III
DNN MODELS' PERFORMANCE ON COUPLING CAPACITANCE FOR PATTERN-B

Tech. Node	Layers	Method	Err_{avg}	Err_{max}	Ratio($Err > 10\%$)
55nm	(2, 3, 6)	MLP-Cap	3.3%	114%	6.6%
55nm	(2, 3, 6)	CNN-Cap	2.4%	13.5%	0.1%
55nm	(2, 4, 6)	MLP-Cap	2.5%	89.2%	4.4%
55nm	(2, 4, 6)	CNN-Cap	1.4%	11.8%	0.0%
15nm	(1, 3, 5)	MLP-Cap	2.5%	87%	3.7%
15nm	(1, 3, 5)	CNN-Cap	1.1%	9.6%	0
15nm	(1, 3, 8)	MLP-Cap	2.0%	49.0%	1.4%
15nm	(1, 3, 8)	CNN-Cap	0.9%	14.3%	0.1%

Fig. 11 shows the coupling capacitances calculated with MLP-Cap and CNN-Caps respectively, along the relative error of each capacitance. From the figure we see that CNN-Cap can accurately predict most coupling capacitance, and the few examples of inaccurate prediction usually corresponds to a small coupling capacitance. This means reasonable and acceptable accuracy. On the contrary, the MLP-Cap is unsatisfactory in many cases, and its maximum error is very large as shown in Fig. 11(a).

More dataset for different layer combinations have been generated and tested. The results show that for all the tested datasets, the average relative error of CNN-Cap on total capacitance and coupling capacitance are just 0.22% and 1.36%, respectively. This experiment demonstrates the good accuracy of CNN-Cap for Pattern-B structures.

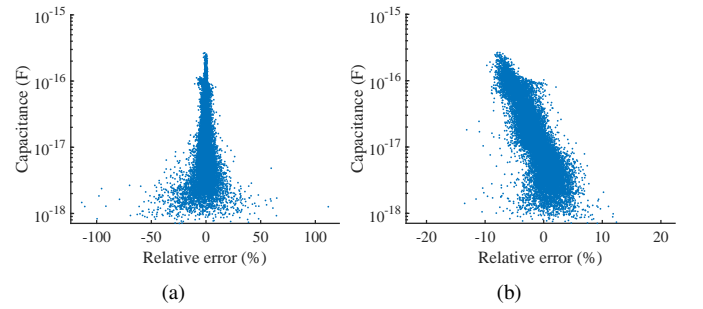


Fig. 11. The calculated coupling capacitance versus relative error for Pattern-B. The dataset corresponds (2, 3, 6) layer combination in the 55nm technology. (a) Results of MLP-Cap. (b) Results of CNN-Cap.

Another capacitance modeling approach which may have good accuracy is based on look-up table and the bilinear interpolation. For the considered Pattern-B structures, if each parameter takes 20 sample values the total numbers stored in the look-up table would be $20^9 = 5.12 \times 10^{11}$, which leads to unacceptable memory cost in reality. This demonstrates the advantage of capacitance modeling technique based on neural networks.

B. Results for Pattern-C Structures

For generating the sample structures of Pattern-C, we assume the total number of conductors on the up and down metal layers ranges from 6 to 8. Such a pattern with a variable number of conductors cannot be handled by a single MLP-Cap. So, we consider the hybrid method discussed at the end of Section III as the baseline. It is called Grid+MLP model. It utilizes the grid-based data representation and shares the same architecture and hyper-parameters as MLP-Cap. And, it is trained with the same approach for MLP-Cap.

The results of CNN-Cap and Grid+MLP model for four datasets of Pattern-C are listed in Table IV and Table V. The results show that CNN-Cap performs much better than Grid+MLP. The maximum relative error of CNN-Cap on total capacitance is not larger than **1.2%**. As for coupling capacitance, the performance of the Grid+MLP is much worse and unacceptable, with the maximum error as large as 542%. On the contrary, the relative error derived from the CNN-Cap is less than 10% for more than **99.5%** predicted capacitances.

TABLE IV
DNN MODELS' PERFORMANCE ON TOTAL CAPACITANCE FOR PATTERN-C

Tech. Node	Layers	Method	Err_{avg}	Err_{max}	Ratio(Err>5%)
55nm	(2, 3, 6)	Grid+MLP	0.4%	7.5%	0.1%
55nm	(2, 3, 6)	CNN-Cap	0.1%	1.0%	0
55nm	(2, 4, 6)	Grid+MLP	0.7%	7.7%	0.1%
55nm	(2, 4, 6)	CNN-Cap	0.2%	1.1%	0
15nm	(1, 3, 5)	Grid+MLP	0.7%	4.3%	0
15nm	(1, 3, 5)	CNN-Cap	0.2%	1.1%	0
15nm	(1, 3, 8)	Grid+MLP	0.5%	5.4%	0.1%
15nm	(1, 3, 8)	CNN-Cap	0.2%	1.2%	0

TABLE V
DNN MODELS' PERFORMANCE ON COUPLING CAPACITANCE FOR PATTERN-C

Tech. Node	Layers	Method	Err_{avg}	Err_{max}	Ratio(Err>10%)
55nm	(2, 3, 6)	Grid+MLP	10.4%	542.4%	27.1%
55nm	(2, 3, 6)	CNN-Cap	1.2%	38.6%	0.3%
55nm	(2, 4, 6)	Grid+MLP	9.1%	489.3 %	22.7%
55nm	(2, 4, 6)	CNN-Cap	1.2%	14.4%	0.1%
15nm	(1, 3, 5)	Grid+MLP	9.8%	492.8%	24.8%
15nm	(1, 3, 5)	CNN-Cap	1.8%	38.8%	0.4%
15nm	(1, 3, 8)	Grid+MLP	11.4%	390.4%	30.8%
15nm	(1, 3, 8)	CNN-Cap	1.5%	12.4%	0.0%

Fig. 12 shows the calculated coupling capacitance (by CNN-Cap) versus relative error for Pattern-C structures. It reveals again that CNN-Cap has very good accuracy on coupling capacitance for most structures. For example, for the test data corresponding to layer combination (2, 3, 6) in the 55nm technology, only 0.3% of all coupling capacitances has error larger than 10%.

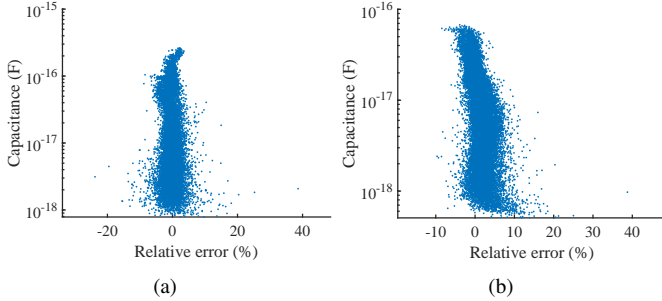


Fig. 12. The calculated coupling capacitance versus relative error for Pattern-C. (a) Results of CNN-Cap for layer combination (2, 3, 6) in the 55nm technology. (b) Results of CNN-Cap for layer combination (1, 3, 5) in the 15nm technology.

More dataset for different layer combinations have been generated and tested. The results show that, for all the tested datasets the average relative error of CNN-Cap on total capacitance and coupling capacitance are just 0.18% and 1.38%, respectively. This experiment demonstrates the good accuracy of CNN-Cap for Pattern-C structures.

C. The Effect of Training Set's Size and More Comparisons

An additional experiment is carried out to evaluate the effect of training set's size on CNN-Cap's accuracy. We change the ratio of the training subset from 90% to 80%, 70%, down to 10%, and then rerun the training process for a dataset of Pattern-C, respectively. For each resulted CNN-Cap model, we examine it with the testing subset. The average relative error on

coupling capacitance and the ratio of the coupling capacitances with error larger than 10% are plotted in Fig. 13. From it we

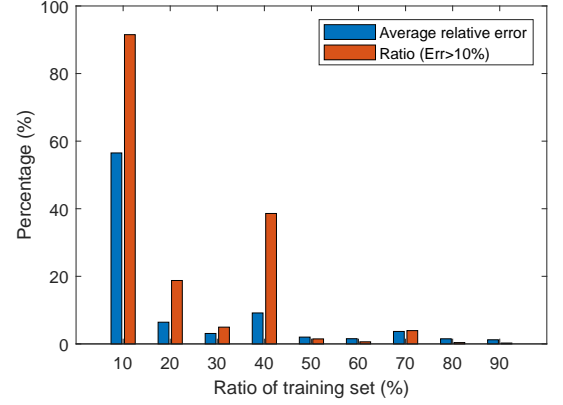


Fig. 13. The accuracy of CNN-Cap on coupling capacitance vs. the size of training subset.

see that, when the ratio of training subset is 50% or larger (meaning with 25000 data or more) the trained CNN-Cap model always has fairly good accuracy. Similar results are observed for the total capacitance. Therefore, a training set with 25000 data through 45000 data is usually sufficient for building an effective CNN-Cap model.

We compare the prediction time of CNN-Cap with Raphael, i.e. the 2-D field solver `rc2` within Raphael. For a dataset including 50000 Pattern-C structures, we extract the capacitances of them with CNN-Cap and Raphael respectively. Their average runtimes per structure are listed in Table VI. From the table we see that CNN-Cap runs **4693X** faster than Raphael. Even though we can run multiple processes of Raphael on the machine, say 32 processes, calculating capacitance with CNN-Cap can still be **147X** faster than running the 2-D field solver.

TABLE VI
THE AVERAGE RUNTIMES OF CNN-CAP AND RAPHAEL FOR CALCULATING THE CAPACITANCES OF A PATTERN-C STRUCTURE

Method	Time (ms)	Speedup
Raphael	704	—
CNN-Cap	0.15	4693X

As for the model size, a CNN-Cap model has 14473418 parameters occupying 13.8 MB storage, which is much smaller than the look-up table based model. The training time for a CNN-Cap model is about 1.3 hours. Considering various layer combinations for a given process technology with 10 metal layers (like that in FreePDK15), we see that building all CNN-Cap models can be completed within a week on a GPU server with 8 Nvidia RTX2080Ti GPUs. And, the CNN-Cap models deliver much better accuracy than the existing methods for pattern capacitance modeling.

V. CONCLUSIONS

In this work, a CNN based capacitance model called CNN-Cap and the corresponding model-building techniques are proposed. They include a grid-based data representation for

2-D pattern structures, and a ResNet-like CNN architecture and corresponding training approach. CNN-Cap is able to predict the total capacitance and coupling capacitances for the 2-D pattern with a variable number of conductors. This largely reduces the number of patterns and the corresponding capacitance models employed in the pattern matching based full-chip capacitance extraction. Extensive experiments have demonstrated the advantages of CNN-Cap over MLP based models and traditional model-building approaches.

In the future, the proposed method may be extended to build capacitance models for 3-D interconnect structures.

REFERENCES

- [1] U. Choudhury and A. Sangiovanni-Vincentelli, "Automatic generation of analytical models for interconnect capacitances," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 4, pp. 470–480, 1995.
- [2] L. Lavagno, L. Scheffer, and G. Martin, *EDA for IC Implementation, Circuit Design, and Process Technology*. CRC press, 2006.
- [3] W. Yu, M. Song, and M. Yang, "Advancements and challenges on parasitic extraction for advanced process technologies," in *26th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2021, pp. 841–846.
- [4] W. Gong, W. Yu, Y. Lü, Q. Tang, Q. Zhou, and Y. Cai, "A parasitic extraction method of VLSI interconnects for pre-route timing analysis," in *Proc. International Conference on Communications, Circuits and Systems (ICCCAS)*, 2010, pp. 871–875.
- [5] K. Nabors and J. White, "FastCap: A multipole accelerated 3-D capacitance extraction program," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, no. 11, pp. 1447–1459, 1991.
- [6] Y. Le Coz and R. Iverson, "A stochastic algorithm for high speed capacitance extraction in integrated circuits," *Solid-State Electronics*, vol. 35, no. 7, pp. 1005–1012, 1992.
- [7] W. Yu, H. Zhuang, C. Zhang, G. Hu, and Z. Liu, "RWCAP: A floating random walk solver for 3-D capacitance extraction of very-large-scale integration interconnects," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 3, pp. 353–366, 2013.
- [8] X. Wang, D. Liu, W. Yu, and Z. Wang, "Improved boundary element method for fast 3-D interconnect resistance extraction," *IEICE Transactions on Electronics*, vol. 88, no. 2, pp. 232–240, 2005.
- [9] W. Yu and X. Wang, *Advanced Field-Solver Techniques for RC Extraction of Integrated Circuits*. Springer, 2014.
- [10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [11] P. Sen, W. H. Woods, S. Sarkar, R. J. Pratap, B. M. Dufrene, R. Mukhopadhyay, C.-H. Lee, E. F. Mina, and J. Laskar, "Neural-network-based parasitic modeling and extraction verification for RF/millimeter-wave integrated circuit design," *IEEE Transactions on Microwave Theory and Techniques*, vol. 54, no. 6, pp. 2604–2614, 2006.
- [12] H. Yao, Y. Qin, and L. Jiang, "Machine learning based mom (ML-MoM) for parasitic capacitance extractions," in *Proc. IEEE Electrical Design of Advanced Packaging and Systems (EDAPS) Symposium*, 2016, pp. 171–173.
- [13] R. Kasai, T. Kanamoto, M. Imai, A. Kurokawa, and K. Hachiya, "Neural network-based 3D IC interconnect capacitance extraction," in *Proc. International Conference on Communication Engineering and Technology (ICCET)*, 2019, pp. 168–172.
- [14] B. Shook, P. Bhansali, C. Kashyap, C. Amin, and S. Joshi, "MLParest: Machine learning based parasitic estimation for custom circuit design," in *Proc. Design Automation Conference*, 2020, pp. 1–6.
- [15] Z. Li and W. Shi, "Layout capacitance extraction using automatic pre-characterization and machine learning," in *Proc. ISQED*, 2020, pp. 457–464.
- [16] W. Yu and Z. Wang, "Capacitance extraction," *Encyclopedia of RF and Microwave Engineering*, 2005.
- [17] J. Cong, L. He, A. Kahng, D. Noye, N. Shiral, and S. Yen, "Analysis and justification of a simple, practical $2^{1/2}$ -D capacitance extraction methodology," in *Proc. Design Automation Conference*, 1997, pp. 627–632.
- [18] OpenRCX, 2021, [Online]. Available: <https://github.com/The-OpenROAD-Project/OpenRCX>.
- [19] W.-Y. Wen, J.-C. Li, S.-Y. Lin, J.-Y. Chen, and S.-C. Chang, "A fuzzy-matching model with grid reduction for lithography hotspot detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 11, pp. 1671–1680, 2014.
- [20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
- [21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.
- [22] K. Bhanushali and W. R. Davis, "FreePDK15: An open-source predictive process design kit for 15nm FinFET technology," in *Proceedings of the 2015 Symposium on International Symposium on Physical Design*, 2015, pp. 165–170.
- [23] FreePDK15, 2020, [Online]. Available: <https://www.eda.ncsu.edu/wiki/FreePDK15:Contents>.
- [24] Synopsys Inc. *Raphael*. Accessed: Jan. 2021. [Online]. Available: <http://www.synopsys.com/silicon/tcad/interconnect-simulation/raphael.html>.