Received 7 September 2023, accepted 7 October 2023, date of publication 10 October 2023, date of current version 18 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3323589

METHODS

Unet-Astar: A Deep Learning-Based Fast Routing Algorithm for Unified PCB Routing

SHIYUAN YIN[1], MIN JIN[1], GANG CHEN[1], GUOLIANG GONG[1], WENYU MAO[1], AND HUAXIANG LU[1,2,3,4]

[1]High-Speed Circuit and Neural Network Laboratory, Institute of Semiconductors, Chinese Academy of Sciences (CAS), Beijing 100083, China
[2]Semiconductor Neural Network Intelligent Perception and Computing Technology Beijing Key Laboratory, Beijing 100083, China
[3]College of Microelectronics, University of Chinese Academy of Sciences, Beijing 100049, China
[4]Materials and Optoelectronics Research Center, University of Chinese Academy of Sciences, Beijing 100049, China

Corresponding author: Gang Chen (chengang08@semi.ac.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant U19A2080, and in part by the Pilot Project of Chinese Academy of Sciences under Grant XDA18040400 and Grant XDB44000000.

ABSTRACT In recent years, there has been extensive research on the routing problem of printed circuit boards (PCBs). Due to the increasing number of pins, high pin density, and unique physical constraints, manual PCB routing has become a time-consuming task to achieve design convergence. Previous work decomposed the problem into escape routing and area routing, focusing on these problems separately. However, there was always a gap between these two problems, requiring significant human effort for iterative algorithm adjustments. Furthermore, previous area routing work mainly focused on routing between ball grid array (BGA) packages in escape routing. However, in practice, many components are not in the form of BGA packages, such as passive devices, decoupling capacitors, and through-hole pin arrays. Therefore, it is necessary to study a unified routing approach. The current unified routing approach adopts the A* algorithm, but there is still room for improvement in routing speed. This paper proposes a new algorithm called Unet-Astar, which accelerates the routing efficiency by employing deep learning algorithms in a simulated environment. Additionally, a Deeper Unet is proposed for generating recommended regions for the routing algorithm. The new network structure can provide more contextual information, thereby improving routing efficiency. Experimental results demonstrate the effectiveness and efficiency of the proposed algorithm. Specifically, for all given test cases, our router achieves approximately a 70% improvement in runtime speed compared to the old router. Another major contribution of this work is the development of a routing problem set generator, which can generate parameterized routing problem sets with different sizes and constraints. This enables the evaluation of different routing algorithms and the generation of training datasets for future data-driven routing methods. All the code has been open-sourced and can be found at https://github.com/Firesuiry/Unet-Astar-For-PCB-Routing.

INDEX TERMS Physical design, printed circuit board, routing, machine learning.

I. INTRODUCTION

With the vigorous development of electronic information technology, the current integrated circuit process has entered the nanometer era. The integration level of components on printed circuit boards (PCBs) is increasing, the number of pins is growing, and the interconnections between components are becoming more complex. Consequently, one of the challenging tasks in PCB design is the routing task. However, existing automatic routing algorithms have low routing success rates and slow speeds. Currently, in industrial applications, a significant reliance on manual PCB routing

The associate editor coordinating the review of this manuscript and approving it for publication was Ludovico Minati.

by engineers leads to a substantial consumption of time and manpower resources. Therefore, there is an urgent need for an intelligent automatic routing algorithm that can be practically applied to modern large-scale electronic circuit design to enhance the design efficiency of Electronic Design Automation (EDA).

In recent years, there has been continuous research in this field. The research in this direction is mainly divided into two parts: area routing and escape routing. The process of routing the pins within a component to the component's edge is called the escape process, while the process of routing between two components is called area routing.

However, from our observation of actual PCB designs, designs typically consist of numerous components such as passive devices, decoupling capacitors, and through-hole pin arrays, which are not components in BGA packages. These non-BGA packaged components are usually irregularly distributed in the PCB design, resulting in uneven congestion distribution and making routing tasks more difficult. Therefore, it is necessary to develop a PCB routing algorithm to address such issues.

Lin et al. [1] first proposed a unified routing approach based on the A* algorithm to tackle this problem. Although this algorithm accomplishes the task, there is still significant room for improvement in routing speed. Additionally, the algorithm cannot leverage the increasingly powerful GPU computational resources available today.

Therefore, in this paper, a routing algorithm that can utilize GPU computing power to accelerate the process is designed. This is achieved by employing a neural network to predict routing regions before the actual routing. The main contributions of this paper are as follows:

In this paper, Unet-Astar, a deep-learning-based fast routing algorithm for unified PCB routing, is proposed. The contributions are as follows.

1) Propose a neural network-based routing guidance method called Unet-Astar, which improves routing efficiency by predicting routing regions before the actual routing process.
2) Introduce a novel neural network architecture called Deeper-Unet for generating recommended regions. This network structure provides additional contextual information, leading to improved routing efficiency.
3) Develop a PCB routing problem generator for model training and facilitating future research.
4) Conduct experiments that demonstrate an average reduction of approximately 70% in runtime compared to previous algorithms.

The rest of this paper is organized as follows. The definitions of the problem is presented in Section II.

Related work on PCB routing is introduced in Section III.

The implementation of the proposed Unet-Astar and Deeper Unet is described in Section IV.

Experimental studies are presented in Section V.

Conclusions summarizing the contributions of this paper are presented in Section VI.

II. BACKGROUND INFORMATION

A. PROBLEM DEFINITION

This section presents some terminology and the relevant problem formulation. A two-layer PCB layout consists of components with defined solder mask defined (SMD) pads and/or through-hole pads. Their definitions are as follows.

Definition 1 (Component): Any fundamental discrete device or physical entity within an electronic system that influences electronics or its related fields.

Definition 2 (SMD Pad/Through-Hole Pad): Pads that traverse all routing layers. It should be noted that, for the PCB routing problem, the allocation and placement of all component layers are given after the layout. To perform routing, a netlist is used to define the connections between different component pads. For ease of research, in this paper, all pads are defined as through-hole pads.

Definition 3 (Netlist): Describes the connections between through-hole pads and/or SMD pads. The routing results need to satisfy design rules for good manufacturability. Typical rules include non-crossing and spacing greater than the clearance requirements.

Based on these definitions, the PCB routing problem is defined as follows.

Problem 1 (PCB Routing): Given a netlist, design rules, and a post-placement layout containing a set of pads, connect all nets without violating the design rules.

III. RELATED WORK

The objective of the extensively studied PCB routing problem is to route all connections between the Ball Grid Array (BGA) packages of chip packaging while satisfying various physical constraints. Additionally, the use of through-holes is often restricted, making the routing planar.

This planar routing approach distinguishes PCB routing as a separate problem from IC routing. Previous research has categorized the PCB routing problem into two types: (1) Escape routing problem and (2) Area routing problem.

The escape routing problem involves routing from the pads of the BGA to their array boundary. The region routing problem is to connect the previously escaped BGAs, often subject to upper/lower bounds on the routing length for each connection.

A. ESCAPE ROUTING AND REGION ROUTING

The escape routing problem involves routing from the pads of the Ball Grid Array (BGA) to their array boundary. As the number of pins on a chip increases, this problem becomes increasingly complex.

Escape routing can be classified into three types: Unordered Escape Routing (UER), Ordered Escape Routing (OER), and Simultaneous Escape Routing (SER). UER and OER focus on determining escape paths from the pads of each component to the component boundary without assigning an escape order. Wang et al. [2] proposed a model based on triangular patterns and sequences to facilitate network flow-based methods for deriving solutions to UER. Fang et al. [3], [4], [5] also employed network flow-based algorithms to perform unordered escape routing. Yan et al. [6] presented a network model that accurately reflects routing resource capacity, resulting in improved results for unordered escape routing. Yan and Wong [7] introduced a network flow model for UER and modeled OER using Boolean satisfiability (SAT) formulas [8], [9]. The UER model is not suitable for practical applications involving designs with two or more components since the escape order between two components is often inconsistent, resulting in lower reliability of region routing. Therefore, the application of OER is limited as the escape order needs to be manually or automatically assigned. However, SER can address the challenges posed by OER and UER. Luo et al. [10], [11] proposed boundary routing as an efficient strategy for internally routing a single component by iteratively tracing feasible orders between two components. Fang et al. [12] introduced an integer linear programming-based method with reduction techniques to achieve ordered escape routing within a reasonable runtime. Kubo et al. [13] developed an iterative over-assignment method to effectively minimize congestion and total wirelength. However, these works determine the escape order only for single-layer sequences, and if not all nets can be routed within a layer, they assign given layers for each net.

The area routing problem primarily focuses on addressing the issue of achieving length matching for routes. Due to the efforts of the escape router, the input to the length matching router is typically considered to be planar.

Ozdal et al. [14] adopted a routing scheme based on Lagrangian relaxation to obtain a routing solution that achieves length matching. Additionally, Ozdal et al. [15] proposed an efficient algorithm based on river routing to solve the length matching routing problem within the channels and used theoretical constant factors for optimality. Yan et al. [16] presented an algorithm that retrieves routing solutions without any routing topology restrictions using a bounded slicing line grid. Yan et al. [17] established an obstacle-aware routing framework based on routing region partitioning and maximum flow algorithm.

In addition to the traditional escape routing and region routing problems, several works [18], [19], [20], [21], [22], [23], [24] have addressed routing problems while considering the constraints of differential pairs. Fang et al. [19], [21] proposed a chip package board code design algorithm that takes into account the constraints of differential pairs. However, their assumption of monotonic routing restricts the practical use of the algorithm. Yan et al. [20] proposed a negotiation-based routing algorithm considering routing congestion to achieve routing for differential pairs. Li et al. [22] employed a two-stage scheme and a minimum-cost maximum-flow algorithm to simultaneously route all differential pairs. Wang and Jiao et al. [23], [24] developed unordered and ordered escape routing algorithms for staggered pin arrays, taking into consideration the constraints of differential pairs.

B. MODEL-BASED EDA APPROACHES

Recently, machine learning (ML) methods have been developed and achieved significant success in the field of computer-aided design (CAD) [25], [26]. ML transforms traditional analysis, modeling, and optimization problems into data-to-data mapping problems, providing efficient and accurate performance evaluation at various design stages [27], [28], [29], [30]. Convolutional neural networks (CNN) as ML models have the ability to extract and abstract features from image-based data, outperforming traditional shallow ML models in handling challenging tasks [31], [32]. In the CAD domain, CNNs have been utilized to detect layout manufacturability and reliability violations [29], [33], [34], [35], [36], [37]. CNNs have been developed to predict congestion heatmaps, and by utilizing the model, unnecessary searches can be avoided, thereby speeding up the overall routing process [38]. CNNs have also been proposed to provide routing guidance by emulating complex manual layout techniques [39]. Intuitively, PCB layout can naturally be represented as image-based data. Liao et al. [40] proposed a deep reinforcement learning approach to address global routing problems in a simulated environment. Chen et al. [41] introduced a model to predict thermal distribution issues in PCB routing, thereby reducing temperature during PCB runtime.

C. PARTIAL SUMMARY AND MOTIVATION

Firstly, the majority of current algorithms primarily rely on CPU computation, which prevents them from harnessing the increasing computational power of GPUs. This limitation hampers the efficiency of the algorithms. The proposed algorithm in this paper addresses this issue by utilizing GPU inference to compute the recommended routing areas, significantly reducing the routing time.

Secondly, among the approaches that incorporate ML models, only two papers [39], [40] directly apply them for routing guidance. Furthermore, these two papers mainly focus on the global routing problem in VLSI routing, and there is a lack of research on the unified routing problem specifically for PCBs. To fill this research gap, this paper proposes a model-based acceleration scheme for unified PCB routing.

IV. PROPOSED ALGORITHM

A. OVERALL FRAMEWORK OF THE ALGORITHM

To expedite the processing of Problem 1, Unet-Astar, a PCB routing framework guided by ML models, is proposed. Our algorithm primarily accelerates the routing process by pre-learning from a large dataset of routing examples and guiding the Astar algorithm.

The process of generating the training dataset for the algorithm is illustrated in Figure 1:

As shown in the figure, to train the model, a PCB problem generation tool is utilized to generate a substantial number of PCB problems. Then, using the Astar algorithm, the routing

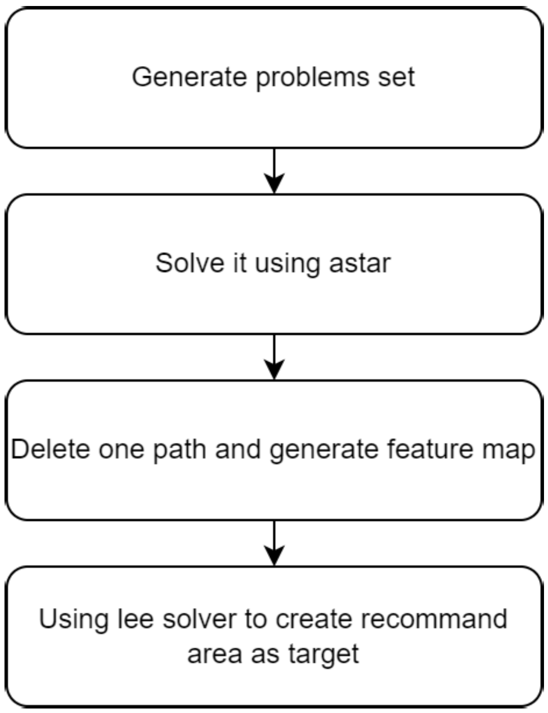

Generate problems set
Solve it using astar
Delete one path and generate feature map
Using lee solver to create recommand area as target

FIGURE 1. Unet-Astar train process.

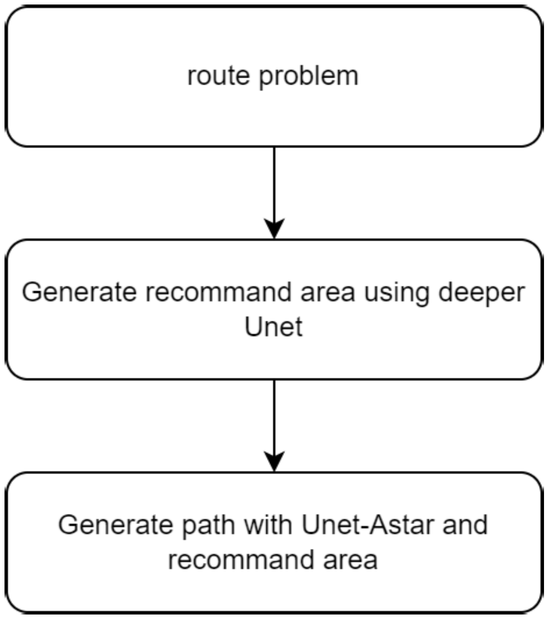

route problem
Generate recommand area using deeper Unet
Generate path with Unet-Astar and recommand area

FIGURE 2. Unet-Astar run process.

is performed. After the routing is completed, one random routing path is removed from the generated routes as the path to be learned. The remaining routes and SMD pads are used as inputs, and the recommendation area generator based on the Lee algorithm is employed to generate the recommendation areas. The recommendation areas serve as the outputs, and this process is iterated repeatedly to generate a large amount of training data.

The algorithm execution process is illustrated in Figure 1.

When encountering a new routing problem, the current routes and other SMD pads are initially input into the model as obstacles. The model infers the recommendation areas, which are then combined with the current obstacles as inputs for the Astar algorithm. The Astar algorithm performs routing and generates new paths. This process is repeated until all routes have been completed.

B. MODEL ARCHITECTURE

The neural network used in this study requires a multi-channel image as input and produces a single-channel image as output. Each grid point in the output image is represented by a floating-point number between 0 and 1, indicating the probability. A value of 0 indicates not recommended, while a value of 1 indicates recommended. This characteristic is similar to that of segmentation networks, hence the adoption of the Unet network as the backbone in this study. However, the traditional U-net [42] is limited to short-range spatial information due to the use of local receptive fields in traditional convolutions. To enable long-range inference, a larger receptive field is required. Therefore, this study improves upon the original U-net by increasing the number of convolutional layers and adding three fully connected layers at the final layer to enhance long-range inference capability. This modified network is referred to as the Deeper Unet. The loss function used for training is defined in Equation 1.

$$loss = 0.5 * mseloss + 0.5 * diceloss \tag{1}$$

The network architecture is shown in Figure 3.

In the figure, "C" represents a double convolutional layer, "P" represents a 2×2 max pooling layer, "D" represents a 2×2 transposed convolutional layer and "FC" represents a fully connected layer.

The structure of the double convolutional layer is illustrated in Figure 4.

In the figure, "R" represents the ReLU function.

Compared to the original U-net, the new Deeper Unet has significantly increased its depth from 5 layers to 9 layers. It also includes additional fully connected layers, enhancing its long-range inference capability.

C. FEATURE EXTRACTION FROM LAYOUT

To extract feature patterns from the layout and perform grid-based routing, the entire layout is rasterized.

A layout with L layers is encoded as a tensor with $L + 1$ channels. (In this paper, the assumption is made that all pads are through-hole, thus representing the starting and ending points using only one layer.)

The first layer represents the starting and ending points of the routing problem, as shown in Figure 5:

The red circles represent the starting and ending points, respectively. The surrounding halo is generated using a normal distribution, which helps the model learn the positions of the starting and ending points more effectively. In the image, heightened luminosity corresponds to proximity to the starting and ending points.

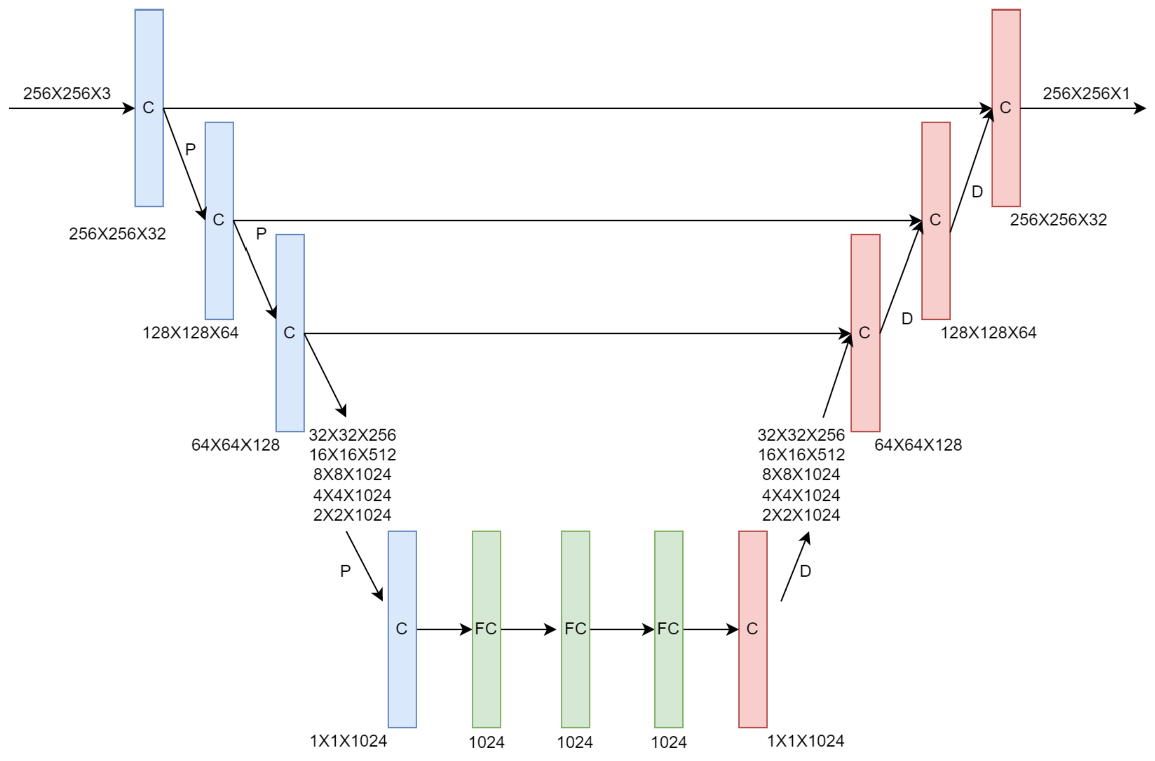


FIGURE 3. Structure of Deeper-Unet.

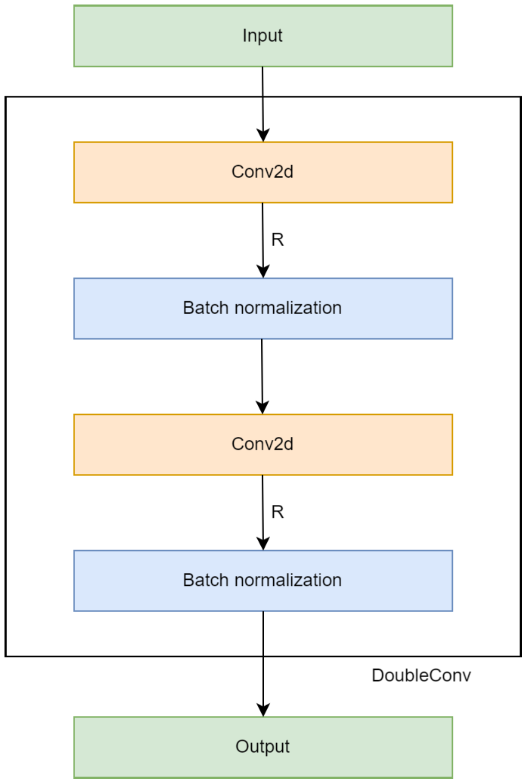


FIGURE 4. Structure of the double convolutional layer.

The remaining layers represent obstacles, as shown in Figures 6 and 7.

Each layer of the PCB corresponds to one channel, where black areas represent 0, indicating the routable region, and white areas represent 1, indicating the pads or interconnections of other nets.

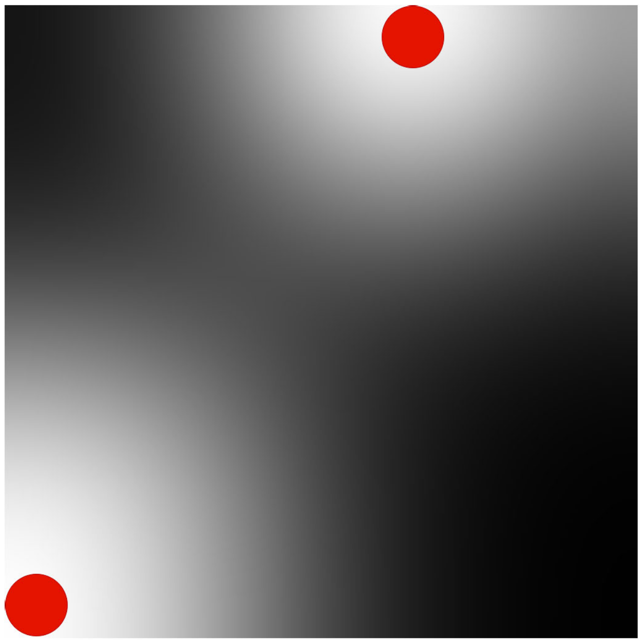


FIGURE 5. Feature map generated for starting and ending points. The red circles denote the starting and ending points. Heightened luminosity corresponds to proximity to the starting and ending points.

D. MODEL TRAINING - TRAINING SET CONSTRUCTION

To facilitate training, a PCB routing problem dataset generator was developed. This generator can automatically generate

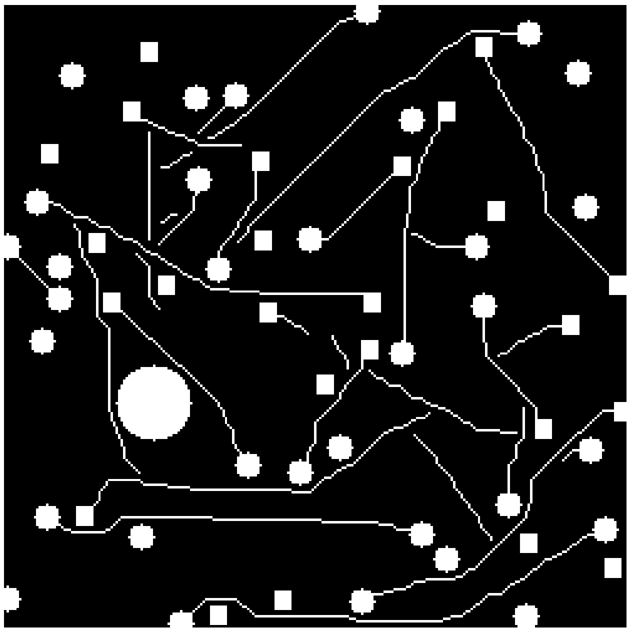


FIGURE 6. **Feature map generated for obstacles in Layer 1.**

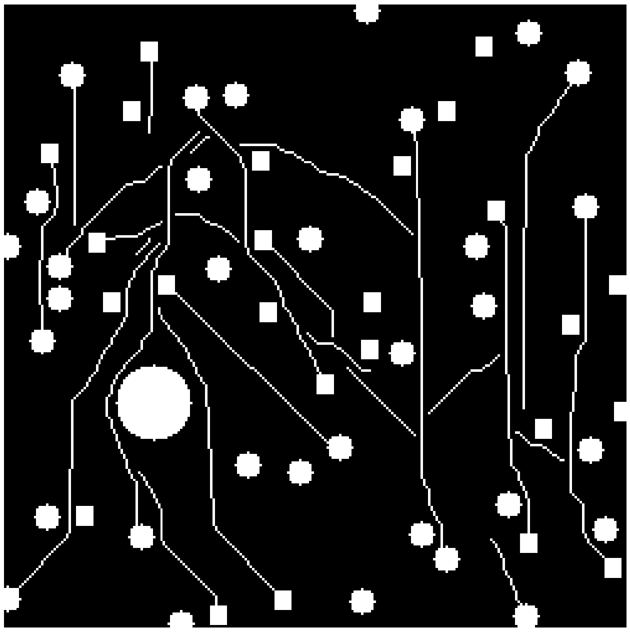


FIGURE 7. **Feature map generated for obstacles in Layer 2.**

a large-scale problem database, including user-specified problem IDs, grid sizes, net counts, pin counts, and obstacle counts. The code for this generator is available at the open-source repository mentioned in the introduction. The generated problem examples were shown in the previous section.

To train the network, the generated problems need to be annotated to provide the recommended regions for the network to learn. A solver similar to the Lee algorithm is used for annotation to generate the recommended regions. The input to the solver is a layout, a starting point, and an ending point, and the output is a recommended region.

The algorithm starts by initializing two search lists: the starting point exploration list and the ending point exploration list. The starting point and ending point are added to their respective lists as initial points, with their distances marked as 0.

Then, iteratively, the points with the minimum distance from both lists are taken out, and their neighbors are computed. All neighbors are traversed, and the distances from the initial points to these neighbors are calculated. The neighbors are added to the lists.

If a neighbor is present in the other list, it indicates that the neighbor can be reached from both the starting point and ending point. The sum of the distances from the starting point and ending point to this neighbor is computed. If this distance is smaller than the shortest path distance, the neighbor is added to the recommended region.

After this step, the exploration in the two lists no longer increases, and only the points that are already present in the other list are explored, until both lists have no more points to explore. (This step is to reduce unnecessary computations, and exploring the entire graph is also feasible.)

At this point, the sum of the distances from all points to the starting point and ending point is calculated. If this distance is smaller than the shortest path distance plus 0.1 times the Euclidean distance between the starting point and ending point, the point is added to the recommended region.

With this, the calculation of the recommended region is completed.

The pseudocode for the training example generation algorithm is shown in Algorithm 1.

To improve the performance of the algorithm in predicting longer paths, some training samples were removed during the generation of the training set. Assuming a problem has 10 connections ranked from 1 to 10 based on their lengths(The shorter the distance, the higher the ranking), the samples in the top 30% of rankings will be deleted. In this example, routes with rankings 1, 2, and 3 will be removed.

An example of a generated sample is shown in Figures 8 and 9.

In the figures, yellow represent pads, blue lines represent connections from other nets, white areas represent the generated recommendation area, light blue represents the removed connections, and red represents the search range of the A* algorithm when no recommendation area is available.

The crimson region emanates from the red point, and in the absence of obstacles, it extends toward the green point, thereby exhibiting maximal search efficiency. However, the presence of obstacles necessitates the red region to diverge exploration in various directions, markedly diminishing the search efficiency. To enhance the efficiency of exploration, the adoption of a neural network approach is chosen to curtail fruitless searches. As a breadth-first algorithm, the Lee algorithm facilitates the determination of the shortest path through systematic traversal, serving as a robust avenue for exploration. Yet, if the neural network were tasked with direct acquisition of this path, the endeavor would

Algorithm 1 Training Set Generation Algorithm

```
1:  Initialize the starting point exploration list and add the starting point to the list. The distance list for the starting point is [0, $\infty$].
2:  Initialize the ending point exploration list and add the ending point to the list. The distance list for the ending point is [$\infty$, 0].
3:  Initialize the shortest path distance as $\infty$.
4:  Initialize the exploration flag as True.
5:  List Collection = [starting point exploration list, ending point exploration list].
6:  while There is node in List Collection do
7:      for i = [0, 1] do // Iterate over the two lists
8:          List = List Collection[i]
9:          Current node = Extract the point with the minimum distance from the list.
10:         Neighbors = All points reachable from the current node.
11:         Exploration flag = explorer(i, exploration flag, neighbors, List Collection, Current node) // Algorithm 2
12:     end for
13: end while
14: for node in List Collection do
15:     if The total distance of the node is less than the shortest path distance + 0.1 times the Euclidean distance between the starting point and ending point then
16:         Add the node to the recommended region.
17:     end if
18: end for
19: Return the recommended region.
```

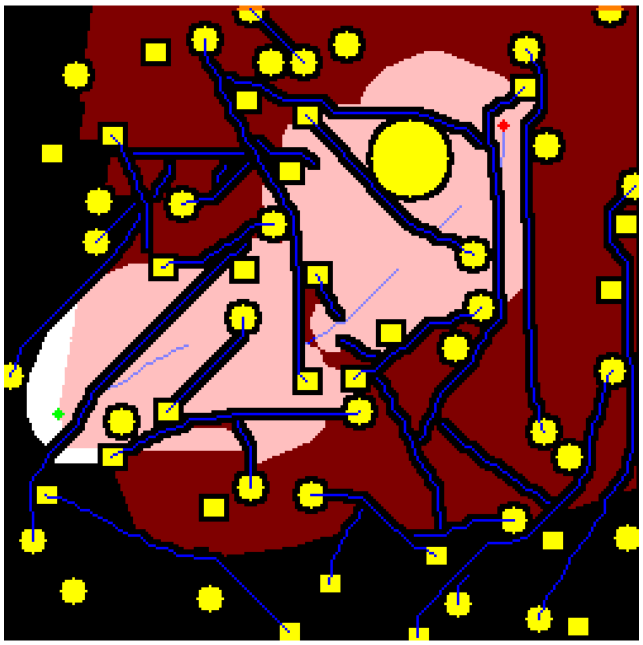


FIGURE 8. Generated recommendation area for layer 1.

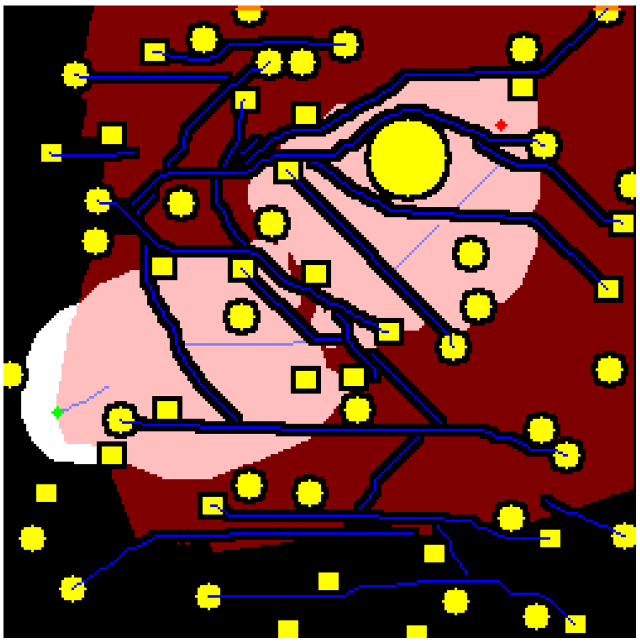


FIGURE 9. Generated recommendation area for layer 2.

prove exceptionally challenging, given the intricate nature of useful insights' acquisition. Hence, in addition to learning the shortest path identified by the Lee algorithm, pathways slightly deviating from the optimal trajectory are also considered for learning. This augmentation provides the neural network with the opportunity to acquire a broader array of knowledge. The white regions in the diagram delineate the areas generated by the Lee algorithm. The illustration reveals the white regions enveloping the green point (origin), the red point (destination), and the pale blue lines (original path). Furthermore, the white regions exhibit notable reduction in extent compared to the crimson regions, underscoring the potential of significantly reducing the search space through effective learning by the neural network.

Algorithm 2 Explorer Algorithm

```
1:  Input: i, exploration flag, neighbors, List Collection, Current node
2:  for neighbor in neighbors do:
3:      if exploration flag is False and neighbor is not in List Collection[1-i] then
4:          continue
5:      end if
6:      Tentative distance = current node distance[i] + distance from current node to neighbor
7:      if Tentative distance is less than neighbor's distance[i] then
8:          Neighbor's distance[i] = Tentative distance
9:          Neighbor's parent node = current node
10:         if Neighbor is not in the current list then
11:             Add neighbor to the current list
12:         end if
13:     end if
14:     if Neighbor is in the other list then
15:         Neighbor's total distance = Neighbor's distance[0] + Neighbor's distance[1] // Minimum distance from the starting point to the ending point passing through the neighbor
16:         if Total distance is less than the shortest path distance then
17:             Shortest path distance = Neighbor's total distance
18:         else
19:             Exploration flag = False
20:         end if
21:     end if
22: end for
23: Return exploration flag
```

E. ALGORITHM EXECUTION

The design of this algorithm (Unet-Astar) is based on the A* algorithm used in the paper [1].

Unlike the original algorithm, the proposed algorithm uses a new cost function.

The new cost function is defined as follows:

$$f(n) = g(s, c) + h(c, t) - \text{liner-power} * r(c)$$

Here, $g(s, c)$ represents the cost from source s to the current position c, $h(c, t)$ represents the estimated cost from the current position c to the target t. The cost includes wirelength, turn penalty, and violation penalty. "liner-power" is a hyperparameter that controls the trade-off between the recommendation area and routing cost. $r(c)$ represents the value of the recommendation area, which is a floating-point number between 0 and 1, predicted by the model before routing.

Additionally, since many shorter wires in PCB routing are relatively simple and do not benefit much from the recommendation area, a hyperparameter "skip-percent" is introduced. Similar to the exclusion of training samples, if a wire is ranked as the n_{th} wire based on its length among a total of 10 wires, it will be directly routed using the traditional algorithm if $n/10 <$ skip-percent.

The distance flow for routing the entire PCB board is as follows:

1. Decompose all multi-terminal nets into multiple two-terminal nets using the Steiner Tree algorithm.

2. Sort all two-terminal nets based on the straight-line length of the wire, with shorter wires having higher rankings.

3. Perform routing in ascending order of rankings, from the lowest ranked wire to the highest.

TABLE 1. Pairwise comparison between the proposed algorithm and others.

	Dice	Hd	Iou
Unet [42]	0.859	42.0	0.776
DeeperUnet	0.931	41.7	0.890
Unet++ [43]	0.861	43.1	0.782

V. EXPERIMENTS

A. EXPERIMENTAL DETAILS AND HARDWARE ENVIRONMENT

The algorithm is implemented using the Python programming language. All algorithms were executed on a personal computer with an Intel Core i7-7700K CPU, an NVIDIA GeForce RTX 2080 Ti GPU, 64GB of RAM, and running the Windows 10 operating system.

B. COMPARATIVE EXPERIMENT ON RECOMMEND AREA GENERATED

This experiment presents a comparative study of the divergent performances exhibited by the DeeperUnet model proposed

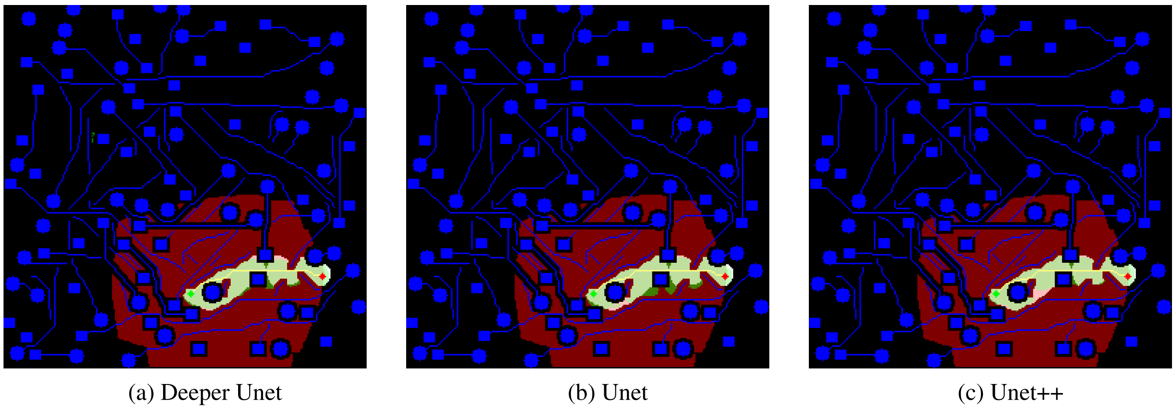


FIGURE 10. The maps generated by different neural networks under conditions of relatively shorter net lengths.

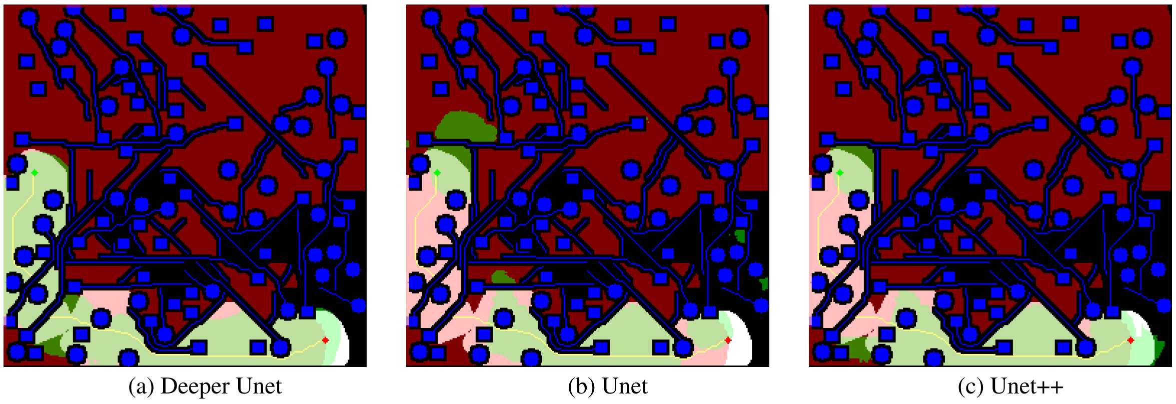


FIGURE 11. The maps generated by different neural networks under conditions of relatively longer net lengths.

within this paper, in contrast to the conventional Unet architecture as well as the Unet++ framework, all assessed upon an identical testing dataset.

To quantitatively evaluate the performance of the proposed method, we calculated three evaluation indicators, dice coefficient (Dice), hausdorff distance (Hd), and intersection over union (IoU), which are calculated as follows.

$$\begin{aligned} \text{Dice} &= \frac{2\left|I_{seg} \cap I_{gt}\right|}{I_{\text{seg}} + I_{gt}} \\ Hd &= \text{Max}\left(\left\{I_{\text{seg}} - I_{gt}\right\}\right) \\ Iou &= \frac{I_{\text{seg}} \cap I_{gt}}{I_{\text{seg}} \cup I_{gt}} \end{aligned}$$

where I_{seg} represents the predicted segmentation result and I_{gt} represents the ground truth.

The quantitative average scores on the test dataset are shown in Table 1. It can be observed that the proposed DeeperUnet has achieved a dramatic improvement in segmentation performance. As for the overlap rate index, the average scores of Dice and Iou are 0.931 and 0.890, respectively. Hd is mainly used to evaluate the segmentation accuracy of the boundary. It can be concluded from Table 1 that the Hd scores of U-Net and U-Net++ are lower than our method.

In addition, we randomly select two different length of net from the prediction results, and the predicted results and the real results are marked in the original image. As shown in Figure 10 and 11.

In Figure 10 and 11, the black regions represent void spaces, the red regions correspond to the areas explored through A* search, while the green regions depict the maps generated by the neural network. The white areas denote the ground truth generated by the Lee algorithm, and the yellow lines indicate the originally eliminated pathways.

Figure 10 illustrates an instance involving a shorter net, while Figure 11 presents an example with a longer net.

In Figure 10, it can be observed that the DeeperUnet model exhibits the closest adherence to the ground truth. The generative regions of the Unet model correspond relatively well to the ground truth; however, the generated map tends to be slightly larger than the ground truth. Similarly, Unet++ demonstrates similarities to the Unet model, yet the generated map appears slightly smaller than the ground truth.

Turning to Figure 11, it is evident that the outcomes generated by the DeeperUnet model closely align with the

TABLE 2. Pairwise comparison between the proposed algorithm and Algorithm 2.

	Wire Length	Via Count	Runtime (s)	Routability (%)
Original	8822	331	302	100
Proposed	8797	257	78	100

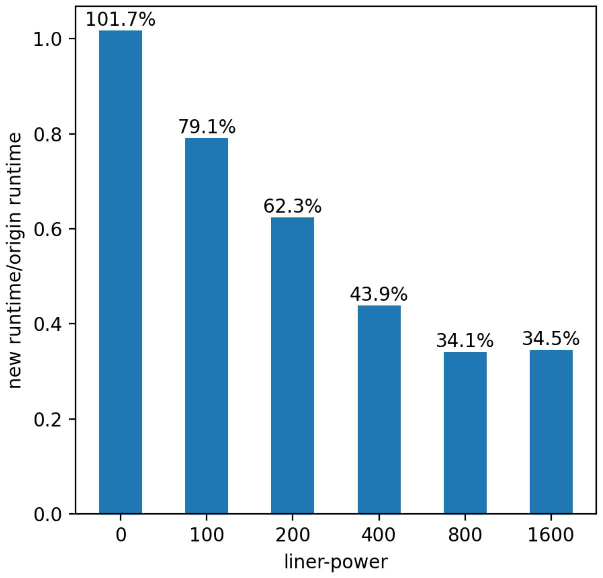


FIGURE 12. Bar chart showing runtime and "liner-power".

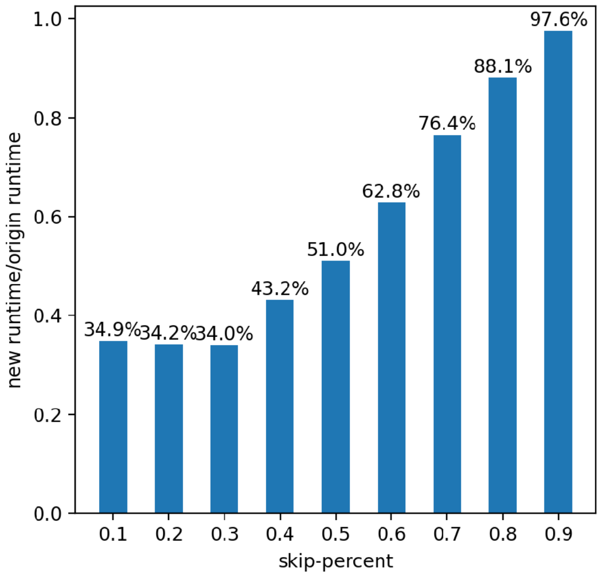


FIGURE 13. Bar chart showing runtime and "skip-percent".

ground truth. In contrast, the maps generated by the Unet and Unet++ models exhibit notable disparities when compared to the ground truth. Particularly in regions distant from the origin and termination points, inaccuracies in prediction become evident.

This observation suggests that certain optimization techniques tailored for medical image segmentation may not necessarily yield beneficial outcomes in this context. Unet++ exhibits the poorest performance in this task. Due to its limited receptive field, Unet delivers reasonable results for shorter nets but struggles to predict the ground truth accurately for longer nets. The proposed DeeperUnet architecture, introduced in this paper, augments the network depth, resulting in a larger receptive field. This affords robust performance across various net configurations and predictive regions.

These findings underscore the advantageous characteristics of the proposed network architecture outlined in this study.

C. HYPER-PARAMETER ANALYSIS

This experiment investigates the sensitivity of the Unet-Astar algorithm to different parameters. For this experiment, 20 test problems were generated using the PCB problem generator as benchmark problems to study the parameter sensitivity of the Unet-Astar algorithm.

The sensitivity of two important parameters, namely "liner-power" and "skip-percent", in the Unet-Astar algorithm were studied. For the "liner-power" parameter, five different values (100, 200, 400, 800, 1600) were considered in this analysis. For the "skip-percent" parameter, nine different values (0.1, 0.2, ..., 0.8, 0.9) were considered in this analysis.

Figure 12 shows the time required for the algorithm to run under different values of "liner-power". Due to the additional processing related to the neural network during the runtime, the algorithm is approximately 1.7% slower than the original Astar algorithm when "liner-power" is set to 0. This indicates that the added recommendation area has almost no impact on the speed of the algorithm. It can be observed that the algorithm achieves the fastest runtime when "liner-power" is set to 800. When "liner-power" is set to 1600, the runtime slightly increases compared to 800. The reason behind this phenomenon is that a too-small "liner-power" value cannot effectively guide the routing process, while a too-large value will impose excessive restrictions on the path search.

Figure 13 shows the time required for the algorithm to run under different values of "skip-percent". It can be observed that the algorithm achieves the fastest runtime

TABLE 3. Pairwise comparison between the proposed algorithm and origin algorithm.

	Wire Length	Via Count	Runtime
WIN	109	191	200
LOSS	91	9	0
p-value	1.6210^{-25}	1.9210^{-43}	$2.08 * 10^{-45}$

when "skip-percent" is set to 0.3. In the range of 0.1-0.3, there is a slight decrease in runtime, indicating that the Unet-Astar algorithm performs worse than the original Astar algorithm within this range. This phenomenon is consistent with expectations because the training was only conducted on samples with "skip-percent" greater than or equal to 0.3.

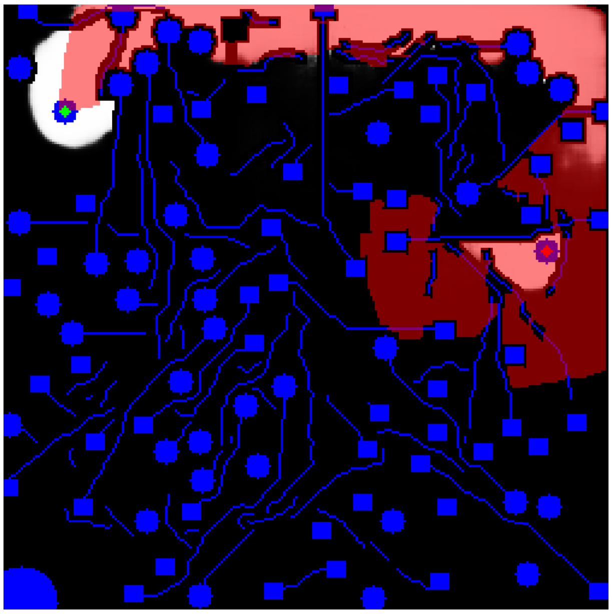
D. COMPARATIVE EXPERIMENT ON PCB ROUTING TASKS

To evaluate the effectiveness of Unet-Astar, its performance was compared with the algorithm implemented in Paper [1](referred to as origin algorithm). The test cases consist of 200 routing problems generated using the problem set generator introduced earlier. The parameters for the Unet-Astar algorithm are set as follows: liner-power=800, skip-percent=0.3. Except for the additional processing related to the neural network, the runtime logic of Unet-Astar is identical to origin algorithm.

Table 2 shows the experimental results and reports the average values of total wire length, via count, routability, and runtime for the 200 test problems. As shown in the table, both algorithms achieve 100% routability for all benchmarks without violating any design rules. However, our proposed algorithm significantly reduces the runtime by approximately 4 times (from 302 to 78) compared to the algorithm proposed in Paper [1]. It also reduces the total wire length by approximately 0.2% (from 8822 to 8797) and the via count by around 22% (from 331.11 to 257.295). These results indicate that our algorithm outperforms origin algorithm in terms of runtime, total wire length, and via count.

To provide a more comprehensive statistical analysis, non-parametric statistical tests were performed. Table 3 presents the pairwise comparison between the proposed algorithm and origin algorithm, focusing on wire length, via count, and runtime. The "WIN" row represents the number of test cases where our proposed algorithm outperforms origin algorithm, and the "LOSS" row represents the number of test cases where our proposed algorithm performs worse. The "p-value" row provides the statistical significance from the Wilcoxon test [44], [45] of the performance difference between the two algorithms. As indicated by the extremely small p-values (all less than 10^{-25}), our proposed algorithm significantly outperforms origin algorithm in terms of wire length, via count, and runtime across all test cases.

In general, based on the results of the Wilcoxon test, the proposed Unet-Astar algorithm significantly outperformed the original algorithm in terms of wire length, via count, and runtime testing, with a significance level of $\alpha = 0.05$.


(a) New Method Layer 1

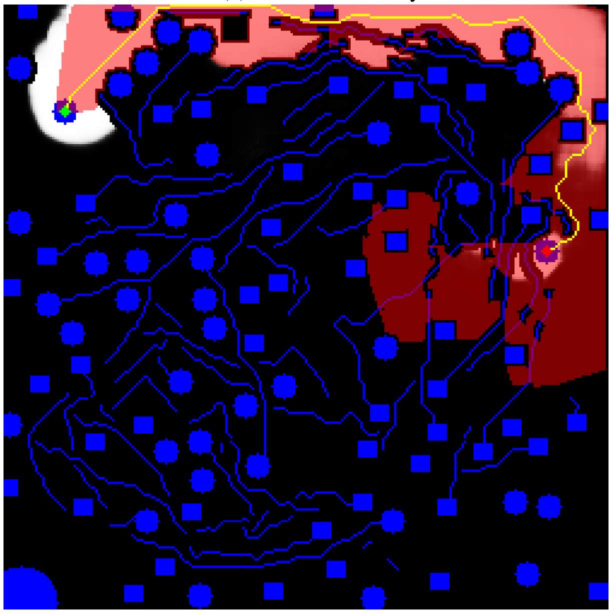
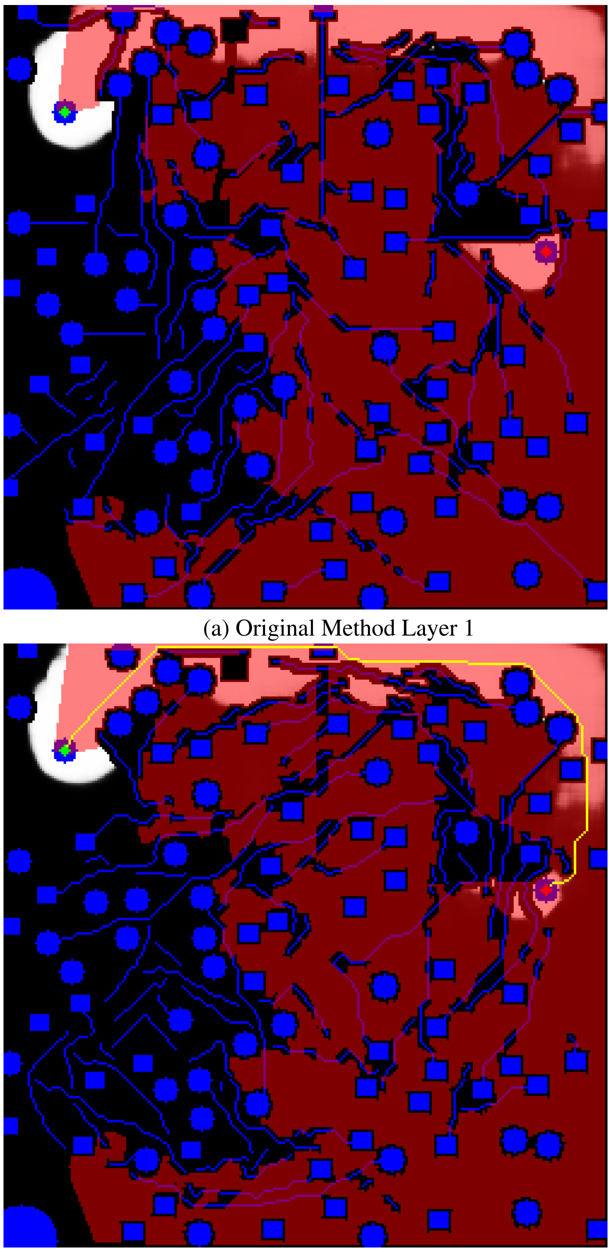

(b) New Method Layer 2

FIGURE 14. Example images of routing using Unet-Astar.

The results of the routing tests are illustrated in Figure 14, 15, demonstrating the superior performance of the proposed Unet-Astar algorithm compared to the original algorithm.

In the figures, the blue color represents the obstacle regions, the red color represents the search scope of the Astar algorithm, the red dots represent the starting points, the green dots represent the ending points, and the yellow lines represent the founded path.

From the figures, it can be observed that the new algorithm, compared to the traditional algorithm, has a much smaller search area. This characteristic is the main reason for the improved runtime of the new algorithm and demonstrates the effectiveness of the proposed algorithm.



(a) Original Method Layer 1

(b) Original Method Layer 2

FIGURE 15. Example images of routing using Astar.

However, it is discernible from the graphical representation that although the paths generated by the novel algorithm closely resemble those produced by the original algorithm, there are instances of redundant curvature when compared to the latter. This issue stems from the instability inherent in the generated recommendation regions. As depicted in Figure 14b, in the vicinity of the red points, the network fails to predict the recommendation region accurately, resulting in a certain degree of curvature in the generated routes to conform to the recommendation region. The underlying cause of this predicament may lie in the inadequacy of the network architecture based on traditional segmentation networks to fully cater to the requirements of this novel problem. Consequently, the necessity arises to devise a fresh network structure tailored to address the demands of the issue at hand. Given the idiosyncrasies associated with human wiring practices, network architectures founded upon attention mechanisms exhibit considerable promise in this context. Accordingly, we intend to pursue further research in this direction in the foreseeable future.

Overall, compared to the previous algorithm, Unet-Astar exhibits outstanding performance. By utilizing GPU computation for generating the recommendation region, the routing algorithm is greatly accelerated.

VI. CONCLUSION

In this paper, Unet-Astar, a PCB routing framework using ML models, is proposed. The model aims to capture remote contextual information from PCB layouts and predict feasible regions to accelerate the routing process. Three experiments were conducted on generated benchmarks, demonstrating that our Unet-Astar achieves significant speedup compared to state-of-the-art PCB uniform routing algorithms.

In the future, further optimization of prediction accuracy for the recommendation region generation model is planned, along with the incorporation of additional constraint conditions to enhance the performance of PCB routing algorithms.

REFERENCES

[1] S.-T. Lin, H.-H. Wang, C.-Y. Kuo, Y. Chen, and Y.-L. Li, "A complete PCB routing methodology with concurrent hierarchical routing," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 1141–1146.

[2] R. Wang, R. Shi, and C.-K. Cheng, "Layer minimization of escape routing in area array packaging," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design*, Nov. 2006, pp. 815–819.

[3] J.-W. Fang, I.-J. Lin, Y.-W. Chang, and J.-H. Wang, "A network-flow-based RDL routing algorithmz for flip-chip design," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 8, pp. 1417–1429, Aug. 2007.

[4] J.-W. Fang and Y.-W. Chang, "Area-I/O flip-chip routing for chip-package co-design," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2008, pp. 518–522.

[5] J.-W. Fang, M. D. F. Wong, and Y.-W. Chang, "Flip-chip routing with unified area-I/O pad assignments for package-board co-design," in *Proc. 46th ACM/IEEE Design Autom. Conf.*, Jul. 2009, pp. 336–339.

[6] T. Yan and M. D. F. Wong, "Correctly modeling the diagonal capacity in escape routing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 31, no. 2, pp. 285–293, Feb. 2012.

[7] T. Yan and M. D. F. Wong, "A correct network flow model for escape routing," in *Proc. 46th ACM/IEEE Design Autom. Conf.*, Jul. 2009, pp. 332–335.

[8] L. Luo and M. D. F. Wong, "Ordered escape routing based on Boolean satisfiability," in *Proc. Asia South Pacific Design Autom. Conf.*, Jan. 2008, pp. 244–249.

[9] L. Luo and M. D. F. Wong, "On using SAT to ordered escape problems," in *Proc. Asia South Pacific Design Autom. Conf.*, Jan. 2009, pp. 594–599.

[10] L. Luo, T. Yan, Q. Ma, M. D. F. Wong, and T. Shibuya, "B-escape: A simultaneous escape routing algorithm based on boundary routing," in *Proc. 19th Int. Symp. Phys. Design*, Mar. 2010, pp. 19–25, doi: 10.1145/1735023.1735033.

[11] L. Luo, T. Yan, Q. Ma, M. D. F. Wong, and T. Shibuya, "A new strategy for simultaneous escape based on boundary routing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 2, pp. 205–214, Feb. 2011.

[12] J.-W. Fang, C.-H. Hsu, and Y.-W. Chang, "An integer-linear-programming-based routing algorithm for flip-chip designs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 1, pp. 98–110, Jan. 2009.

[13] Y. Kubo and A. Takahashi, "Global routing by iterative improvements for two-layer ball grid array packages," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 4, pp. 725–733, Apr. 2006.

[14] M. M. Ozdal and M. D. F. Wong, "A length-matching routing algorithm for high-performance printed circuit boards," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 12, pp. 2784–2794, Dec. 2006.

[15] M. M. Ozdal and M. D. F. Wong, "Algorithmic study of single-layer bus routing for high-speed boards," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 3, pp. 490–503, Mar. 2006.

[16] T. Yan and M. D. F. Wong, "BSG-route: A length-constrained routing scheme for general planar topology," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 11, pp. 1679–1690, Nov. 2009.

[17] J.-T. Yan and Z.-W. Chen, "Obstacle-aware length-matching bus routing," in *Proc. Int. Symp. Phys. Design*, Mar. 2011, pp. 61–68, doi: 10.1145/1960397.1960412.

[18] M. M. Ozdal, M. D. F. Wong, and P. S. Honsinger, "Simultaneous escape-routing algorithms for via minimization of high-speed boards," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 1, pp. 84–95, Jan. 2008.

[19] J.-W. Fang, K.-H. Ho, and Y.-W. Chang, "Routing for chip-package-board co-design considering differential pairs," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2008, pp. 512–517.

[20] T. Yan, P.-C. Wu, Q. Ma, and M. D. F. Wong, "On the escape routing of differential pairs," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 2010, pp. 614–620.

[21] J.-W. Fang and Y.-W. Chang, "Area-I/O flip-chip routing for chip-package co-design considering signal skews," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 29, no. 5, pp. 711–721, May 2010.

[22] T.-H. Li, W.-C. Chen, X.-T. Cai, and T.-C. Chen, "Escape routing of differential pairs considering length matching," in *Proc. 17th Asia South Pacific Design Autom. Conf.*, Jan. 2012, pp. 139–144.

[23] K. Wang, S. Dong, H. Wang, Q. Chen, and T. Lin, "Mixed-crossing-avoided escape routing of mixed-pattern signals on staggered-pin-array PCBs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 33, no. 4, pp. 571–584, Apr. 2014.

[24] F. Jiao and S. Dong, "Ordered escape routing with consideration of differential pair and blockage," *ACM Trans. Design Autom. Electron. Syst.*, vol. 23, no. 4, pp. 1–26, May 2018, doi: 10.1145/3185783.

[25] T. Chen, G. L. Zhang, B. Yu, B. Li, and U. Schlichtmann, "Machine learning in advanced IC design: A methodological survey," *IEEE Des. Test*, vol. 40, no. 1, pp. 17–33, Feb. 2023.

[26] M. Rapp, H. Amrouch, Y. Lin, B. Yu, D. Z. Pan, M. Wolf, and J. Henkel, "MLCAD: A survey of research in machine learning for CAD keynote paper," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 10, pp. 3162–3181, Oct. 2022.

[27] T. Chen, Q. Sun, and B. Yu, "Machine learning in nanometer AMS design-for-reliability: (Invited paper)," in *Proc. IEEE 14th Int. Conf. ASIC (ASICON)*, Oct. 2021, pp. 1–4.

[28] T. Chen, Q. Sun, C. Zhan, C. Liu, H. Yu, and B. Yu, "Deep H-GCN: Fast analog IC aging-induced degradation estimation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 7, pp. 1990–2003, Jul. 2022.

[29] W. Jin, L. Chen, S. Sadiqbatcha, S. Peng, and S. X.-D. Tan, "EMGraph: Fast learning-based electromigration analysis for multi-segment interconnect using graph convolution networks," in *Proc. 58th ACM/IEEE Design Autom. Conf. (DAC)*, Dec. 2021, pp. 919–924.

[30] T. Chen, Q. Sun, C. Zhan, C. Liu, H. Yu, and B. Yu, "Analog IC aging-induced degradation estimation via heterogeneous graph convolutional networks," in *Proc. 26th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2021, pp. 898–903.

[31] N. Jmour, S. Zayen, and A. Abdelkrim, "Convolutional neural networks for image classification," in *Proc. Int. Conf. Adv. Syst. Electric Technol. (IC_ASET)*, Mar. 2018, pp. 397–402.

[32] T. Chen, B. Duan, Q. Sun, M. Zhang, G. Li, H. Geng, Q. Zhang, and B. Yu, "An efficient sharing grouped convolution via Bayesian learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 7367–7379, Dec. 2022.

[33] H. Yang, J. Su, Y. Zou, Y. Ma, B. Yu, and E. F. Y. Young, "Layout hotspot detection with feature tensor generation and deep biased learning," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 6, pp. 1175–1187, Jun. 2019.

[34] R. Chen, W. Zhong, H. Yang, H. Geng, X. Zeng, and B. Yu, "Faster region-based hotspot detection," in *Proc. 56th ACM/IEEE Design Autom. Conf. (DAC)*, Jun. 2019, pp. 1–6.

[35] H. Geng, H. Yang, L. Zhang, J. Miao, F. Yang, X. Zeng, and B. Yu, "Hotspot detection via attention-based deep layout metric learning," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Nov. 2020, pp. 1–8.

[36] B. Zhu, R. Chen, X. Zhang, F. Yang, X. Zeng, B. Yu, and M. D. F. Wong, "Hotspot detection via multi-task learning and transformer encoder," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Nov. 2021, pp. 1–8.

[37] H. Zhou, W. Jin, and S. X.-D. Tan, "GridNet: Fast data-driven EM-induced IR drop prediction and localized fixing for on-chip power grid networks," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Nov. 2020, pp. 1–9.

[38] Z. Zhou, Z. Zhu, J. Chen, Y. Ma, B. Yu, T.-Y. Ho, G. Lemieux, and A. Ivanov, "Congestion-aware global routing using deep convolutional generative adversarial networks," in *Proc. ACM/IEEE 1st Workshop Mach. Learn. CAD (MLCAD)*, Sep. 2019, pp. 1–6.

[39] K. Zhu, M. Liu, Y. Lin, B. Xu, S. Li, X. Tang, N. Sun, and D. Z. Pan, "GeniusRoute: A new analog routing paradigm using generative neural network guidance," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2019, pp. 1–8.

[40] H. Liao, W. Zhang, X. Dong, B. Poczos, K. Shimada, and L. Burak Kara, "A deep reinforcement learning approach for global routing," *J. Mech. Design*, vol. 142, no. 6, Jun. 2020, Art. no. 061701, doi: 10.1115/1.4045044.

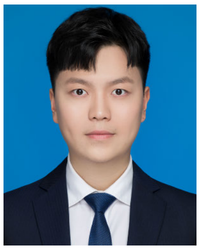
[41] T. Chen, S. Xiong, H. He, and B. Yu, "TRouter: Thermal-driven PCB routing via nonlocal crisscross attention networks," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 42, no. 10, pp. 3388–3401, Oct. 2023.

[42] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham, Switzerland: Springer, 2015, pp. 234–241.

[43] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: Redesigning skip connections to exploit multiscale features in image segmentation," *IEEE Trans. Med. Imag.*, vol. 39, no. 6, pp. 1856–1867, Jun. 2020.

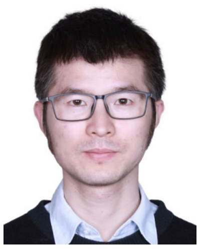
[44] C. García-Martínez and M. Lozano, "Evaluating a local genetic algorithm as context-independent local search operator for metaheuristics," *Soft Comput.*, vol. 14, no. 10, pp. 1117–1139, Aug. 2010.

[45] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011, doi: 10.1016/j.swevo.2011.02.002.



SHIYUAN YIN received the bachelor's degree in integrated circuit design and integrated systems from the University of Electronic Science and Technology. He is currently pursuing the Ph.D. degree with the High-Speed Circuit and Neural Network Laboratory, Institute of Semiconductors, Chinese Academy of Sciences. His research focuses on advanced information technology, specifically in the fields of computer science and artificial intelligence. His research interests include the application of reinforcement learning techniques to heuristic optimization algorithms, aiming to improve their efficiency by learning optimization algorithm features; investigates the use of large neighborhood search algorithms for solving learning parity with noise (LPN) problems, with the objective of enhancing the problem-solving capabilities; and the challenging task of PCB routing, where he combines deep learning and heuristic algorithms to develop an intelligent solution. This research aims to overcome the limitations of current routing methods, which often require manual intervention and waste valuable human resources.

MIN JIN received the degree in electronic science and technology from Wuhan University, in 2008, and the Ph.D. degree in microelectronics and solid-state electronics from the University of Chinese Academy of Sciences, in 2013. She is currently an Assistant Researcher with the Institute of Semiconductors, Chinese Academy of Sciences. Her research interests include neural networks, neural network algorithms, computational intelligence, and high-performance computing.



GANG CHEN received the Ph.D. degree from the Institute of Semiconductors, Chinese Academy of Sciences (CAS), in 2013. He is currently an Associate Researcher with the Institute of Semiconductors, CAS. He has published more than ten academic papers and filed six invention patents. His research interests include intelligent optimization algorithms, mixed-signal blind demodulation, and artificial intelligence chips.

GUOLIANG GONG received the bachelor's degree from the Department of Biomedical Engineering, Tianjin Medical University, in 2006, and the Ph.D. degree in engineering from the Institute of Semiconductors, Chinese Academy of Sciences (CAS), in 2012. He joined the High-Speed Circuits and Neural Networks Laboratory, Institute of Semiconductors, CAS, where he is currently the Deputy Director of the Brain-Like Computing Research Center and an Associate Researcher. His primary research interests include intelligent algorithms and brain-like computing systems, image processing chips, AI chips, brain-like neural computing chips, computing systems, neural network algorithms, and their applications.

WENYU MAO received the degree in electronic information engineering from Guizhou University, in 2008, and the Ph.D. degree in electronics from the University of the Chinese Academy of Sciences, in 2013. He is currently an Assistant Researcher. His research interests include artificial neural networks, computational intelligence algorithms, and systems.

HUAXIANG LU received the bachelor's degree from the Department of Information and Electronic Engineering, Zhejiang University, in 1985, and the master's and Ph.D. degrees in microelectronics from the Institute of Semiconductors, Chinese Academy of Sciences (CAS), in 1988 and 1993, respectively. Since 1997, he has been a Researcher with the Institute of Semiconductors, CAS. Currently, he holds various positions, including the Director of the Neural Networks and Computational Intelligence Professional Committee of the Chinese Association for Artificial Intelligence, the Director of the Beijing Key Laboratory for Semiconductor Neural Network Intelligent Perception and Computing Technology, and the Director of the Brain-Like Computing Center, Institute of Semiconductors, CAS. He has published more than 50 academic papers, coauthored one monograph, and holds more than ten authorized invention patents. His main research interests include brain-like neural computing methods, microelectronic neural computing chips and systems, uncertainty, incomplete information processing, brain-like neural computing chips, brain-like neural computing technology and application systems, and information and signal processing. He has received several awards and honors, such as the First Prize of Beijing Science and Technology Progress Award, the "Outstanding Young Scholar Award" from CAS, the Third Prize of the National Invention Award, the "Major Scientific and Technological Achievement Award of the Eighth Five-Year Plan" of China, the "Top Ten Electronic Science and Technology Achievements of '95" Award, and the Second Prize of Ministerial Science and Technology Progress Award.