Graph-Transformer-based Surrogate Model for Accelerated Converter Circuit Topology Design

Shaoze Fan[1], Haoshu Lu[1], Shun Zhang[2], Ningyuan Cao[3], Xin Zhang[2], and Jing Li[1]
[1]New Jersey Institute of Technology [2]MIT-IBM Watson AI Lab [3]University of Notre Dame

Abstract

Unlike circuit parameter and sizing optimizations, the automated design of analog circuit topologies poses significant challenges for learning-based approaches. One challenge arises from the combinatorial growth of the topology space with circuit size, which limits the topology optimization efficiency. Moreover, traditional circuit evaluation methods are time-consuming, while the presence of data discontinuity in the topology space makes the accurate prediction of circuit performance exceptionally difficult for unseen topologies. To tackle these challenges, we design a novel Graph-Transformer-based Network (GTN) as the surrogate model for circuit evaluation, offering a substantial acceleration in the speed of circuit topology optimization without sacrificing performance. Our GTN model architecture is designed to embed voltage changes in circuit loops and current flows in connected devices, enabling accurate performance predictions for circuits with unseen topologies. Taking the power converter circuit design as an experimental task, our GTN model significantly outperforms an analytical approach and baseline methods directly utilizing graph neural networks. Furthermore, GTN achieves less than 5% relative error and 196× speed-up compared with high-fidelity simulation. Notably, our GTN surrogate model empowers an automatic circuit design framework to discover circuits of comparable quality to those identified through high-fidelity simulation while reducing the time required by up to 98.2%.

1 INTRODUCTION

The rapid advancement of artificial intelligence has led to a new era of innovation in electronic design automation (EDA), particularly in the domain of analog circuit design. The analog circuit design workflow comprises four phases: circuit topology design, physical layout design, parasitic extraction, and validation. While learning-based EDA tools have achieved notable success in the lower-level design phases [6, 12, 18], the challenge of automating circuit topology design persists, predominantly relying on the expertise and experiences of human designers. Moreover, the field of analog circuit design offers abundant opportunities for novel topologies, as demonstrated by the continuous evolution of power converter topologies year after year [8]. Hence, the development of learning-based topology design tools holds immense promise and relevance in this dynamically expanding domain.

*Corresponding authors: jingli@njit.edu, xzhang@us.ibm.com.

DAC '24, June 23–27, 2024, San Francisco, CA, USA

ACM ISBN 979-8-4007-0601-1/24/06.
https://doi.org/10.1145/3649329.3656258

However, the automatic design of circuit topologies faces formidable challenges. The topology design space grows faster than an exponential increase with the circuit complexity and experiences a combinatorial explosion. For example, Figure 1 shows that the number of circuit topologies rapidly escalates to 191.8K for 5-component circuits, when each component can be chosen from 4 types and have 2 ports to be connected with other component ports or 3 terminal ports. Consequently, while a single simulation for a 5-component circuit may require just a few seconds, evaluating all these diverse topologies, each with multiple device parameters, demands an astonishing 111 days. Recent works have introduced machine learning algorithms tailored for circuit topology automation, encompassing tree-based search [5, 23] and genetic algorithms [13], which intelligently explore the circuit topology space. Nevertheless, the effectiveness of these algorithms hinges on the time cost and accuracy of circuit evaluation. Thus, they still suffer intolerably computational costs when employing high-fidelity simulations to assess the searched circuit designs.

Leveraging learning-based methods as surrogate models to predict circuit performance holds great promise for accelerating circuit evaluation, but it brings about distinctive challenges in the context of circuit topology design. Unlike surrogate models aimed at optimizing device parameters of a fixed topology that remains the same for both training and testing [2], those for topology design must possess the capability to estimate the performance of circuits with topologies previously unobserved in training data, as training a separate model for each topology is impractical for design automation. This necessitates that the model comprehends the inherent topological structures of circuits. Moreover, learning-based surrogate models for topology design suffer from severe data discontinuity. Unlike problems in the general AI field, where similarities in topologies often yield similar predictions, circuit design exhibits the phenomenon wherein even a minor change in topology can dramatically alter circuit characteristics and performance. Additionally, the data imbalance in topology space, due to the fact that good circuit candidates are inherently sparse, aggravates the difficulties in learning accurate prediction. As a result, applying existing models developed in the broader AI field, such as graph neural networks, proves less effective. Instead, learning-based surrogate models for topology design must be engineered to capture the fundamental physics underlying diverse topologies, enabling the transfer of knowledge gleaned from a limited set of topologies to those that are previously unseen. In pursuit of effective circuit topology design automation, we introduce a pioneering Graph-Transformer-based Network (GTN) as our surrogate model, aimed at precise circuit performance prediction and expedited circuit evaluation and topology exploration. Specifically, we make the following contributions.

- We redefine a circuit topology as a topological graph, where circuit devices and their connection nodes are both represented as

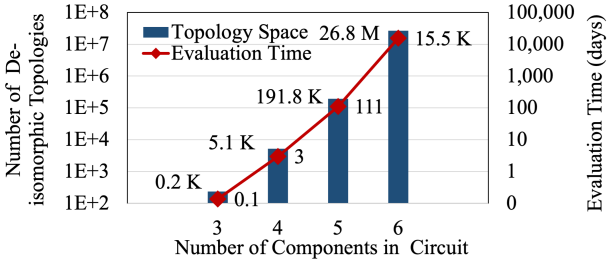


Figure 1: For circuits with 4 different component types, the topology space and circuit evaluation time using high-fidelity simulation increase combinatorially with circuit complexity.

graph vertices. This representation, though resulting in larger graphs compared to those with devices alone as vertices, enables the exact preservation of topological information, including the connectivity of different device ports. This proves pivotal for detecting cases, such as shortcuts arising from certain device port connections, which are crucial in identifying undesirable topologies generated by topology search algorithms. Furthermore, it can truthfully convey current flows among devices linked to the same connection node. To identify voltage changes within the circuit, we employ the minimum loop algorithm [10] to identify minimally overlapping loops where voltage changes can be effectively characterized with the smallest number of devices.

- We construct a novel GTN neural architecture that harnesses the topological knowledge to embed voltage changes in circuit loops and current flows between connected devices. This is achieved by employing multi-head attention mechanisms, both for aggregating embeddings of neighboring devices linked to the same connection node cultivating the current flow information, and for aggregating embeddings of devices within the same circuit loop cultivating the voltage change information. Although circuit loops can capture knowledge essential for accurate circuit performance prediction, their greater distance compared to neighbors makes training with a cold start difficult. Thus, we introduce a specialized learning procedure that starts with training only the aggregation of neighboring vertices via a special masking technique and progressively adds the aggregation of loop devices once the remaining networks have warmed up well.
- We focus on the power converter circuit topology design as an experimental task in this work. Evaluation results show that our GTN model reduces the relative squared error of an analytical approach and baseline methods directly utilizing various graph neural networks by more than 87.7% and 16.1%, respectively. Furthermore, GTN can predict circuits' power efficiency and output voltage with less than 5% relative error and 196× speed-up compared with high-fidelity simulation.
- We integrated our GTN surrogate model into a tree-based search for a circuit topology design automation framework. GTN empowers this framework to discover circuits of comparable quality to those identified through high-fidelity simulation while reducing the time required by up to 98.2%.

2 RELATED WORKS

AI-based Circuit Design Automation. Applying learning-based models for circuit design tasks, such as physical implementation [6, 12] and parameter optimization [18] of integrated circuits, have attracted increasing attention. However, most works focus on circuit optimization using a given circuit topology. For topology exploration, search algorithms have been investigated, including evolutionary, genetic [13], and tree-based search algorithms [5, 23].

Learning-based Circuit Evaluation. Conventional SPICE simulation software takes an extremely long time to evaluate one topology with fixed parameters. Analytical methods, like State-Space Averaging [14] for converter circuits, can reduce the time cost of evaluating one topology with different parameters, but they can only be applied to a small fraction of topologies that satisfy certain theoretical assumptions. Thus, recent works developed deep learning models to speed up circuit performance and characteristic estimation tasks, such as estimating dynamic voltage drop [20, 24], post-layout interconnect parasitics[16], and net lengths[19] of integrated circuits. Various Graph Neural Networks (GNN) were designed for estimating electromagnetic properties [21], power [22], transient electromigration stress [9], Circuit De-obfuscation Runtime [4], as well as guiding analog IC placement [11], layout parasitics and device parameter prediction [15], symmetry constraint extraction [3], reverse engineering [1] and arithmetic block identification [7].

However, most prior works focused on tasks that involve training and testing using a single topology or similar topologies. They tend to perform suboptimally when confronted with the challenges posed by a diverse topology design space with data discontinuity and imbalance. In contrast, this work aims for accurate performance prediction of diverse and unseen topologies explored by topology design automation frameworks.

3 PROBLEM STATEMENT

In this work, we aim to design a fast-running surrogate model f that can learn from a small amount of ground truth circuit evaluation results to accurately predict the key performance of a circuit with unobserved topology generated by an automatic circuit topology design framework. In this way, the surrogate model that can help the design automation framework generate circuits with the same quality as the framework using high-fidelity simulation while reducing the time cost as much as possible. In this context, a circuit consists of a *topology* s and a *device parameter* d. The circuit topology contains a set of device components with ports and edges connecting the ports. Each component has a device type (from the set of available devices in the design task) with device parameters (e.g., inductance, capacitance, transistor dimensions, and switching control parameters) and ports (i.e., left and right ports).

The accuracy of the surrogate model is measured by the Relative Squared Error $RSE = \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$, where $\hat{y}_i$ is the performance predicted by the model, y_i is the ground-truth performance given by the high-fidelity simulation F_{sim}, and $\bar{y}$ is the sample mean of the ground-truth performance. Intuitively, the RSE gives a normalized mean squared error. Thus, our goal is to design a surrogate model f that achieves the minimum RSE:

$$f^* = \arg\min_f \frac{\sum_i (F_{sim}(s_i, d_i) - f(s_i, d_i))^2}{\sum_i (F_{sim}(s_i, d_i) - \bar{F_{sim}})^2}$$

In this paper, we use the two-phase power converter circuit design as the exemplar and experimental design task. Specifically, we

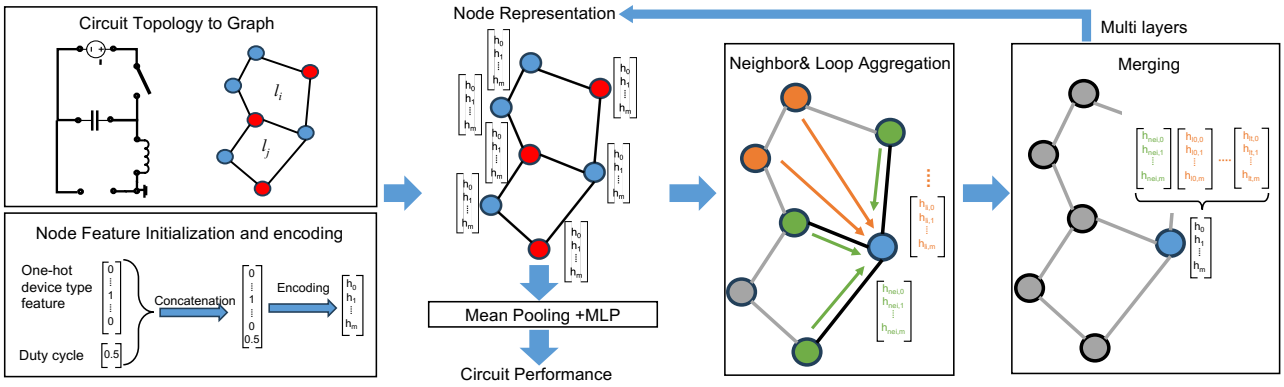


Figure 2: GTN Surrogate Model Structure.

consider 5-components circuits with devices types of capacitors (C), inductors (L), phase-I switches (S_a), and phase-II switches (S_b). Each component has two ports without direction. In addition to components, three external terminal ports are essential in each circuit: the input voltage (Vin), output voltage (Vout), and ground (Gnd) ports. For device parameter d, we consider the duty cycle of switches that affects the output voltage. We consider two key performance metrics of power converters, the output voltage $V_{\text{out}}(s, d)$ and power conversion efficiency ($\eta_{s,d}$). When automatically design power converter circuits, a high-quality converter is the one that has high efficiency and V_{out} close to a target output voltage V_0.

4 GTN SURROGATE MODEL

In this section, we present our topological graph representation of circuits and the insights from Kirchhoff's Current and Voltage Laws that are exploited in the novel model architecture of our Graph-Transformer-based Network (GTN) surrogate model.

4.1 Topological Graph Representation

In contrast to existing learning-based surrogate models that represent only the devices as vertices in a circuit's topological graph, our approach redefines the topological graph to include both devices and their connection nodes as graph vertices. While the graph representation with devices alone yields smaller graphs and expresses device connectivity, it lacks the vital information regarding which ports of devices are connected. This omission results in identical graph representations for circuits with different performances, such as high-quality circuits and those with shortcuts. In contrast, our topological graph representation (akin to the netlist representation) preserves precise topological information, proving essential in identifying undesirable topologies generated by topology search algorithms. Additionally, it accurately conveys current flows within the circuit, a key characteristic leveraged in our GTN architecture.

As described in Section 3, each device component is associated with a specific device type, while a connection node can either be one of the terminal ports or an internal connection. To effectively capture these distinct vertex types in the topological graph, we employ a one-hot representation as the vertex feature, where each bit corresponds to a particular device or connection type. Additionally, we concatenate the device parameters (normalized to be at the same scale of 1) into the vertex feature, where the parameter is set to 0 if the vertex is not the corresponding device type.

Furthermore, we supplement our topological graph with explicitly identified circuit loops, which are utilized in our GTN model to capture the conservation of charge and energy in circuit loops. While all circuit loops hold voltage information, they encompass largely overlapping voltage data. Moreover, using all circuit loops makes the model more complex and harder to train. To address this challenge, we leverage the minimum loop algorithm [10] implemented in the Networkx package to identify minimally overlapping loops (e.g., l_i and l_j in Figure 2). This strategic approach allows us to accurately characterize voltage information with a minimal number of devices, enhancing the GTN model's effectiveness.

4.2 Physics Insights Behind GTN Architecture

Real-world circuits often exhibit inherent complexity, rendering direct analysis challenging. It is also challenging for learning-based surrogate models to grasp their underlying physics if blindly employing standard deep learning models. To facilitate the model's comprehension of critical physics information embedded within circuit topologies, we draw inspiration from Kirchhoff's two fundamental laws of electrical circuits, which encapsulate the unique interplay of current, voltage, and resistance in electrical circuits. Specifically, Kirchhoff's Current Law dictates that the current entering a connection node or device within a circuit equals the current departing from the same node or device. Kirchhoff's Voltage Law states that, for any closed loops within an electrical circuit, the sum of the electrical voltage across the loop must equate to zero.

Our approach to incorporating these physics insights involves two key steps. Firstly, we explicitly encode the connected nodes (in addition to devices) as graph vertices and identify smallest loops within our topological graph representation. Subsequently, we tailor the architecture of our GTN model to aggregate the information within the small sub-circuits of interconnected vertices and smallest loops, which captures the nuances of charge and energy conservation. This synergistic integration bolsters GTN's capability in unraveling the intricate physics underpinning complex circuit topologies, enhancing its effectiveness in modeling circuits.

Note that the state-space averaging method [14] for analyzing power converter circuits is also built upon Kirchhoff's two laws. However, it solves the state-space equations based on the assumption that the current or voltage on each device is stable. The equation matrix, which is a canonical form for writing the differential equations of a circuit, presents challenges for certain circuit (e.g.,

circuits containing inductors connected in series). Our empirical evaluation of state-space averaging for 5-component circuits shows that 46% circuits cannot derive the canonical matrix. Additionally, among circuits that can derive the canonical matrix, an additional 7% were unable to derive the inverse matrix, a prerequisite for the state space averaging method. All these circuits have diverse performance and cannot be set to a constant performance value.

4.3 GTN Model Architecture

Our innovative GTN neural architecture, depicted in Figure 2, leverages topological knowledge and physics laws of voltage changes in circuit loops and current flows between connected devices. Specifically, it first embeds the vertex features detailed in Section 4.1 into vertex embeddings and then employs multiple layers that share a common architecture but do not share weights. Each layer consists of 3 steps: neighboring vertex aggregation, loop device aggregation, and merging. Intuitively, the neighboring vertex aggregation step can assimilate embedded information concerning current flows arriving and departing a connection node or device. Meanwhile, the loop device aggregation step aggregates embedded information pertaining to voltage changes across circuit loops. Harmoniously combining these embedded information facilitates learning the topological and physical characteristics of circuit topologies.

The neighbor and loop aggregations generate the aggregated embedding using the multi-head attention (MHA) mechanism in Transformer [17], and the merging step combines these two embeddings into the embedding of each topological graph vertex.

Concretely, we use MHA to denote the attention calculation:

$$MHA(e_i, e_j) = softmax(\frac{e_i \boldsymbol{W}_Q \cdot e_j \boldsymbol{W}_K}{\sqrt{D_k}}) e_j \boldsymbol{W}_V \tag{1}$$

where e_i and e_j are two embeddings, $\boldsymbol{W}_Q \in \mathbb{R}^{D_{in} \times D_k}$, $\boldsymbol{W}_K \in \mathbb{R}^{D_{in} \times D_k}$ and $\boldsymbol{W}_V \in \mathbb{R}^{D_{in} \times D_{out}}$ are the query, key and value matrices, respectively. And D_{in}, D_{out} and D_k are the matrix dimensions.

The neighbor aggregation step generates the neighbor embedding h_{nei} for each vertex n by aggregating the embeddings of neighboring vertices $Nei(n)$ of vertex n using multi-head attention:

$$h^n_{nei} = \sum_{m \in Nei(n)} MHA_{nei}(h^n, h^m) \tag{2}$$

The loop aggregation step generate one loop embedding $h^n_{l_i}$ for each device or terminal port vertex n in each loop l_i using MHA:

$$h^n_{l_i} = \sum_{m \in Loop(l_i)} MHA_l(h^n, h^m) \tag{3}$$

in which $Loop(l_i)$ denotes the set of device and terminal port vertices that are in the loop l_i. Note that the internal connections nodes that do not have voltages do not participate the loop aggregation.

In the merging step, after getting the neighbor embedding h^n_{nei} and all the loop embeddings $h^n_{l_0}, h^n_{l_1}, ...h^n_{l_t}$, where t is the total number of loops that vertex n is involved in, we combine all these embeddings using a weighted summation function:

$$h^n = \sum_{h \in (h_{nei}, h^n_{l_0}, h^n_{l_1}, ...h^n_{l_t})} softmax(\frac{h \boldsymbol{W}_Q \cdot h \boldsymbol{W}_K}{\sqrt{D_k}}) h \tag{4}$$

The 3 steps generate the embedding of each vertex in one layer, and we iterate these processes for multiple layers. Lastly, as a graph-level prediction task, we conduct the mean pooling to generate the representation of the entire circuit and use a multi-layer fully connected network to generate the final performance prediction.

4.4 Sub-network Warm Up

Although circuit loops can capture knowledge essential for accurate prediction, their larger distance leads to over-smoothing and makes training with a cold start difficult. In comparison, the neighbor embedding is more local and easier to learn. Thus, we introduce a specialized learning procedure that starts with training only the aggregation of neighboring vertices via a special masking technique applied to the merging step (omitted due to space limit). Only after the remaining networks have warmed up well, do we remove the masking and add the aggregation of loop devices.

5 EVALUATION

This section evaluates our Graph-Transformer-based Network (GTN) surrogate model for circuit design. We first show the prediction accuracy and speed of GTN. Then, we compare GTN with different baseline surrogate models. To comprehensively assess the advantage of our GTN model in enhancing circuit design, we integrate GTN into a circuit generation framework and compare it with the framework using the software simulator. The experiments are conducted on two servers: one has an Intel Xeon 6238R CPU with two NVIDIA RTX A4000 GPUs. The other has an Intel(R) Xeon(R) Silver 4216 CPU with two NVIDIA RTX A5000 GPUs.

5.1 Experiment Setup

Dataset. We conduct the evaluation on the power converter design task with 5 components. The total number of vertices of the corresponding topological graphs is often larger than 13 with internal connection nodes and 3 terminal ports. The capacitance is set as `C` ($10\mu F$) and the inductance as `L` ($100\mu H$). The frequency is $1MHz$ and the input voltage is $100V$. For external devices, we consider an input resistor of 0.1Ω for `Vin`, and an output resistor of 100Ω and an output capacitor of $10\mu F$ for `Vout`. The duty cycle ranges from 0.1 to 0.9, with a step size of 0.2. We exhaustively generate 959k converter circuits (with 191.8k topologies) that cover almost all those with valid outputs. Note that the circuits with the same topology but different duty cycles are considered distinct. We evaluated their efficiency and output voltage using the NGSPICE-38 simulator as ground truth. Regarding the model training, we randomly sample from the entire dataset at 6 various ratios to construct separate training datasets. The sampling ratios spanned from 0.33% to 1.98%, with an increment of 0.33%. We randomly sample 1% of the dataset as the test dataset and 0.15% of the dataset for validation purposes.

GTN Implementation. For topological graph representation, we convert the circuit topology into a graph and encode vertices into embeddings with dimension of 48, which is the size of D_{in} and D_{out}. Then, we identify loops using `Networkx`. We store the graph and loops in the format of adjacency matrix, which can simplify the mask operations in the training. As for the model, we construct GTN by comprising a total of 4 layers with ReLU activation. For the multi-head attention mechanism within each layer, we configure the number of heads as 4 and D_k as 6. The MLP network for prediction has two hidden layers with sizes of 64 and 16, respectively. In this way, the total number of parameters in the GTN is around 157k.

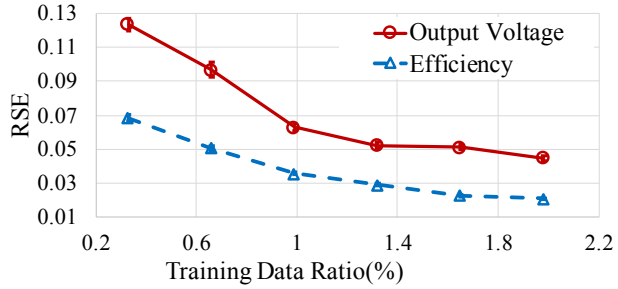
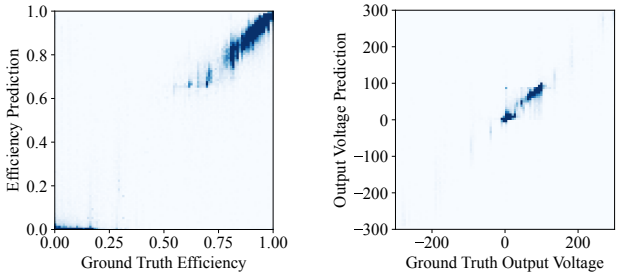


Figure 3: Relative Squared Error with increasing training data ratios. The more training data used, the higher the accuracy.



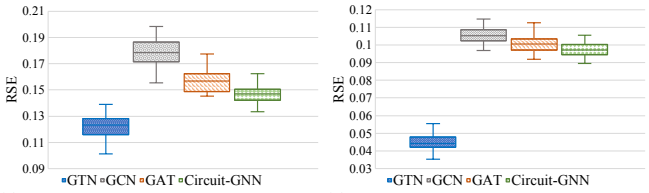
(a) Efficiency (b) Output Voltage

Figure 4: Prediction distribution of the GTN for converter circuits. The more data points cluster around the diagonal line, the closer the prediction to the ground truth, and the better the surrogate model performs.

We train each model for 700 epochs in total and aggregate the loop devices after 450 epochs of sub-network warm up.

Baseline Models. We compare our GTN model with three baseline models: Graph Convolutional Network (GCN), Graph Attention Network (GAT) [19], and Circuit-GNN [21]. GCN updates a node's feature by combining its feature with features from neighboring nodes. Similarly, GAT aims to learn node representations in a graph but dynamically calculates neighboring node weights using an attention mechanism. Different from GCN and GAT, Circuit-GNN [21] is designed to learn to simulate the electromagnetic properties of distributed circuits with a special GNN model that captures information from both nodes and edges across the entire graph.

Experiments of Circuit Generation. We integrated our GTN surrogate model into the framework in [5] that generates custom power converters from design specifications using Monte Carlo Tree Search (MCTS). To demonstrate the advantage of the GTN model, we conduct a comparative analysis with the same framework but incorporate the high-fidelity NGSpice simulator for evaluation. The simulator plays the same role as the surrogate model in the framework. Most experiment settings for circuit generation are the same as the ones in [5]. The target output voltage is set as 50V, equivalent to the 0.5 target conversion ratio in [5]. Major differences include (1) values of the duty cycle are from 0.1 to 0.9, with a step size of 0.2; (2) the circuit evaluation is queried for each topology with a specific duty cycle but not duty cycle sweeping; (3) we query the top 1% of the generated circuits for high fidelity evaluation.



(a) Model RSEs with 0.33% training data. (b) Model RSEs with 1.98% training data.

Figure 5: Comparison of our GTN with baseline models (GCN, GAT, and Circuit-GNN) for output voltage prediction of power converter circuits – all models benefit from more training data, while GTN significantly outperforms baselines.

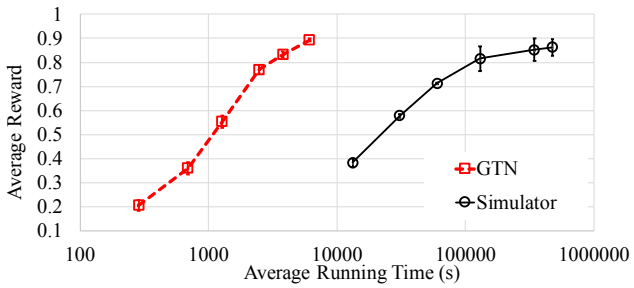


Figure 6: Performance comparison of GTN and simulator in enhancing circuit design. x-axis: the log-running time. y-axis: reward of circuits. *GTN*: generation framework with GTN. *Simulator:* the framework with NGSpice simulator.

5.2 Experiment Results

Performance of GTN Surrogate Model. Figure 3 shows the performance of our model using different training ratios. Trained with 40 seeds, each model calculates the average RSE on the testing set. As the training set expands, the RSEs for efficiency and output voltage monotonically decrease. When we use 1.98% of the dataset for training, the RSE of the efficiency is 2.08% and the output voltage is 4.48%. Figure 4 visualize the distribution of efficiency and output voltage generated by the GTN versus the corresponding ground-truth value on the entire dataset. 4(a) and 4(b) shows that most data points are aligned with the diagonal line. This confirms that our GTN can precisely evaluate the entire dataset and is accurate enough for circuit design.

As for the computation time, we measured the average inference time, that is, the time needed to make a prediction given a circuit. The average inference time is 0.56 sec. over 16k data points. The NGSpice simulation time is 111.7 sec. The GTN achieves over 196× speed-up, highlighting its computational efficiency advantage.

In conclusion, our GTN is a fast-running surrogate model that can learn from a small amount of ground truth circuit evaluation and accurately predict the key performance of unseen circuits.

Comparison with Baselines. We compare the GTN with the baseline models for predicting circuit output voltage. Fig.5(a) shows RSE of 0.33% training data and Fig.5(b) shows the ones with 1.98% training data. We observe that the GTN achieves 16.1% and 53.9% performance improvement respectively compared to Circuit-GNN, and Circuit-GNN performs better than GAT and GAT performs better than GCN. The superiority of GTN and the performance trend remain consistent across different experimental configurations and

datasets. We also compute the RSE when using the state-space averaging method. When setting the voltage prediction of the circuits that cannot be analyzed by state-space averaging as 0 (i.e., disconnect) or 1 (i.e., shortcut), the RSE of state-space averaging is at least 1.0. In comparison, the RSE of our GTN model is only 0.123, achieving a relative improvement of 87.7%.

The results demonstrate that our GTN model, utilizing the multi-head attention mechanism to aggregate both neighbor and loop information, outperforms baseline models focusing solely on neighbor aggregation. Our model can effectively represent nodes within the graph and capture crucial current and voltage knowledge essential for accurate circuit performance prediction.

Accelerating Circuit Generation using GTN. Figure 6 shows the performance of the power converter generation frameworks using different circuit evaluation methods. The reward R for each converter circuit is defined as:

$$R(s, d, V_0) = \eta_{s,d} \cdot 1.1^{-\left(\frac{15(V_{\text{out}}(s,d)-V_0)}{|V_0|}\right)^2}. \quad (5)$$

Recall that $V_{\text{out}}(s, d)$ is the output voltage and $\eta_{s,d}$ denotes the power conversion efficiency. A high-quality converter has high efficiency and V_{out} close to a target output voltage V_0 which is 50V in our experiment. Thus higher R means better quality of a power converter circuit. We run each set 200 times and output the average reward of the generated circuits. Compared with the simulator, the proposed GTN surrogate model significantly reduces the overall design time by up to 98.2% performance without degradation of the quality of the generated power converters. This improvement shows that, during the circuit topology generation process, GTN can swiftly and accurately predict the efficiency and output voltage of the generated circuit. Consequently, our GTN model guarantees that the framework keeps sufficiently exploring the space with better circuit designs. In conclusion, the results demonstrate the effectiveness of the proposed GTN surrogate model that can accurately evaluate the performance of the circuits with a much lower time cost compared to the high-fidelity simulator.

6 CONCLUSION

This work introduces a novel Graph-Transformer-based Network (GTN) surrogate model to accelerate circuit evaluation and topology exploration without sacrificing performance. GTN cultivates voltage information in circuit loops and current information in connected devices, enabling accurate predictions for unseen topologies. For power converter circuit design, GTN outperforms analytical and baseline methods, reduces the design time by up to 98.2%, and offers a powerful tool for automatic topology design. In the future, we plan to enhance GTN by incorporating additional factors, such as layout parasitics, apply GTN to larger-scale power converters, and explore its utility in designing other types of circuit topologies.

ACKNOWLEDGMENTS

This material is based upon work supported in part by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under Award Number DE-SC0024424 and the Advanced Research Projects Agency-Energy (ARPA-E) under Award Number DE-AR0001210, and by the U.S. National Science Foundation under Grant Number CNS-1948457. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

References

[1] L. Alrahis, A. Sengupta, J. Knechtel, S. Patnaik, H. Saleh, B. Mohammad, M. Al-Qutayri, and O. Sinanoglu. 2021. GNN-RE: Graph neural networks for reverse engineering of gate-level netlists. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41, 8 (2021), 2435–2448.

[2] V. Bui, F. Chang, W. Su, M. Wang, Y. Murphey, F. Da Silva, C. Huang, L. Xue, and R. Glatt. 2022. Deep Neural Network-Based Surrogate Model for Optimal Component Sizing of Power Converters Using Deep Reinforcement Learning. *IEEE Access* 10 (2022), 78702–78712.

[3] H. Chen, K. Zhu, M. Liu, X. Tang, N. Sun, and D. Pan. 2021. Universal symmetry constraint extraction for analog and mixed-signal circuits with graph neural networks. In *ACM/IEEE Design Automation Conference (DAC)*. 1243–1248.

[4] Z. Chen, G. Kolhe, S. Rafatirad, C. Lu, S. PD, H. Homayoun, and L. Zhao. 2020. Estimating the circuit de-obfuscation runtime based on graph deep learning. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 358–363.

[5] S. Fan, N. Cao, S. Zhang, J. Li, X. Guo, and X. Zhang. 2021. From Specification to Topology: Automatic Power Converter Design via Reinforcement Learning. In *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. 1–9.

[6] K. Hakhamaneshi, N. Werblun, P. Abbeel, and V. Stojanović. 2019. BagNet: Berkeley analog generator with layout optimizer boosted with deep neural networks. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 1–8.

[7] Z. He, Z. Wang, C. Bail, H. Yang, and B. Yu. 2021. Graph learning-based arithmetic block identification. In *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 1–8.

[8] A. Jackson, N. Ellis, and R. Pilawa-Podgurski. 2022. A Capacitively-Isolated Dual Extended LC-tank Hybrid Switched-Capacitor Converter. In *IEEE Applied Power Electronics Conference and Exposition (APEC)*. 1279–1283.

[9] W. Jin, L. Chen, S. Sadiqbatcha, S. Peng, and S. Tan. 2021. EMGraph: Fast Learning-Based Electromigration Analysis for Multi-Segment Interconnect Using Graph Convolution Networks. In *Design Automation Conference (DAC)*. 919–924.

[10] T. Kavitha, K. Mehlhorn, D. Michail, and K. Paluch. 2008. An algorithm for minimum cycle basis of graphs. *Algorithmica* 52, 3 (2008), 333–349.

[11] Y. Li, Y. Lin, M. Madhusudan, A. Sharma, W. Xu, S. Sapatnekar, R. Harjani, and J. Hu. 2020. A customized graph neural network model for guiding analog IC placement. In *International Conference On Computer Aided Design (ICCAD)*. 1–9.

[12] Y. Lu, S. Pentapati, L. Zhu, K. Samadi, and S. Lim. 2020. TP-GNN: A Graph Neural Network Framework for Tier Partitioning in Monolithic 3D ICs. In *ACM/IEEE Design Automation Conference (DAC)*. 1–6.

[13] T. McConaghy, P. Palmers, M. Steyaert, and G. Gielen. 2011. Trustworthy Genetic Programming-Based Synthesis of Analog Circuit Topologies Using Hierarchical Domain-Specific Building Blocks. *IEEE Transactions on Evolutionary Computation* 15, 4 (2011), 557–570.

[14] W. M. Polivka, P. R.K. Chetty, and R. D. Middlebrook. 1980. State-Space Average modelling of converters with parasitics and storage-time modulation. In *1980 IEEE Power Electronics Specialists Conference*. 119–143.

[15] H. Ren, G. Kokai, W. Turner, and T. Ku. 2020. ParaGraph: Layout parasitics and device parameter prediction using graph neural networks. In *ACM/IEEE Design Automation Conference (DAC)*. 1–6.

[16] B. Shook, P. Bhansali, C. Kashyap, C. Amin, and S. Joshi. 2020. MLParest: Machine Learning based Parasitic Estimation for Custom Circuit Design. In *ACM/IEEE Design Automation Conference (DAC)*. 1–6.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N Gomez, Ł. Kaiser, and I. Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[18] H. Wang, K. Wang, J. Yang, L. Shen, N. Sun, H. Lee, and S. Han. 2020. GCN-RL circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning. In *ACM/IEEE Design Automation Conference (DAC)*. 1–6.

[19] Z. Xie, R. Liang, X. Xu, J. Hu, Y. Duan, and Y. Chen. 2021. Net2: A Graph Attention Network Method Customized for Pre-Placement Net Length Estimation. In *Asia and South Pacific Design Automation Conference (ASP-DAC)*. 671–677.

[20] Z. Xie, H. Ren, B. Khailany, Y. Sheng, S. Santosh, J. Hu, and Y. Chen. 2020. PowerNet: Transferable Dynamic IR Drop Estimation via Maximum Convolutional Neural Network. In *Asia and South Pacific Design Automation Conference (ASP-DAC)*. 13–18.

[21] G. Zhang, H. He, and D. Katabi. 2019. Circuit-GNN: Graph neural networks for distributed circuit design. In *International Conference on Machine Learning*. PMLR, 7364–7373.

[22] Y. Zhang, H. Ren, and B. Khailany. 2020. GRANNITE: Graph Neural Network Inference for Transferable Power Estimation. In *ACM/IEEE Design Automation Conference (DAC)*. 1–6.

[23] Z. Zhao and L. Zhang. 2020. An Automated Topology Synthesis Framework for Analog Integrated Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 12 (2020), 4325–4337.

[24] H. Zhou, W. Jin, and S. Tan. 2020. GridNet: Fast Data-Driven EM-Induced IR Drop Prediction and Localized Fixing for On-Chip Power Grid Networks. In *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. 1–9.