Transfer Learning Framework for 3D Electromagnetic Structures

Oluwaseyi Akinwande*, Sri Laxmi Ganna‡, Rahul Kumar‡, Madhavan Swaminathan*‡
*School of Electrical and Computer Engineering, Georgia Institute of Technology, USA
‡School of Electrical Engineering and Computer Science, Pennsylvania State University, USA

***Abstract* — In the realm of machine-learning-based electronic design automation (EDA), several factors contribute to inefficiency, posing various challenges. Initially, the lack of flexibility in input structures hinders the sharing of information across different circuit topologies. Additionally, substantial costs are incurred in terms of simulation run-times during the data generation process due to the necessity of creating a large training dataset for each circuit topology. To this effect, in this article, we address the dual problem of how to (1) develop a general unified surrogate model that can handle a variety of circuit topologies, and (2) employ previously trained models and adapt them to new models. We provide a formulation for transforming 3D electromagnetic (EM) circuits into versatile circuit graphs, for a variety of topologies, imbued with structural information. The absence of such frameworks represents a gap in machine-learning-based electronic design automation which we fill by providing a set of building blocks to achieve significant improvements in modeling tasks. Lastly, we present a versatile forward modeling framework that allows one to quickly obtain the output response given a set of design parameters. We achieve the overarching goal of reducing the resources needed to create a machine-learning model library for signal integrity (SI) applications in microelectronics packaging.**

***Keywords* — transfer learning, surrogate modeling, electromagnetic modeling, high-speed channels, semiconductor packaging.**

I. Introduction

While advances have been made in the realm of electronic design automation (EDA) with the deployment of tools, flows, and methodologies (TFM) for electromagnetic (EM) modeling across various products, some challenges persist. Particularly, traditional methods like full wave 3D EM solvers, despite being fundamental, are proving to be cost- and time-intensive for large and complex designs. Recently, machine learning (ML) methods have been employed to automate TFM in EDA. ML methods like neural networks (NNs) are able to learn to simulate the EM properties of a circuit and provide the output response quickly. NNs are quite attractive due to their capability to map complex and non-linear relationships between the design space and output response. They have been used for modeling high-speed channels and power delivery networks [1], [2].

In this work, the focus extends beyond achieving adequate fidelity, with respect to low numerical errors and obeying underlying physical laws in NN modeling of circuits [3], [4], to address a new challenge. Like most ML-based EDA models, the models introduced above in [1]–[4] are highly specific to the training data. If one wants to apply the same ML architecture to model a different electronic system, a new training set needs to be generated. As a result, the number of required training samples becomes very large when building a library of ML models. This poses a challenge due to limited resources and expensive EM simulation run-times when creating large datasets.

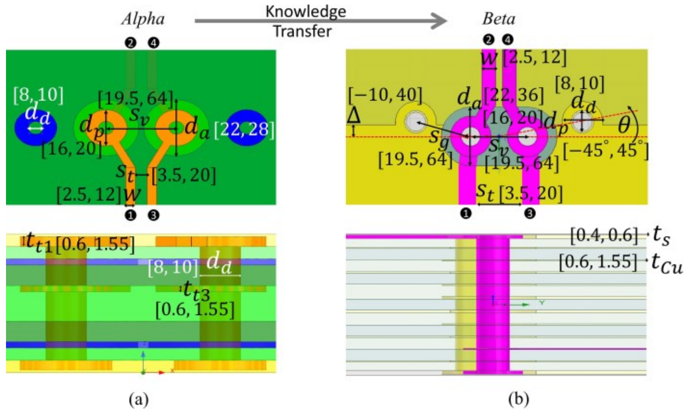


Fig. 1. Can we (1) train a backbone neural net on a single topology (e.g. *Alpha*), (2) capture the underlying electromagnetics (EM), (3) create a unified global structure and (4) adapt to new unseen topologies (e.g. *Beta*)? All length dimensions in mil.

In the realm of ML-based EDA, several factors contribute to inefficiency, posing various challenges. Initially, the lack of flexibility in input structures hinders the sharing of information across different circuit topologies. Additionally, substantial costs are incurred in terms of simulation run-times during the data generation process due to the necessity of creating a large training dataset for each circuit topology. These challenges beg the questions:

1) Can we architect a unified model that can handle variable input structure?
2) Can we seamlessly transfer domain knowledge to new topologies, eliminating the need to initiate the learning process anew?

Fig. 1 provides an illustration for this dual inquiry.

To tackle the variable input structure, we propose using a specialized graph neural network (GNN) architecture that can encode structural information into the data. Next, we consider the EM domain and model each circuit as a graph, the components in each circuit as nodes in a graph, and their EM coupling as edges between the nodes. Furthermore, we develop a pre-trained GNN backbone to serve as a unified model that can adapt across different circuit topologies. Lastly, we investigate *few-shot learning*, where a GNN backbone is adapted to evaluate new unseen topologies.

2024 IEEE/MTT-S International Microwave Symposium - IMS 2024 | 979-8-3503-7504-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/IMS40175.2024.10600177

II. Learning Representation for Multi-Port Devices: Signal Flow Graph (SFG)

Signal flow graph (SFG) is a graphical representation from microwave theory for understanding the interactions and relationships between different components within a system using transmitted and reflected EM waves [5]. Nowadays, coupling coefficient matrix can be regarded as one of the approaches to describe EM interactions through graphical representation [6]. However, coupling coefficients only describe the adjacency and include limited information about components, yielding inaccuracy while synthesizing EM structures [7]. For instance, coupling coefficient method presents great usage for canonical filter synthesis with direct/cross couplings when resonators are identical to each other [8], but shows limitations on representing non-canonical filters [9] or coupled transmission lines.

In contrast, our proposed method utilizes signal flow graph (SFG) that explicitly represents network device ports as nodes and EM interactions between ports as edges. This approach offers a more grounded theoretical foundation compared to manually assigning EM interactions. The SFG topology, combined with the raw geometrical properties of the network device, builds structural information into the architecture of the graph neural network (GNN) model. For instance, the SFG for a two-port network is illustrated in Fig. 3.

III. Forward Modeling With Graph Neural Network (GNN)

A. Definition (Circuit Graph)

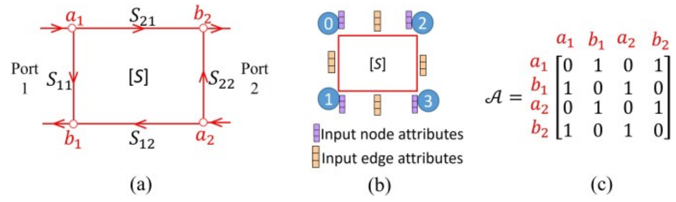
A circuit graph $\mathsf{G} = \{\mathsf{V}, \mathsf{V}_a, \mathsf{E}, \mathsf{E}_a, \mathsf{A}, X\}$, where $\mathsf{V}, \mathsf{V}_a, \mathsf{E} \subseteq \mathsf{V} \times \mathsf{V}, \mathsf{E}_a$ are the sets of nodes, node attributes, edges, edge attributes, respectively, and A, X are the adjacency and raw parameter matrices, respectively.

B. Definition (Adjacency Matrix)

An adjacency matrix, whose entries

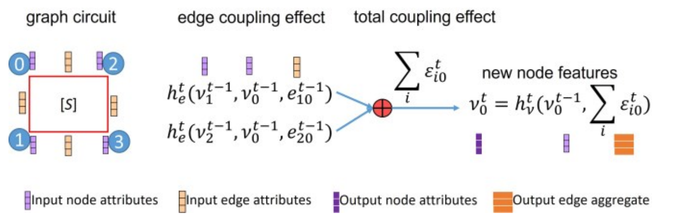
$$\mathsf{A}_{ij} = \delta_{ij} = \begin{cases} 1 & \text{if there is an edge between nodes } i \text{ and } j \\ 0 & \text{otherwise} \end{cases}, \quad (1)$$

represents the connectivity between nodes in V, where δ_{ij} is the Kronecker delta.



$$\mathcal{A} = \begin{array}{c|cccc} & a_1 & b_1 & a_2 & b_2 \\ \hline a_1 & 0 & 1 & 0 & 1 \\ b_1 & 1 & 0 & 1 & 0 \\ a_2 & 0 & 1 & 0 & 1 \\ b_2 & 1 & 0 & 1 & 0 \end{array}$$

Fig. 3. Representation of a two-port network. (a) The signal flow graph [5]. (b) The circuit graph. (c) The adjacency matrix relaying the connectivity.



graph circuit
edge coupling effect
total coupling effect
0
2
1
3
[S]
$h_e^t(v_1^{t-1}, v_0^{t-1}, e_{10}^{t-1})$
$h_e^t(v_2^{t-1}, v_0^{t-1}, e_{20}^{t-1})$
$\sum_i \varepsilon_{i0}^t$
new node features
$v_0^t = h_v^t(v_0^{t-1}, \sum_i \varepsilon_{i0}^t)$
Input node attributes
Input edge attributes
Output node attributes
Output edge aggregate

Fig. 4. Compositions of *message-passing* and *aggregation* for each node in parallel at the t^{th} GNN encoder layer. h^t_e, h^t_v are the edge and node processors composed of multi-layer perceptrons (MLPs).

For example, consider a two-port network device, the circuit graph G is shown in Fig. 3(b), and the adjacency matrix A is given in Fig. 3(c).

The goal of the GNN forward model is to quickly obtain the output response given a set of design parameters. The model maps a given circuit to its output response or n-port network parameters. A high-level block diagram of the forward model is given in Fig. 2.

IV. Inside the GNN Encoder

Consider the GNN encoder block in the flow shown in Fig. 2. The GNN encoder is a k-layer neural net and takes as input the circuit graph G. Each layer in the GNN encoder is a composition of *message-passing* and *aggregation* operations for each node in parallel. This is illustrated in Fig. 4. *Message-passing* is done when each node in a graph circuit G sends EM coupling information about itself to its neighbors which also correspond their coupling information back. To illustrate this mathematically, consider node 0 in Fig. 4, each edge coupling effect on node 0 is computed at the t^{th} GNN

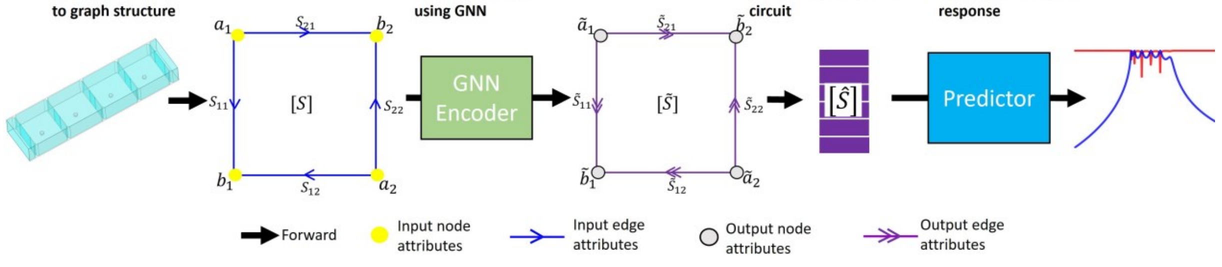


Fig. 2. The GNN forward model uses the circuit graph G with the raw parameter input vector to learn the mapping from the circuit to its frequency response.

Algorithm 1: Transfer Learning

```
Input: Initialization set of all GNN parameters
       $\theta^{(0)} \in dom(g)$, GNN model $g$, raw parameter
       matrix $X$, number of ports $n$ for the multi-port
       device, learning rate $\lambda = 2 \times 10^{-4}$
Output: Backbone GNN parameters $\theta_\alpha$, fine-tuned
        GNN parameters $\theta_\beta$
Map circuit data to graph structured data:
$\{n, X\} \to \mathsf{G} \sim p_\alpha, p_\beta$
Pre-train and create the GNN backbone model:
while $\theta_\alpha$ do not converge do
    Sample a minibatch from $\{\mathsf{G}^{(l)}\}_{l=1}^{N} \sim p_\alpha$
    for $k = 0, 1, 2, \dots, N$ do
        $\hat{y}^{(k)} = g(\mathsf{G}^{(k)}, \theta_\alpha^{(k)})$
        $\mathsf{J}(\hat{y}^{(k)}) = \frac{1}{m}\sum_{i=1}^{m}\left[\left(\Re(\hat{y}_i^{(k)}) - \Re(y_i^{(k)})\right)^2 + \left(\Im(\hat{y}_i^{(k)}) - \Im(y_i^{(k)})\right)^2\right]^{\frac{1}{2}}$
        Update: $\theta_\alpha^{(k)} \leftarrow \theta_\alpha^{(k)} + \lambda \cdot \frac{\partial \mathsf{J}^{(k)}}{\partial \theta_\alpha^{(k)}}$
Adapt pre-trained model: $\theta_\beta \leftarrow \theta_\alpha$
Fine-tune GNN backbone on new unseen circuits:
while $\theta_\beta$ do not converge do
    Sample a minibatch from $\{\mathsf{G}^{(l)}\}_{l=1}^{N} \sim p_\beta$
    for $k = 0, 1, 2, \dots, N$ do
        $\hat{y}^{(k)} = g(\mathsf{G}^{(k)}, \theta_\beta^{(k)})$
        $\mathsf{J}(\hat{y}^{(k)}) = \frac{1}{m}\sum_{i=1}^{m}\left[\left(\Re(\hat{y}_i^{(k)}) - \Re(y_i^{(k)})\right)^2 + \left(\Im(\hat{y}_i^{(k)}) - \Im(y_i^{(k)})\right)^2\right]^{\frac{1}{2}}$
        Update: $\theta_\beta^{(k)} \leftarrow \theta_\beta^{(k)} + \lambda \cdot \frac{\partial \mathsf{J}^{(k)}}{\partial \theta_\beta^{(k)}}$
```

encoder layer as

$$\mathsf{E}_{10}^{t} = h_e^t(v_1^{t-1}, v_0^{t-1}, e_{10}^{t-1}) \quad \text{and} \quad \mathsf{E}_{20}^{t} = h_e^t(v_2^{t-1}, v_0^{t-1}, e_{20}^{t-1}), \tag{2}$$

where $v_i \in \mathsf{V}_\alpha$, $e_i \in \mathsf{E}_\alpha$, and h_e^t is the edge processor, which is composed of a multi-layer perceptron (MLP). *Aggregation* is done when each node in G computes the total coupling information received. Mathematically, consider node 0 in Fig. 4, the total coupling effect received by node 0 at the t^{th} GNN encoder layer is computed as $\bigoplus_{i\in\text{neighbors}} \mathsf{E}_{i0}^{t}$. Lastly, we perform the update for the new node features with encoded information about its understanding of its environment. For node 0, this is given as $v_0^t = h_v^t(v_0^{t-1}, \bigoplus_{i\in\text{neighbors}} \mathsf{E}_{i0}^{t})$, where h_v^t is the node processor, which is composed of an MLP.

V. Application

To demonstrate the effectiveness of the proposed framework, we consider modeling plated through-hole (PTH) vias-in-package which are drilled through layers of printed circuit board (PCB) and conformally plated with copper. They are arranged in ground-signal-signal-ground (GSSG) configuration, with the vias having different transitions between differential microstrip and differential striplines on

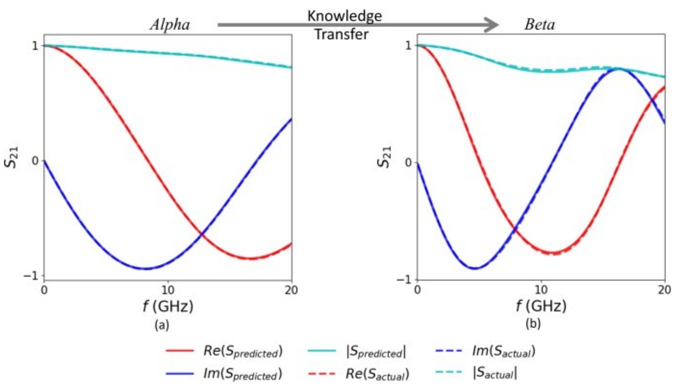


Fig. 5. (a) *Alpha* forward modeling predictions showing S_{21} with the pre-trained GNN backbone model (indicated with solid lines). (b) *Beta* forward modeling predictions showing S_{21} with the fine-tuned GNN backbone model (indicated with solid lines). They are compared with the EM simulation (indicated with dashed lines) for random design tuples in the test set. The real, imaginary and magnitude components of the frequency responses are shown in red, blue and cyan, respectively.

different layers [4]. The package vias and their design parameters are shown in Fig. 1, with *Alpha* having 6 stack-up layers while *Beta* has 12 stack-up layers. We pose the following question: Can we (1) train a backbone neural net on a single topology (e.g., *Alpha*), (2) capture the underlying EM, (3) create a unified global structure, and (4) adapt to new unseen topologies (e.g., *Beta*)?

A. Data Generation and Model Setup

To generate the training data for *Alpha*, we perform a parametric sweep of the design space with the frequency responses being swept from $0.02 - 20$ GHz with steps of 19.98 MHz for each combination of the design parameters. The frequency response of interest is the S_{21} response with 1001 frequency points. Using Latin hypercube sampling (LHS), we determine 1858 samples to be analyzed and solved with Ansys HFSS [10], and we extract their S-parameters. The dataset is divided into 1700 training and 158 testing samples. Next, we implement the transfer learning framework. The outline of the transfer learning framework is shown in Algorithm 1.

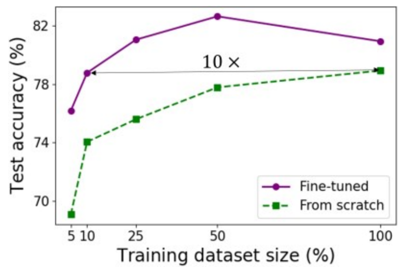
VI. Evaluation Metric

We evaluate our resulting GNN forward models with two metrics, namely: (1) the cumulative accuracy and (2) the mean absolute error (MAE), between the actual response y and the predicted response $\hat{y}$, taken over all the frequency points in the response, respectively defined as

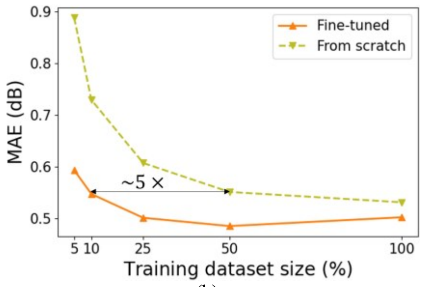
$$\text{Accuracy} = \frac{1}{N}\frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M} \mathbb{1}\left(\left|\frac{|y_i(\omega_j)| - |\hat{y}_i(\omega_j)|}{|y_i(\omega_j)|}\right| \le \delta\right) \tag{3}$$

where $\mathbb{1}(\cdot)$ is the indicator function, δ is the specified tolerance, and

$$\Delta = \frac{1}{N}\frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M} \left|20\log_{10}|y_i(\omega_j)| - 20\log_{10}|\hat{y}_i(\omega_j)|\right|. \tag{4}$$



(a)



(b)

Fig. 6. Fine-tuning capabilities of the pre-trained model to new unseen *Beta* topology. (a) Test accuracy of the fine-tuned GNN backbone model compared with training the GNN model from scratch. For the same test accuracy, the fine-tuned model achieves $10\times$ improvement over the model trained from scratch by leveraging the transfer learning capabilities ($\delta = 0.1$). (b) MAE of the fine-tuned GNN backbone model compared with training the GNN model from scratch. For the same MAE, the fine-tuned model achieves $\sim 5\times$ improvement over the model trained from scratch by leveraging the transfer learning capabilities. For fine-tuning, 100% of the training dataset represents 1000 samples.

VII. Discussion

The pre-training was implemented on *Alpha* to obtain the GNN backbone model and Fig. 5(a) presents the results. Next, we adapt the same GNN backbone model to predict the new unseen topologies, *Beta*, by fine-tuning on a smaller dataset. Fig. 5(b) and Fig. 6 illustrates the results of the transfer learning. We find that the knowledge transferred provided a warm start when evaluating the fine-tuned models even with very small dataset sizes. Lastly, we find that there is a close correlation between the output responses from the GNN forward models and the EM simulator in the results given in Fig. 5. Even though the topological variations in the number of stack-up layers, transitions between different kinds of transmission lines, and the design parameters induce significant EM behavioral change, the model is able to learn the complex representation of the EM coupling during pre-training in order to generalize to new topologies.

VIII. Conclusion

The dual problem of how to (1) develop a general unified surrogate model that can handle a variety of topologies, and (2) employ previously trained models and adapt them to new models, is addressed. We provide a formulation for transforming EM circuits into versatile circuit graphs imbued with structural information. Furthermore, our transfer learning framework swiftly generates output responses from design parameters, reducing resources required for creating ML-model libraries in signal integrity applications.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. CNS 2137259 - Center for Advanced Electronics through Machine Learning (CAEML). The authors thank Xingchen Li for his advice on this paper.

References

[1] Q.-J. Zhang and L. Zhang, “Neural network techniques for high-speed electronic component modeling,” in *Proc. IEEE MTT-S Int. Microw. Workshop Ser. Signal Integrity High-Speed Interconnects*, Feb. 2009, pp. 69–72.
[2] C. M. Schierholz, K. Scharff, and C. Schuster, “Evaluation of neural networks to predict target impedance violations of power delivery networks,” in *Proc. IEEE 28th Conf. Electr. Perform. Electron. Packag. Syst. (EPEPS)*, Oct. 2019, pp. 1–3.
[3] O. Akinwande, O. Waqar Bhatti, K.-Q. Huang, X. Li, and M. Swaminathan, “Surrogate modeling with complex-valued neural nets and its application to design of sub-thz patch antenna-in-package,” in *IEEE MTT-S Int. Microw. Symp. Dig.*, 2023, pp. 423–426.
[4] O. Akinwande, S. Erdogan, R. Kumar, and M. Swaminathan, “Surrogate modeling with complex-valued neural nets for signal integrity applications,” *IEEE Trans. Microw. Theory Techn.*, vol. 72, no. 1, pp. 478–489, 2024.
[5] D. M. Pozar, *Microwave engineering*, 4th ed. USA: John Wiley and Sons, Inc., 2012.
[6] G. Zhang, H. He, and D. Katabi, “Circuit-GNN: Graph neural networks for distributed circuit design,” in *Proc. 36th Int. Conf. Mach. Learn.*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 09–15 Jun 2019, pp. 7364–7373.
[7] J.-S. Hong, *Microstrip Filters for RF/Microwave Applications*, 2nd ed. USA: John Wiley and Sons, Inc., 2011.
[8] R. Cameron, “Advanced coupling matrix synthesis techniques for microwave filters,” *IEEE Trans. Microw. Theory Techn.*, vol. 51, no. 1, pp. 1–10, 2003.
[9] X. Li, X. Jia, S. Erdogan, and M. Swaminathan, “Design and characterization of bandpass filter with multiple zeros on glass interposer for 6g applications,” in *IEEE Radio Wireless Symp. (RWS)*, 2023, pp. 98–100.
[10] Ansys Inc., “Ansys HFSS,” `https://www.ansys.com/`, 2022.