CNN-Cap: Effective Convolutional Neural Network-based Capacitance Models for Interconnect Capacitance Extraction

DINGCHENG YANG, HAOYUAN LI, WENJIAN YU, YUANBO GUO, and WENJIE LIANG, Department of Computer Science and Technology, BNRist, Tsinghua University, China

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) and three-dimensional (3-D) interconnect structures. With a novel grid-based data representation, the proposed method is able to model 2-D pattern structure and 3-D window structure with a variable number of conductors to largely reduce the number of patterns or increase the accuracy. 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 55 nm and a 15 nm process technologies have demonstrated that the error of total capacitance produced with 2-D CNN-Cap is always within 1.3%, and the error of produced coupling capacitance is less than 10% in over 99.5% probability. For 3-D structures, CNN-Cap predicts the total capacitance with less than 5% error in 99% probability and with a maximum error of 7.7%. For the tested 2-D and 3-D structures, the CNN-Cap run on a GPU server is more than 4,000× and 12,000×, respectively, faster than the conventional field solver Raphael, while consuming negligible memory.

CCS Concepts: • **Hardware → Modeling and parameter extraction**; • **Computing methodologies → Neural networks;**

Additional Key Words and Phrases: Interconnect capacitance extraction, pattern matching, machine learning, convolutional neural network

ACM Reference format:
Dingcheng Yang, Haoyuan Li, Wenjian Yu, Yuanbo Guo, and Wenjie Liang. 2023. CNN-Cap: Effective Convolutional Neural Network-based Capacitance Models for Interconnect Capacitance Extraction. *ACM Trans. Des. Autom. Electron. Syst.* 28, 4, Article 56 (May 2023), 22 pages.
https://doi.org/10.1145/3564931

This work is partially supported by the National Key R&D Program of China (Grant No. 2021ZD0114703) and the National Natural Science Foundation of China (Grant No. 62090025). The preliminary version has been presented at the IEEE/ACM International Conference on Computer-Aided Design (ICCAD) in 2021.
Authors' addresses: D. Yang, H. Li, W. Yu (corresponding author), Y. Guo, and W. Liang, Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing, China, 100084; emails: {ydc19, lihaoyua17}@mails.tsinghua.edu.cn, yu-wj@tsinghua.edu.cn, guoyb17@mails.tsinghua.edu.cn, liang-wj18@tsinghua.org.cn.

1084-4309/2023/05-ART56 $15.00
https://doi.org/10.1145/3564931

1 INTRODUCTION

With the continuous down scaling of process technologies, the interconnect wires in **integrated circuits (ICs)** 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 [3, 11, 24]. 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 [6].

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 [8, 12, 15, 19, 22, 25, 27, 28], 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 [24]. These make the commercial tool based on pattern-matching may result in large error even on total capacitance. So, there is a strong need for new capacitance modeling technique to improve both accuracy and runtime performance of the 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 [13], only a few published works are about employing DNN in the area of parasitic extraction [9, 14, 17, 18, 23]. Moreover, most of them either deal with the numerical computing in the field-solver approach [23] or do not involve the capacitance calculation for a given interconnect geometry [18]. The most relevant work to the capacitance modeling or calculation is Reference [9], 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 perceptron (MLP)** neural networks, and the demonstrated error on total capacitance can be larger than 10% [9]. The practicality and effectiveness of the technique in Reference [9] 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 [14]. Recently, a machine learning modeling approach was proposed for capacitance extraction of

middle-end-of-line (MEOL) structures and interconnects with systematic process variations [1]. It utilizes the MLP neural network or the support vector regression technique and exhibits good accuracy for computing the capacitances in the regular structure of MEOL pattern.

In this work, we aim to develop DNN-based techniques to improve the pattern-matching-based capacitance extraction methodology for general **back-end-of-line (BEOL)** interconnect structures. The major contributions are as follows:

- A grid-based data representation and the corresponding DNN modeling approach are proposed for 2-D and 3-D interconnect structures, which may include a variable number of conductors. It separates the tasks of calculating total capacitance and coupling capacitance to potentially reduce the difficulty of training accurate model for capacitance extraction. For the pattern-matching-based methodology, 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 **convolutional neural network (CNN)**-based model, called CNN-Cap, which is derived from the ResNet architecture and inherits the ability of capturing spatial information, is proposed to predict the capacitances of 2-D or 3-D structures 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 capacitance extraction. The 2-D CNN-Cap can replace field solver for the pattern building procedure in the pattern-matching-based methodology, while the 3-D CNN-Cap can improve the accuracy resulted from the pattern matching with much less cost than invoking field solver.

Extensive experiments with two process technologies demonstrate that the proposed CNN-Cap for 2-D pattern has much better accuracy on capacitance calculation than the counterpart model based on MLP neural network. For all the tested 2-D 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. For the tested 3-D structures, CNN-Cap can predict **over 99%** of total capacitance with less than 5% error and **over 95%** of coupling capacitances with error less than 10%. And, the maximum error on total capacitance is just 7.7%. The CNN-Cap runs **4,693**× faster than 2-D field solver on a GPU server, while it consumes negligible memory compared to the look-up-table-based capacitance model. The speedup of 3-D CNN-Cap over 3-D field solver Raphael is more than **12,000**×. Even considering that the field solver can run parallely on a multi-core machine, our experiments show that the proposed CNN-Cap is at least **several hundred times faster**, with little sacrifice on accuracy.

The rest of this article is organized as follows: The background of full-chip capacitance extraction based on pattern matching and DNN-based capacitance modeling are introduced in Section 2. In Section 3, we propose the techniques of CNN-Cap for both 2-D and 3-D structures. Then, numerical experiment results are presented in Section 4. Finally, we draw the conclusion. Some preliminary results of this article were presented in Reference [21]. We extend it with the consideration of 3-D structures and more technical details and experimental results.

2 BACKGROUND

2.1 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

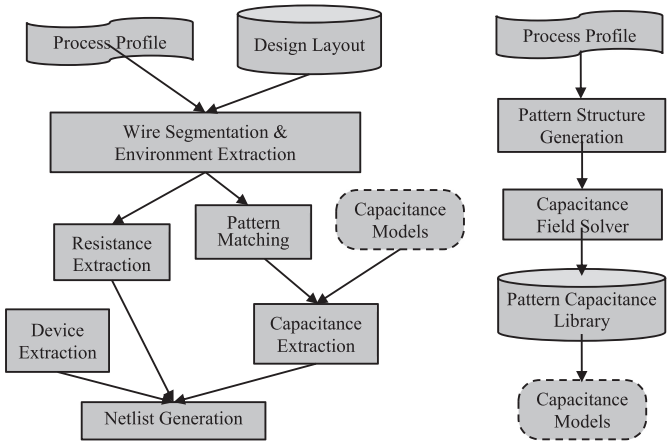


Fig. 1. The pattern matching-based full-chip parasitic extraction [24].

matching-based extraction methodology includes three major modules: (1) pattern generation, (2) capacitance model building, and (3) layout capacitance extraction [11, 14, 25]. It is illustrated as Figure 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 that considers 3-D geometric effects on capacitance with 2-D interconnect patterns through sweeping 3-D geometry of interconnects in two perpendicular directions [4, 26]. It is adopted by the OpenROAD project for parasitic extraction [16]. During the layout capacitance extraction, extraction windows are generated 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 Figure 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 Figure 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, then 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

$$C_A = C_{1f1} + C_{1o} + C_{1f2}, \tag{1}$$

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

$$C_B = C_{2f1} + C_{2o} + C_{2f2}, \tag{2}$$

for the view along B. Then,

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

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

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 Figure 3. Pattern-A, shown in Figure 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

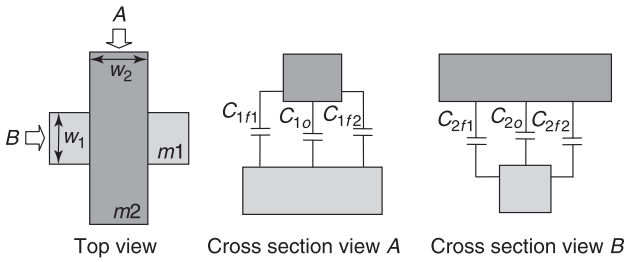


Fig. 2. 2.5-D capacitance calculation for a crossover structure [26].

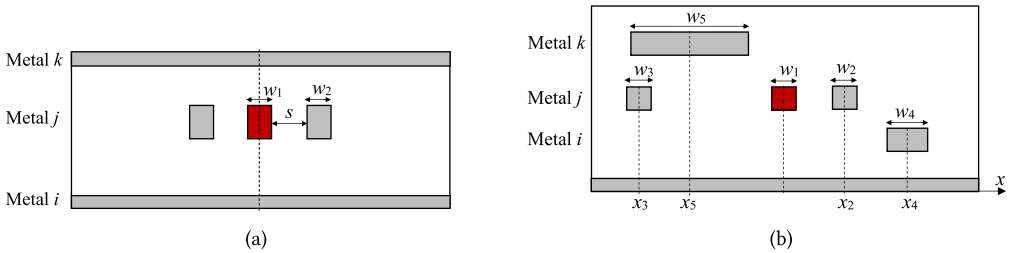


Fig. 3. Two typical 2-D interconnect structure patterns (multi-dielectric environment is not drawn). (a) Pattern-A: "sandwich" structure with three parallel wires. (b) Pattern-B: three metal layers with a fixed number of conductors.

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 Figure 3, the multi-dielectric environment is considered in these structures.

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 neighbor 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.

For revamping the accuracy drawback of 2.5-D extraction method, one measure is to allow invoking 3-D field solver for some extraction window with complicated structure. This does improve the accuracy to some extent, but causes great increase of runtime due to the excessive computational time of 3-D capacitance extraction.

2.2 Neural Network-based Capacitance Extraction

The neural network with multiple hidden layers of neurons is often referred to as **deep neural network (DNN)**. Let $\boldsymbol{x} \in \mathbb{R}^n$ be the input data and $\boldsymbol{y} \in \mathbb{R}^m$ be the output data, then, the DNN can be viewed as a function $f(\boldsymbol{x};\boldsymbol{\theta}) : \mathbb{R}^n \rightarrow \mathbb{R}^m$ 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 Figure 4. During the training, a large amount of input data along with the corresponding outputs (called *labels*) are fed to the network. The network parameters $\boldsymbol{\theta}$ are optimized to minimize the difference between $f(\boldsymbol{x};\boldsymbol{\theta})$ and the corresponding labels. When the difference is sufficiently

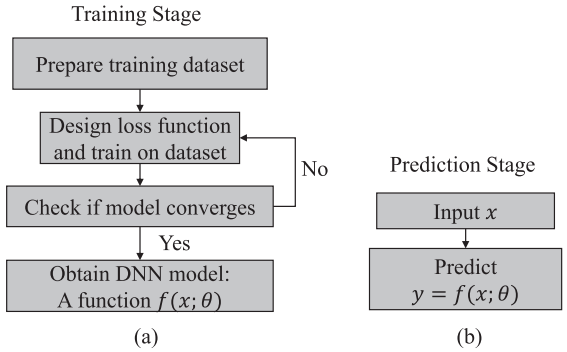


Fig. 4. The work flow of a DNN: (a) training stage, (b) prediction stage.

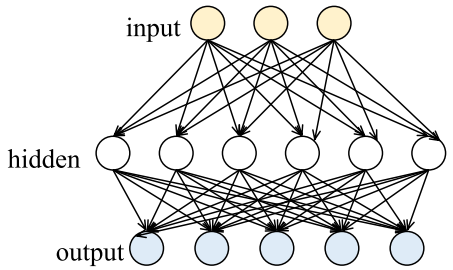


Fig. 5. An example of MLP network. The yellow, white, and blue circles stand for neurons in the input, hidden, and output layers.

small, the network is trained to become a good regressor, which can be used to do prediction (see Figure 4(b)).

The MLP neural network is a simple yet popular neural network, as shown in Figure 5 where only one hidden layer is drawn. Suppose there are n_l hidden layers. Let $\boldsymbol{h}^{(i)}$ be the variables residing at ith hidden layer's neurons. Those on the input layer and output layer can be denoted by $\boldsymbol{h}^{(0)}$ and $\boldsymbol{h}^{(n_l+1)}$, 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 ith layer, it means $\boldsymbol{h}^{(i)} = g_i(\boldsymbol{h}^{(i-1)})$ 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.

The MLP network has been studied in capacitance extraction of several 3-D structures [9]. 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 Figure 3(a)). For the example of Pattern-B (see Figure 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 $\boldsymbol{x}$ is just a vector including these geometry parameters, and the output $\boldsymbol{y}$ 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**

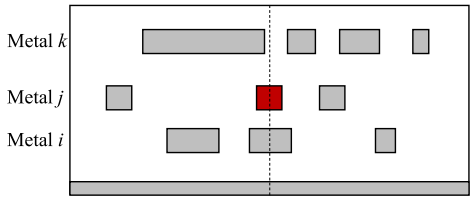


Fig. 6. A 2-D structure of three metal layers with a variable number of conductors (multi-dielectric environment not drawn). The structure is referred to as Pattern-C.

error (MSE) is employed as the loss function for minimization:

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

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

3 BUILDING CNN-CAP MODELS FOR 2-D AND 3-D INTERCONNECT CAPACITANCE EXTRACTION

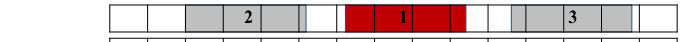
In this section, we first propose the idea of leveraging CNN to improve the capacitance modeling of structure patterns. Then, we present a grid-based data representation, the CNN architecture, and the training approach in details. Finally, we present the data generation approach and other details.

3.1 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.

A 2-D pattern of cross-section view is illustrated by Figure 6. Its structure is more general than that of Pattern-B in Section 2.1 The master conductor is still at the center of the middle layer. The structure in Figure 6 is referred to as Pattern-C. Although the conductors can lie on more than three metal layers, we assume they are just on three metal layers (with indexes i, j, and k) in the rest of this article unless explicitly stated. The proposed method can be easily extended to the patterns with more than three metal layers.

As a remedy for the inaccuracy of the 2.5-D extraction method, it is desirable to have a DNN model well predicting the capacitances of 3-D interconnect structures. It is expected to run much faster than 3-D field solver for the structure within extraction window. We consider this 3-D structure can include a variable number of conductors, like Pattern-C, and also assume it includes conductors lying on three metal layers (where the master conductor is in the middle layer). It is more straightforward to view this 3-D interconnect structure as an image.



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 2-D 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.

3.2 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 [20], we propose to evenly divide the width of extraction window into L grid cells. Thus, the conductor placement of a metal layer in 2-D structure 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 Figure 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.

This grid-based representation cannot identify the master conductor. So, we must encode more information 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: $\boldsymbol{d} \in \mathbb{R}^L$. We modify it to obtain an L-dimensional vector $\boldsymbol{x}$ for a sub-problem of calculating total capacitance. The scheme of the modification is that, if the master conductor covers the ith grid, then we set $x_i = d_i + 1$. The obtained feature vector $\boldsymbol{x}$ is shown as the third row in Figure 7. For calculating the coupling capacitance between the master and an environmental conductor, a unique feature vector is also generated by modifying $\boldsymbol{d}$. Besides adding 1 to the elements corresponding the cells overlapped with the master, we identify the environmental conductor with the following operation: For each grid i that the environmental conductor covers, we set $x_i = -d_i$. Now, for calculating different coupling capacitance there is a different feature vector for data representation. In Figure 7, the last two rows show the feature vectors for the sub-problems of calculating coupling capacitances C_{12} and C_{13}, respectively. Notice that the data representation clearly indicates which capacitance is calculated, but the scheme of modifying density vector d is not unique. For example, we can set $x_i = d_i + 2$ for master conductor and $x_i = d_i + 1$ for environmental conductor, which results in similar performance of the trained CNN model in our experiments.

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 train 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 Figure 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.

This data representation for 2-D pattern is naturally extendable to 3-D structures. The difference is that for each metal layer the data is an $L \times L$ 2-D grid, instead of 1-D grid. The approach for

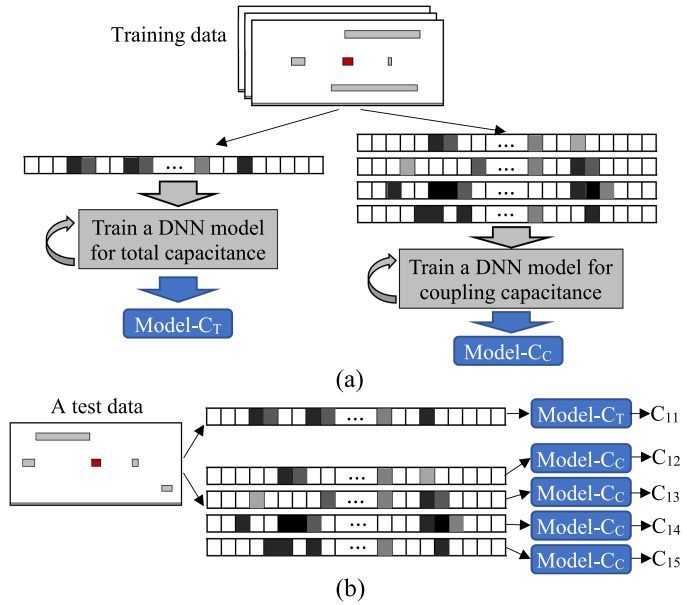


Fig. 8. The proposed method for building DNN-based capacitance models. (a) The training stage. (b) The prediction stage. Here, 2-D structure is considered as an example.

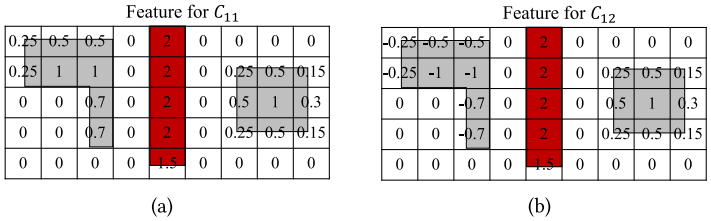


Fig. 9. An example of a metal layer's data representation for 3-D interconnect structure. The case includes three conductors where the master conductor is in the middle. (a) The data representation for calculating the total capacitance C_{11}. (b) The data representation for calculating the coupling capacitance C_{12} (Conductor 2 is on the left).

identifying master conductor and environmental conductor need not change. Figure 9 shows an example of a metal layer's data representation for 3-D interconnect structure. And, the idea of building two different models for predicting total capacitance and coupling capacitances, respectively, is also inherited. If we assume the structure includes three metal layers, then the data representation is similar to an image with three color channels. This data representation combines the spatial information and the indication of which capacitance is calculated and is therefore very compact. So, it is possible that a small-size DNN could perform well enough in the inference with these small-dimensional data. This would enable the high efficiency of the proposed CNN-Cap models.

3.3 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-based models (called CNN-Cap) for predicting the capacitances of interconnect structures.

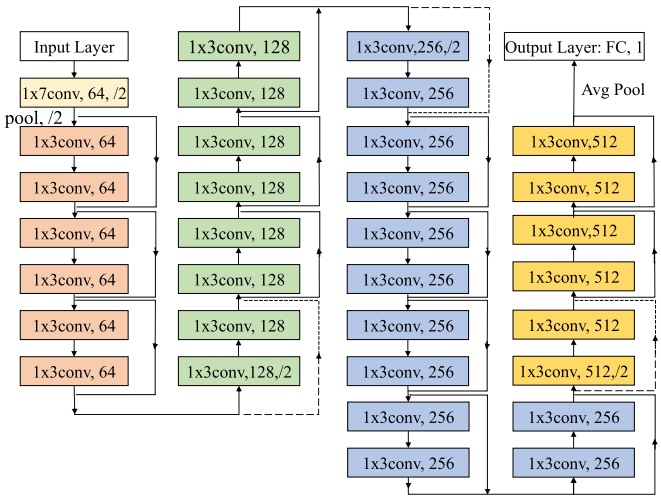


Fig. 10. The architecture of proposed CNN-Cap for 2-D structures.

Either for 2-D or 3-D structure, there are two models with same architecture used for total capacitance and coupling capacitance, respectively.

The architecture of CNN-Cap is derived from the famous ResNet architecture [7] and that for 2-D structure is shown in Figure 10. For the 2-D CNN-Cap, the input data (pattern structure with extraction information) is vectors. So, we just use 1-D convolutional layer (shown as colored block in Figure 10). 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 a 1-D average pooling layer with pooling size 32. Finally, the fixed-length vector will be fed to the **fully connected (FC)** layer with an output dimension of one to predict capacitance. In Figure 10, 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 with pooling size 2 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 $\boldsymbol{h}_1$ and a deeper vector $\boldsymbol{h}_2$ as input and takes $\mathcal{F}(\boldsymbol{h}_1) + \boldsymbol{h}_2$ 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. The shortcut connection in CNN-Cap architecture preserves the low-level features all the time, which makes training easier [7]. The architecture of CNN-Cap for 3-D capacitance extraction follows the original ResNet, except that the output is just a single number instead of a vector. It is briefly drawn as Figure 11. Specifically, a ResNet model consists of a 7×7 convolutional layer and four stages. In each stage, there are several blocks stacked. The first block halves the input by a pooling layer or a down-sampling layer, while the other blocks output a tensor of the same shape as its input. As an example, Figure 11 shows the details of the last block in ResNet-50 [7], including three **convolutional (conv)** layers with their kernel size and number of output channels labelled, three **batch normalization (BN)** layers and three ReLU layers. For the details of other stages and blocks, please refer to Reference [7].

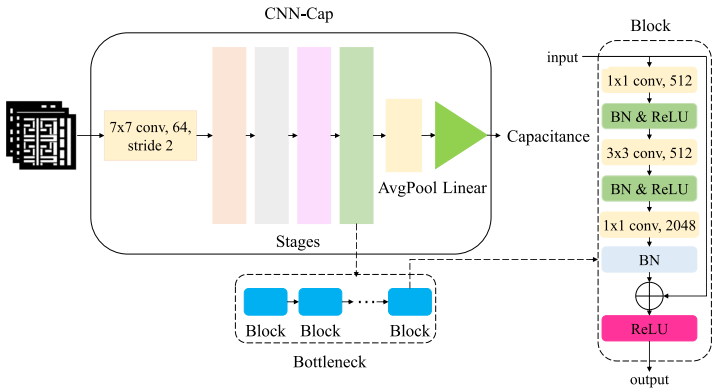


Fig. 11. The architecture of proposed CNN-Cap for 3-D structures.

Thanks to the modified and advanced architectural modules, 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, which include 18, 34, and 50 convolutional layers, respectively [7]. 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 employed in our experiments, if not explicitly stated. For training CNN-Cap, the **stochastic gradient descent (SGD)** optimizer and the Adam optimizer [10] 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:

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

where N is the number of training data, $\boldsymbol{x}^{(i)}$ indicates the ith input data, and $\boldsymbol{y}^{(i)}$ is the corresponding label. The division in Equation (5) stands for an element-wise division operation. This MSRE loss function includes the relative error, so 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 MSRE is better than MSE for the task of training the Model-C_C for coupling capacitances (see Figure 8).

3.4 Dataset Generation and Other Discussion

In this work, a data is a 2-D or 3-D interconnect structure with the proposed data representation, and the corresponding label includes the capacitances computed with a field solver. For a given process technology and a specified triplet (i, j, k) of metal layer indices, we generate a set of data for training the CNN-Cap models. A different set of data is employed to validate the performance of the trained models. The data generation schemes for 2-D and 3-D extraction tasks are as follows:

For the 2-D extraction, we 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

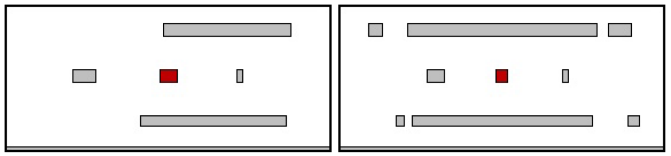


Fig. 12. Two randomly generated data for: (a) Pattern-B and (b) Pattern-C. The multi-dielectric environment is not depicted.

master conductor be a smaller value with larger probability and not exceed 10 times the minimum width. Specifically, given the minimum width of metal wire $\tilde{w}_{min}$, the width of master conductor w is sampled with 5% probability from a uniform distribution $U(\tilde{w}_{min}, 10\tilde{w}_{min})$ and 95% probability from a categorical distribution. There are 10 outcomes in the categorical distribution, namely, $\{\tilde{w}_{min}, 2\tilde{w}_{min}, \ldots, 10\tilde{w}_{min}\}$. And, the probability $P(w = \eta\tilde{w}_{min})$ is $\frac{\exp(-\lambda\eta)}{\sum_{j=1}^{10}\exp(-\lambda j)}$, which means the probability decays exponentially with the variable w. The parameter λ is set to 0.5 in our experiments. As for the environmental conductors, we first randomly determine the number of environmental conductors in each metal layer. The widths and positions of environmental conductors in a layer are sequentially sampled from uniform distributions, and their sampling interval is calculated by Algorithm 1 to ensure that the remaining environmental conductors can be legally placed.

ALGORITHM 1: Generating a layer of environmental conductors for a 2-D pattern

Input: Number of environmental conductors n, the minimum width w_{min}, the minimum spacing s_{min}, width of the pattern l.
Output: n generated environmental conductors represented by their horizontal coordinates.

1: $x_1^- \sim U(0, l-(n-1)s_{min}-nw_{min})$ ▷ $x_1^- \geq 0, x_1^- + nw_{min} + (n-1)s_{min} \leq l$
2: $x_1^+ \sim U(x_1^- + w_{min}, l-(n-1)(s_{min}+w_{min}))$ ▷ $x_1^+ - x_1^- \geq w_{min}, w_1^+ + (n-1)s_{min} + (n-1)w_{min} \leq l$
3: **for** $i \leftarrow 2$ to n **do**
4: $x_i^- \sim U(x_{i-1}^+ + s_{min}, l-(n-i)s_{min}-(n-i+1)w_{min})$ ▷ $x_i^- - x_{i-1}^+ \geq s_{min}$. The upper bound is similar to x_1^-.
5: $x_i^+ \sim U(x_i^- + w_{min}, l-(n-i)(s_{min}+w_{min}))$ ▷ Similar to x_1^+.
6: **end for**
7: **return** $[x_1^-, x_1^+], \ldots, [x_n^-, x_n^+]$

Similarly, the random data can be generated for the conventional patterns, such as 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. Figure 12 shows the geometries of the conductors in two data generated with our approach. Figure 12(a) stands for a data for Pattern-B and Figure 12(b) is for Pattern-C.

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 form a long vector and then input it to the MLP network. However, this hybrid model cannot have high accuracy, because the spatial

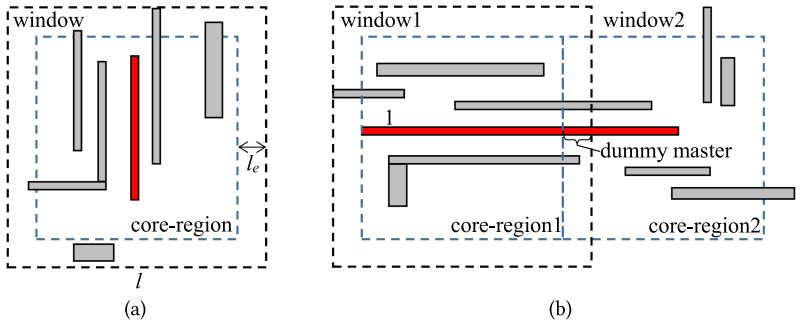


Fig. 13. Top view of extraction window (only showing the metal layer including the master conductor). (a) The case with a small-size master conductor within the core region. (b) The case with a large-size master conductor exceeding the window.

information in the grid-based representation is lost. The comparison experiments in Section 4.2 validate this.

For 3-D extraction, a data is a 3-D window structure including interconnect wires on three metal layers. Considering the actual usage of CNN-Cap model in the scenario of full-net/full-chip extraction, we can assume the master conductor resides on the layer in the middle. The conductor settings in the 3-D structures should also facilitate the assembly of capacitances extracted from different windows, while the structures should cover as more as possible topologies in real ICs.

Considering the existence of long interconnect wires, we see that the master conductor can approach to the boundary of window. Directly computing the capacitances between the master and environmental conductors in the window may cause large error due to the discrepancy between the electrical field under the setting of extraction window and that in real circumstance. To resolve this issue, we propose a concept of *core-region* within the window. As shown in Figure 13, the core-region is formed with shrinking the window by a distance l_e. If master conductor is within the core-region (see Figure 13(a)), then the computed coupling capacitance can have sufficient accuracy, since only the faraway part (that out of the window) of environmental conductor is ignored. Otherwise, like that shown in Figure 13(b), we should cut the part outside the core-region (called *dummy master* here) off the master conductor and regard the dummy master as an environmental conductor during the extraction computation. Take the structure in Figure 13(b) as an example. The remaining part of conductor 1 that is not master conductor in the extraction of window1 will be set as the master conductor in the core-region of next window, i.e., core-region2 in window2 (contour not drawn). With this treatment, the capacitance results from different extraction windows can be combined to deliver accurate capacitance results. Therefore, a data for 3-D extraction should be a window structure where the master conductor therein may be cut by the core-region. The touching of master conductor and dummy master (i.e., environmental conductor) cause singularity of electric field and may bring difficulty in capacitance modeling.

The sample structures for 3-D extraction can be generated randomly or from real IC design. Due to more geometry parameters, the randomly generated 3-D structures may not reflect or cover various actual situations. So, if there are multiple real designs under same process technology, then generating the training data for 3-D structure by chopping their layout can be an easy and good choice.

4 EXPERIMENTAL RESULTS

The proposed CNN-Cap and other DNN models for comparison are implemented with PyTorch. For the structures in datasets, we obtain the capacitance results (as labels of data) with the

Table 1. The Geometric Parameters of the Metal Layers in the Two Tested Process Technologies (in Unit of μm)

Layer index	55 nm Tech.			15 nm 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

golden-standard capacitance solver Raphael [8]. All experiments are carried out on a Linux server with two Intel Xeon Silver 4214 CPUs at 2.2 GHz and eight Nvidia RTX2080Ti GPUs.

2-D pattern structures are generated following two process technologies. One is a 55 nm process technology from industry, while the other is the 15 nm process technology in FreePDK15 [2, 5]. 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. The geometric parameters for the metal layers in the two process technologies are listed in Table 1.

For the two technologies, we randomly choose some three-metal-layer combinations to generate the 2-D patterns. For each pattern, we generated 50,000 sample structures with the approach in Section 3.4 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 1,024, 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 55 nm 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 2-D CNN-Cap on Pattern-B structures and then evaluate its performance on Pattern-C structures. After that, we present the experimental results on 3-D capacitance extraction. The comparisons on runtime and model size are studied last.

4.1 Results for 2-D Pattern-B Structures

For Pattern-B structures, the MLP-based model presented in Section 2.2 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

Table 2. DNN Models' Performance on Total Capacitance for 2-D Pattern-B Structures

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

Table 3. DNN Models' Performance on Coupling Capacitance for 2-D Pattern-B Structures

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

functions, ReLU, Sigmoid, and Tanh, are tested. The results show that Tanh function consistently surpasses 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 of MSE (4) and MSRE (5) are tested with MLP-Cap. The results show that the MSE loss function makes MLP-Cap perform better.

In this experiment, the pattern is the same as that in Figure 3(b) which includes 1, 3, and 1 conductors on three metal layers, respectively. And, the structure can be described with nine geometric parameters, which are the input to MLP-Cap. The results for predicting total capacitance and coupling capacitance, for four datasets from the both technologies, are shown in Tables 2 and 3, respectively. In the tables, every two rows correspond to a dataset generated for a pattern. The column "Layers" contains the triplet (i, j, k), representing the indices of three metal layers. 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 2 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 3 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 detail, 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%.

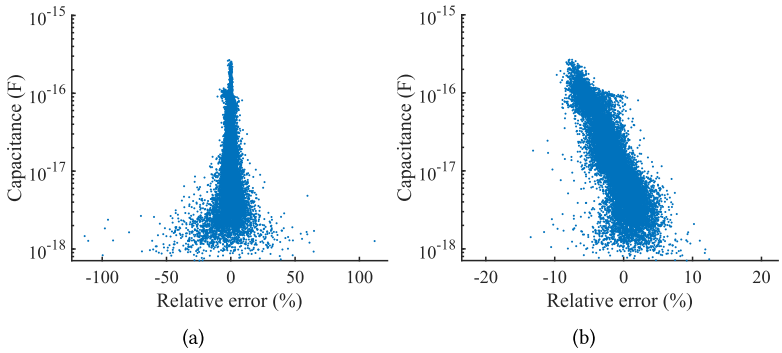


Fig. 14. The calculated coupling capacitance versus relative error for 2-D Pattern-B structures. Layer combination (2, 3, 6) layer combination in the 55 nm technology. (a) Results of MLP-Cap. (b) Results of CNN-Cap.

Figure 14 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 correspond 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 Figure 14(a).

In addition to those in Tables 2 and 3, the dataset for layer combination (2, 5, 7) under the 15 nm technology has 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.

Another capacitance modeling approach that 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, then 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.

4.2 Results for 2-D 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 in Section 3.4 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 five datasets of Pattern-C are listed in Tables 4 and 5. 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 **1.3%**. 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. 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.

Figure 15 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

Table 4. DNN Models' Performance on Total Capacitance for 2-D Pattern-C Structures

Tech. Node	Layers	Method	Err_{avg}	Err_{max}	Ratio(Err>5%)
55 nm	(2, 3, 6)	Grid+MLP	0.4%	7.5%	0.1%
		CNN-Cap	0.1%	1.0%	0
55 nm	(2, 4, 6)	Grid+MLP	0.7%	7.7%	0.1%
		CNN-Cap	0.2%	1.1%	0
15 nm	(1, 3, 5)	Grid+MLP	0.7%	4.3%	0
		CNN-Cap	0.2%	1.1%	0
15 nm	(1, 3, 8)	Grid+MLP	0.5%	5.4%	0.1%
		CNN-Cap	0.2%	1.2%	0
15 nm	(2, 5, 7)	CNN-Cap	0.2%	1.3%	0

Table 5. DNN Models' Performance on Coupling Capacitance for 2-D Pattern-C Structures

Tech. Node	Layers	Method	Err_{avg}	Err_{max}	Ratio(Err>10%)
55 nm	(2, 3, 6)	Grid+MLP	10.4%	542.4%	27.1%
		CNN-Cap	1.2%	38.6%	0.3%
55 nm	(2, 4, 6)	Grid+MLP	9.1%	489.3%	22.7%
		CNN-Cap	1.2%	14.4%	0.1%
15 nm	(1, 3, 5)	Grid+MLP	9.8%	492.8%	24.8%
		CNN-Cap	1.8%	38.8%	0.4%
15 nm	(1, 3, 8)	Grid+MLP	11.4%	390.4%	30.8%
		CNN-Cap	1.5%	12.4%	0.0%
15 nm	(2, 5, 7)	CNN-Cap	1.2%	15.3%	0.0%

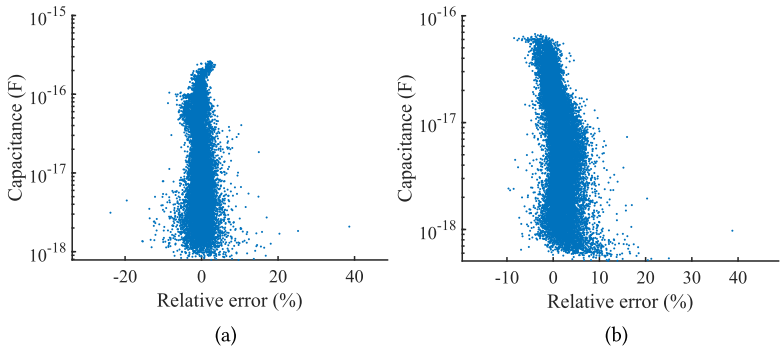


Fig. 15. The calculated coupling capacitance versus relative error for 2-D Pattern-C structures. (a) Results of CNN-Cap for layer combination (2, 3, 6) in the 55 nm technology. (b) Results of CNN-Cap for layer combination (1, 3, 5) in the 15 nm technology.

capacitance for most structures. For example, for the test data corresponding to layer combination (2, 3, 6) in the 55 nm technology, only 0.3% of all coupling capacitances has error larger than 10%. And, the large error always occurs on the very small coupling capacitances.

To demonstrate that CNN-Cap performs well not only when the conductors lie on three metal layers, an experiment is first conducted on the Pattern-C dataset for layer combination (1, 3) under the 15 nm technology. To apply CNN-Cap to this scenario, the number of channels modified to 2. The experimental results show that the average relative error of CNN-Cap on total capacitance and

Table 6. The Total-capacitance Errors of 2-D CNN-Cap Trained on Different Datasets for Layer Combination (1, 3, 5)

Training Dataset	Testing on Pattern-B			Testing on Pattern-C		
	Err_{avg}	Err_{max}	Ratio(Err>5%)	Err_{avg}	Err_{max}	Ratio(Err>5%)
Whole Pattern-C	0.3%	4.8%	0	0.2%	1.1%	0
Pattern-B$_{half}$+Pattern-C$_{half}$	0.2%	1.0%	0	0.1%	1.0%	0

Table 7. The Coupling-capacitance Errors of 2-D CNN-Cap Trained on Different Datasets for Layer Combination (1, 3, 5)

Training Dataset	Testing on Pattern-B			Testing on Pattern-C		
	Err_{avg}	Err_{max}	Ratio(Err>10%)	Err_{avg}	Err_{max}	Ratio(Err>10%)
Whole Pattern-C	2.5%	46.4%	2.1%	1.8%	38.8%	0.4%
Pattern-B$_{half}$+Pattern-C$_{half}$	2.0%	13.3%	0.1%	2.0%	26.0%	0.5%

coupling capacitance are 0.2% and 1.9%, respectively, and the maximum relative error of CNN-Cap on total capacitance and coupling capacitance are 1.9% and 9.3%, respectively. Then, we conduct an experiment on the Pattern-C dataset for layer combination (1, 3, 5, 7) under the 15 nm technology. The results show that the average relative error on total capacitance and coupling capacitance are 0.3% and 1.9%, respectively, and the maximum relative error on total capacitance and coupling capacitance are 1.4% and 22.9%, respectively. There are only 0.5% of coupling capacitances with error larger than 10%. The master conductor resides on layer 3 in these two experiments.

To demonstrate the generalization ability of CNN-Cap, we show the results of training and testing on different datasets in Tables 6 and 7. The previous datasets for layer combination (1, 3, 5) under the 15 nm technology are used. We first test the performance of the CNN-Cap trained with original Pattern-C training subset on the Pattern-B test cases. Then, we randomly select half of the training data of Pattern-B and Pattern-C cases and combine them to obtain a training subset of the same size. With it, we train a CNN-Cap model and test its performance on the Pattern-B and Pattern-C test cases. From the results in Tables 6 and 7, we see that the model trained on Pattern-C data can generalize to the unseen data (Pattern-B) to a certain extent, i.e., the average relative errors of total capacitance and coupling capacitance are just 0.3% and 2.5%, respectively. This is because Pattern-C is a more complicated than Pattern-B. However, there are 2.1% of all coupling capacitances that have relative error larger than 10%. This is because Pattern-B and Pattern-C are two disjoint datasets. Specifically, the total number of conductors in the up and down metal layers is 2 in Pattern-B structure, while it ranges from 6 to 8 in Pattern-C structure. On the contrary, with the combined training subset the performance of CNN-Cap is similarly well as the results in Tables 2 and 3. And, it performs much better for predicting the capacitances of Pattern-B structures. This demonstrates how our method will be deployed in a more general scenario.

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 Figure 16. From it, we see that, when the ratio of training subset is 50% or larger (meaning with 25,000 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 25,000 data through 45,000 data is usually sufficient for building an effective CNN-Cap model.

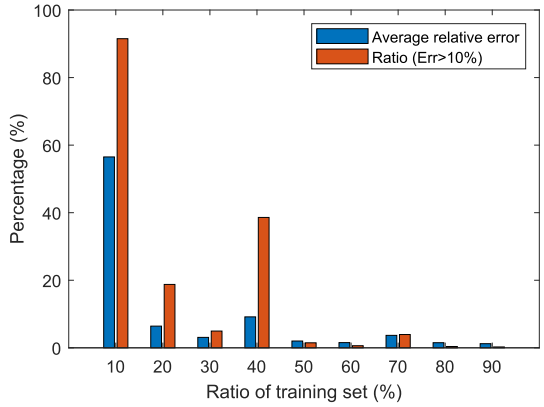


Fig. 16. The accuracy of 2-D CNN-Cap on coupling capacitance vs. the size of training subset.

Table 8. CNN-Cap's Performance on Total Capacitance and Coupling Capacitances for the 3-D Structure with (1, 2, 3) Metal-Layer

Architecture	Error on Total Cap.			Error on Coupling Cap.		
	Err_{avg}	Err_{max}	Ratio(Err>5%)	Err_{avg}	Err_{max}	Ratio(Err>10%)
ResNet-34	1.1%	16%	1.3%	**3.1%**	**44%**	**4.1%**
ResNet-50	**1.1%**	**7.7%**	**1.0%**	3.4%	63%	4.2%

4.3 Results for 3-D Structures

With a real SRAM design whose layout size is about $165\mu m \times 188\mu m$, we collect 3-D interconnect structures by chopping the layout with a $5\mu m \times 5\mu m$ window. The distance l_e determining the core-region (see Figure 13(a)) in each window is set $0.5\mu m$, which is large enough to make accurate result of capacitance extraction. We have obtained 8,685 such window structures, each of which derives a data after specifying a (i, j, k) metal-layer combination and a master conductor. Notice in each window structure there is possible dummy master touching the master conductor (see Figure 13(b)).

For the metal-layer combination (1, 2, 3) in the SRAM design, we build a dataset with 13,579 sample structures for extraction after specifying the master conductor. It is then randomly split into a training subset (with 90% of the samples) and a testing subset (with 10% of the samples). The number of grid cells along one direction (L) is set to 200, which ensures that the grid size (equal to 25 nm) is less than the half of minimum wire spacing. With the training subset the 3-D CNN-Cap models are trained. The hyper-parameters include batch size set to 32, learning rate set to 10^{-4}. The inference results on the testing subset are listed in Table 2 for the model derived from ResNet-34 and ResNet-50, respectively.

From Table 8, we see that most of total capacitances have error within 5%; only for 1% of testing data the error of total capacitance can be larger than 5%, with the maximum just 7.7%. For the coupling capacitance, most cases have error within 10%, and only about 4% of tested data violates this criterion. Comparing the two architectures for CNN-Cap, we see that the both have similar performance and that ResNet-50 may derive better accuracy on total capacitance with smaller maximum error. In Figure 17, we show the plots of capacitance result versus relative error. They demonstrate the accuracy of the proposed CNN-Cap for 3-D capacitance extraction.

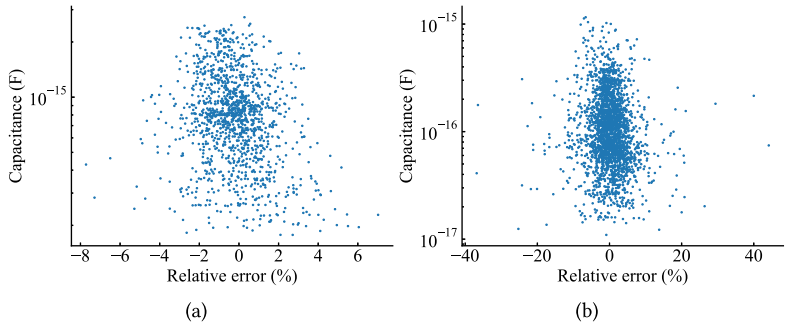


Fig. 17. The capacitance result of CNN-Cap versus relative error for 3-D structures. (a) Results of total capacitance. (b) Results of coupling capacitance.

Table 9. The Average Runtimes of CNN-Cap, Raphael, and RWCap for Calculating the Capacitances of a 2-D or 3-D Structure

Method	2-D Structure			3-D Structure				
	Raphael	CNN-Cap	Sp.	Raphael	RWCap	CNN-Cap	Sp1	Sp2
Time (ms)	704	0.15	4,693×	95,346	1,420	7.43	12,833×	191×

Sp1 and Sp2 are the speedup ratios of CNN-Cap to Raphael and RWCap (run with 32 threads), respectively.

4.4 Comparisons on Runtime and Model Size

We compare the prediction time of 2-D CNN-Cap with the 2-D field solver `rc2` within Raphael. For a dataset including 50,000 Pattern-C structures, we extract the capacitances of them with CNN-Cap and Raphael, respectively. Their average runtimes per structure are listed in Table 6. The testing batch size is 2,048. As for 3-D extraction, 1,358 structures from the testing subset are extracted with CNN-Cap, Raphael `rc3`, and the random walk-based solver RWCap [22, 28], respectively. Raphael runs with setting the number of grids to 10^7, while RWCap runs with 32-thread parallel computing and a tolerance of 1% for the 1-σ error of total capacitance. The testing batch size for 3-D CNN-Cap is 32. Their average runtimes are also listed in Table 9. Notice that multiple inferences are carried out with CNN-Cap to produce the total capacitance and coupling capacitances for a sample structure.

From the table, we see that 2-D CNN-Cap runs **4,693**× faster than Raphael `rc2`, while 3-D CNN-Cap runs more than **10,000**× faster than Raphael `rc3` and **191**× faster than the parallel RWCap. Even though we can run multiple processes of Raphael on the machine, say, 32 processes, calculating capacitance with CNN-Cap can still be **several hundred times** faster than running the conventional field solvers.

As for the model size, a 2-D CNN-Cap model has 14,473,418 parameters occupying 55.2 MB storage, which is much smaller than the look-up-table-based model. For the model size of 3-D CNN-Cap, it has 23,510,081 parameters occupying 89.7 MB storage.

The training time for a 2-D 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 eight Nvidia RTX2080Ti GPUs. And, the CNN-Cap models deliver much better accuracy than the existing methods for pattern capacitance modeling. As for the training of 3-D CNN-Cap model, in our experiment setting, its time cost is not larger than five hours. Therefore, the time for building the 3-D CNN-Cap for a process technology is also affordable.

5 CONCLUSIONS AND DISCUSSIONS

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 and 3-D interconnect 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 and 3-D structure with a variable number of conductors. This largely reduces the number of 2-D patterns and the corresponding capacitance models or improves the accuracy of the pattern matching-based extraction methodology. Extensive experiments have demonstrated the accuracy and efficiency of the CNN-Cap model and its advantages over MLP-based models and traditional model-building approaches.

The proposed grid-based representation in CNN-Cap cannot unambiguously handle non-Manhattan conductor shapes, and the proposed method is customized for a process technology and requires retraining when deployed to new process technology. In the future, extending the CNN-Cap models to consider non-Manhattan conductor shapes, the process-variation effects, and making it independent from process technology will be explored.

REFERENCES

[1] Mohamed Saleh Abouelyazid, Sherif Hammouda, and Yehea Ismail. 2022. A fast and accurate middle end of line parasitic capacitance extraction for MOSFET and FinFET technologies using machine learning. In *27th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 371–376.

[2] Kirti Bhanushali and Rhett W. Davis. 2015. FreePDK15: An open-source predictive process design kit for 15nm FinFET technology. In *Symposium on International Symposium on Physical Design*. 165–170.

[3] U. Choudhury and A. Sangiovanni-Vincentelli. 1995. Automatic generation of analytical models for interconnect capacitances. *IEEE Trans. Comput.-aid. Des. Integ. Circ. Syst.* 14, 4 (1995), 470–480.

[4] J. Cong, L. He, A. Kahng, D. Noice, N. Shirali, and S. Yen. 1997. Analysis and justification of a simple, practical $2^{1/2}$-D capacitance extraction methodology. In *Design Automation Conference*. 627–632.

[5] NCSU EDA group. 2020. FreePDK15. Retrieved from https://eda.ncsu.edu/freepdk15/.

[6] Weibing Gong, Wenjian Yu, Yongqiang Lü, Qiming Tang, Qiang Zhou, and Yici Cai. 2010. A parasitic extraction method of VLSI interconnects for pre-route timing analysis. In *International Conference on Communications, Circuits and Systems (ICCCAS)*. 871–875.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

[8] Synopsys Inc. 2021. TCAD-Raphael. Retrieved from http://www.synopsys.com/silicon/tcad/interconnect-simulation/raphael.html.

[9] R. Kasai, T. Kanamoto, M. Imai, A. Kurokawa, and K. Hachiya. 2019. Neural network-based 3D IC interconnect capacitance extraction. In *International Conference on Communication Engineering and Technology (ICCET)*. 168–172.

[10] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

[11] Luciano Lavagno, Louis Scheffer, and Grant Martin. 2006. *EDA for IC Implementation, Circuit Design, and Process Technology*. CRC Press.

[12] Y. Le Coz and R. Iverson. 1992. A stochastic algorithm for high speed capacitance extraction in integrated circuits. *Solid-state Electron.* 35, 7 (1992), 1005–1012.

[13] Y. LeCun, Y. Bengio, and G. Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

[14] Z. Li and W. Shi. 2020. Layout capacitance extraction using automatic pre-characterization and machine learning. In *International Symposium on Quality Electronic Design*. 457–464.

[15] K. Nabors and J. White. 1991. FastCap: A multipole accelerated 3-D capacitance extraction program. *IEEE Trans. Comput.-aid. Des. Integ. Circ. Syst.* 10, 11 (1991), 1447–1459.

[16] The-OpenROAD-Project. 2021. OpenRCX. Retrieved from https://github.com/The-OpenROAD-Project/OpenRCX.

[17] Padmanava Sen, Wayne H. Woods, Saikat Sarkar, Rana J. Pratap, Brian M. Dufrene, Rajarshi Mukhopadhyay, Chang-Ho Lee, Essam F. Mina, and Joy Laskar. 2006. Neural-network-based parasitic modeling and extraction verification for RF/millimeter-wave integrated circuit design. *IEEE Trans. Microw. Theor. Techniq.* 54, 6 (2006), 2604–2614.

[18] Brett Shook, Prateek Bhansali, Chandramouli Kashyap, Chirayu Amin, and Siddhartha Joshi. 2020. MLParest: Machine learning based parasitic estimation for custom circuit design. In *Design Automation Conference*. 1–6.
[19] Xiren Wang, Deyan Liu, Wenjian Yu, and Zeyi Wang. 2005. Improved boundary element method for fast 3-D interconnect resistance extraction. *IEICE Trans. Electron.* 88, 2 (2005), 232–240.
[20] Wan-Yu Wen, Jin-Cheng Li, Sheng-Yuan Lin, Jing-Yi Chen, and Shih-Chieh Chang. 2014. A fuzzy-matching model with grid reduction for lithography hotspot detection. *IEEE Trans. Comput.-aid. Des. Integ. Circ. Syst.* 33, 11 (2014), 1671–1680.
[21] Dingcheng Yang, Wenjian Yu, Yuanbo Guo, and Wenjie Liang. 2021. CNN-Cap: Effective convolutional neural network based capacitance models for full-chip parasitic extraction. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 1–9.
[22] Ming Yang and Wenjian Yu. 2020. Floating random walk capacitance solver tackling conformal dielectric with on-the-fly sampling on eight-octant transition cubes. *IEEE Trans. Comput.-aid. Des. Integ. Circ. Syst.* 39, 12 (2020), 4935–4943.
[23] H. Yao, Y. Qin, and L. Jiang. 2016. Machine learning based mom (ML-MoM) for parasitic capacitance extractions. In *IEEE Electrical Design of Advanced Packaging and Systems Symposium (EDAPS)*. 171–173.
[24] Wenjian Yu, Mingye Song, and Ming Yang. 2021. Advancements and challenges on parasitic extraction for advanced process technologies. In *26th Asia and South Pacific Design Automation Conference (ASP-DAC)*. 841–846.
[25] W. Yu and X. Wang. 2014. *Advanced Field-solver Techniques for RC Extraction of Integrated Circuits*. Springer.
[26] Wenjian Yu and Zeyi Wang. 2005. Capacitance extraction. In *Encyclopedia of RF and Microwave Engineering*, K. Chang (Ed.). John Wiley & Sons Inc., 565–576.
[27] Wenjian Yu, Bolong Zhang, Chao Zhang, Haiquan Wang, and Luca Daniel. 2016. Utilizing macromodels in floating random walk based capacitance extraction. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 1225–1230.
[28] W. Yu, H. Zhuang, C. Zhang, G. Hu, and Z. Liu. 2013. RWCap: A floating random walk solver for 3-D capacitance extraction of very-large-scale integration interconnects. *IEEE Trans. Comput.-aid. Des. Integ. Circ. Syst.* 32, 3 (2013), 353–366.

Received 3 May 2022; revised 15 September 2022; accepted 23 September 2022