INTEGRATION, the VLSI journal 106 (2026) 102551

Contents lists available at ScienceDirect

Integration, the VLSI Journal

journal homepage: www.elsevier.com/locate/vlsi

Few-shot learning GNN-EQL model with g_m/I_D method for analog integrated circuit design

Xin Xiong, Hongjian Zhou, Pingqiang Zhou *

School of Information Science and Technology, ShanghaiTech University, Shanghai, China

ARTICLE INFO

Keywords:
Analog integrated circuit modeling
Graph neural network
Equation learner network
g_m/I_D method
Few-shot learning

ABSTRACT

Analog integrated circuit design typically involves extensive analytical derivations to evaluate circuit performance metrics. Although SPICE simulations facilitate efficient prediction of these metrics, the simulation process remains time-consuming as the dimensionality of circuit parameters increases. In this article, we propose integrating Graph Neural Networks (GNN) with Equation Learner Networks (EQL), employing them as pretrained models within a limited range of design parameters. Our results demonstrate that datasets constructed using the g_m/I_D method capture design points more efficiently compared to the random sampling of width-to-length (W/L) ratios. Furthermore, experimental validation indicates that the pretrained GNN-EQL model achieves superior performance compared to other pretrained models when the parameter range expands across three different amplifier designs. Finally, our approach significantly reduces the required samples by up to 20X when adapting the pretrained model to broader parameter ranges, compared to training a new model from scratch.

1. Introduction

Analog integrated circuit design is a highly complex and time-consuming task. With the continuous advancement of semiconductor process nodes, the parameters of transistors have become increasingly complicated, making it difficult for traditional long-channel models to accurately capture their physical characteristics. This evolution presents significant challenges for analog integrated circuit design, as it becomes infeasible to directly analyze circuit performance metrics using precise analytical formulas. By leveraging Computer-Aided Design (CAD) tools, designers can quickly obtain the performance metrics of circuits. However, since transistor parameters typically vary over continuous ranges, the resulting design space becomes extremely large. Even minor adjustments to a single parameter often require rerunning simulation software for analysis, making the analog integrated circuit design process both time-consuming and labor-intensive.

To accelerate the process of analog integrated circuit design, researchers have proposed using models to map the relationship between circuit parameters and performance metrics. Researchers have proposed various modeling approaches, such as polynomial-based [1] models and support vector machine [2,3] models. The study by [4] proposed a matrix-based method for circuit performance prediction. [5] employed a genetic algorithm (GA) to optimize polynomial modeling results. [6] further improved model accuracy by integrating the random forests method. With the rapid development of neural networks, the powerful nonlinear fitting capabilities have gained widespread attention and demonstrated significant potential in circuit modeling applications [7,8]. However, such models are typically data-driven and often require a large number of samples to achieve high accuracy. As the range of transistor parameters expands, a larger number of training samples is needed to sufficiently cover the enlarged design space. These samples are usually generated using circuit simulation tools, which significantly increases the computational cost and simulation overhead. To reduce the size of the dataset, various statistical sampling techniques have been introduced to optimize the sampling strategy, such as Latin Hypercube Sampling [9], Monte Carlo Sampling [10], and Bayesian Optimization-Based [11] methods. These techniques mainly focus on selecting representative points in the sample space to improve modeling efficiency. However, they often overlook the underlying physical meaning of the circuit design parameters, which may limit the potential for further improvement in model performance.

To address the aforementioned challenges, this paper adopts the g_m/I_D method for dataset generation, and integrates Graph Neural Network with Equation Learner (GNN-EQL) to model the performance metrics of analog integrated circuits. The GNN-EQL model consists of two main components: a graph neural network for extracting structural features of the circuit, and an equation learning neural network that generates mathematical expressions based on the extracted features

* Corresponding author.
E-mail address: zhoupq@shanghaitech.edu.cn (P. Zhou).

https://doi.org/10.1016/j.vlsi.2025.102551
Received 5 May 2025; Received in revised form 21 July 2025; Accepted 9 September 2025
Available online 16 September 2025

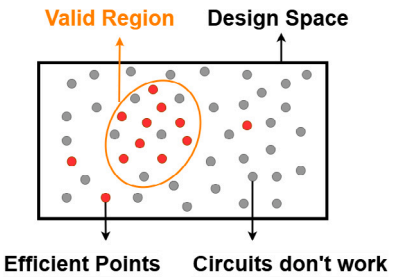


Fig. 1. The illustration of valid region in design space.

for performance prediction. Firstly, the dataset is generated using the g_m/I_D method to train the GNN-EQL model. As shown in Fig. 1, the g_m/I_D method focuses on the normal operating region of circuits and significantly improves the effectiveness of data sampling points. Compared to datasets generated using the conventional W/L random sampling method, the proposed g_m/I_D-based approach has improved the accuracy of the model by nearly 3.5X.

In the generalization stage, the sampling range of transistor parameters is expanded, and datasets covering increasingly larger design spaces are progressively generated using the g_m/I_D method. A zero-shot learning strategy is first employed, with the model trained in a smaller design space and tested on datasets from extended design spaces. Experimental results show that the proposed GNN-EQL model outperforms traditional Graph Neural Networks (GNN) and Multi-Layer Perceptrons (MLP) in terms of prediction accuracy. Subsequently, in a few-shot learning scenario, the pre-trained GNN-EQL model is fine-tuned using small additional samples drawn from the expanded design space. Compared to training a new model from scratch, the fine-tuning approach significantly reduces the required number of training samples by approximately 20X.

The following parts of this paper are organized as follows. In Section 2 we will introduce the definition of the problem and in Section 3 we will introduce how to generate high-quality datasets using the g_m/I_D method and we will demonstrate the GNN-EQL model. In Section 4 we show our results of the experiment on amplifiers and in Section 5 we conclude our work.

2. Preliminary

In this section, we will introduce the preliminary of this paper, including the formulation of the analog integrated circuit modeling problem, and g_m/I_D parameter definition.

2.1. Problem formulation

To build GNN-EQL model using g_m/I_D method for Analog Integrated Circuits, the problem can be described as follows:

$$y = f(\mathbf{x}, \theta) \tag{1}$$

where $\mathbf{x}$ corresponds to the design parameters vector, such as W/L parameters of transistor, y corresponds to the circuit performances metrics, such as gain, bandwidth and phase margin. $f(\mathbf{x}, \theta)$ represents model that map from design parameters vector $\mathbf{x}$ to the circuit performance metrics y under the model parameters vector θ.

We use mean squared error (MSE) to find the optimized value of θ, and it can be summarized as follows:

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} (y_i - f(\mathbf{x_i}, \theta))^2 \tag{2}$$

where N represents the number of training samples.

Let us suppose a vector of design parameters $\mathbf{x}$, and it samples from design space A, which means $\mathbf{x} \subseteq A$. From a larger design space A' which means $A \subseteq A'$, we sample another vector of design parameters $\mathbf{x}'$. The model is trained on vectors $\mathbf{x}$ sampled from the design space A and tested on the vectors $\mathbf{x}'$.

2.2. g_m/I_D method

The g_m/I_D design method was first introduced by [12] in 1996, primarily aimed at low-power analog circuit design. Since then, it has been further developed for modeling circuits with nanoscale transistors [13]. The transconductance g_m represents the gain capability of a transistor. According to the small-signal model, g_m quantifies how effectively the gate–source voltage V_{GS} the drain current I_D when the transistor operates in the saturation region. The expression for g_m can be derived from the drain current equation in the saturation region [14]. Further derivation leads to the expression for the ratio of transconductance g_m to drain current I_D

$$\frac{g_m}{I_D} = \frac{2}{V_{GS} - V_T} = \frac{2}{V_{OV}} \tag{3}$$

where V_{OV} denotes the overdrive voltage. Notably, Eq. (3) excludes process-dependent parameters such as carrier mobility μ and oxide capacitance C_{ox}, which means that the g_m/I_D ratio can be experimentally measured under a given process. The ratio g_m/I_D quantifies the transconductance produced per unit of drain current, thereby reflecting the efficiency of converting current into gain. For a fixed drain current I_D, a higher g_m/I_D indicates greater achievable gain at the same power consumption, making it particularly important in low-power design. Moreover, the g_m/I_D parameter can also effectively serves as an effective indicator of the transistor's operating region [15]. Designers frequently adjust g_m/I_D to strike a balance between power efficiency and performance trade-offs [16]. Several studies have leveraged g_m/I_D methods for analog design optimization [17], as well as transfer learning between different technology nodes [18].

3. Proposed method

The proposed method is shown in Fig. 2. The whole flow of our work is to train the GNN-EQL model based on the g_m/I_D sampling method within a small range of design parameters, and we generalize the model to a larger range of design parameters. Firstly, we transform the circuit netlist into graph notation a nd we use the g_m/I_D method to generate datasets. GNN is used as a feature extractor to aggregate features of the node, and the final output node features are flatten and sent to the EQL network. EQL can distract an analytical expression from the features and predict the circuit performance metrics. Once trained within a small range of design parameters, the GNN-EQL model can be generalized to a larger range. For zero-shot generalization, we directly test the pretrained model on the datasets of large range. For few-shot generalization, we first use some data points to fine-tune the original model, and we test the accuracy of the model compared with training a new model from scratch.

3.1. Generate dataset using g_m/I_D method

The following section will introduce how to generate datasets for training models based on the g_m/I_D method. The process begins with simulations of individual MOS transistors using models provided in the process design kit (PDK). As illustrated in Fig. 3, the curves of I_D/W versus g_m/I_D are generated for various channel lengths L. By selecting different g_m/I_D values along these curves and recording the corresponding data points, a g_m/I_D lookup table can be constructed. This table can be stored for direct use in later steps. Specifically, for a given channel length L and target g_m/I_D value, the corresponding I_D/W ratio can be retrieved from the lookup table. With a known

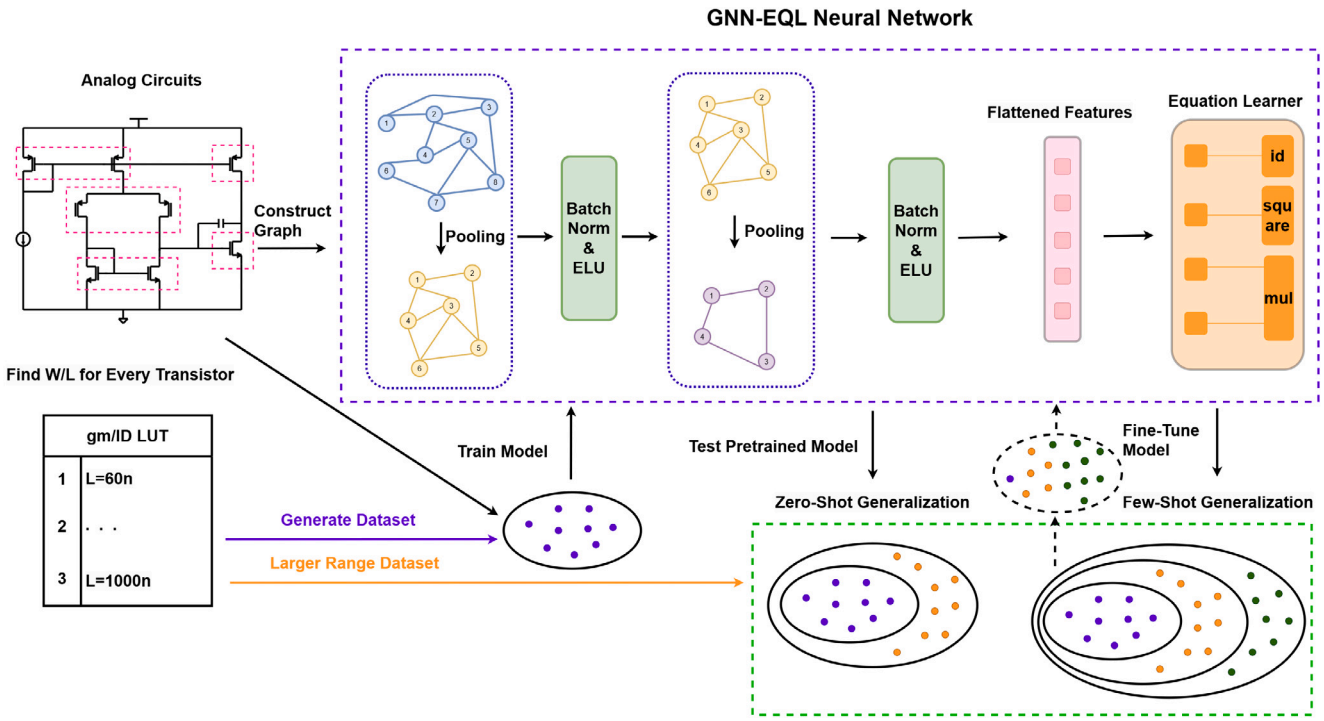
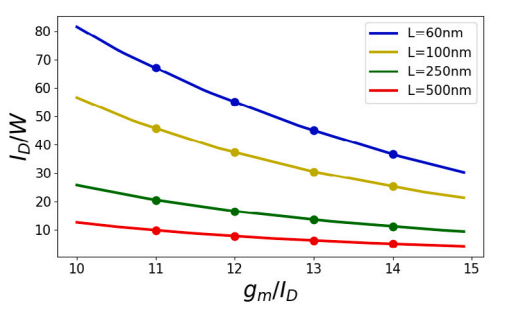


Fig. 2. The overall flow of generalizing GNN-EQL model to a larger range of design parameters.



$Length$ \ $\frac{g_m}{I_D}$	10	11	12	13
$60nm$	67	55	45	36
$100nm$	45	37	30	25
$250nm$	20	16	13	11
$500nm$	10	8	6	5

Fig. 3. Convert the g_m/I_D graph into a lookup table.

drain current I_D, the transistor width $Width$ can be calculated using the following equation:

$$Width = \frac{I_D}{I_D/W} \tag{4}$$

Thus, geometric parameters $Length$ and $Width$ of the transistor can be determined via the lookup table.

Analog integrated circuits are generally composed of multiple functional blocks, such as differential pairs, active loads, and current mirrors. Device mismatch within these blocks can severely degrade circuit performance. To mitigate mismatch effects, transistors within the same functional block should be assigned identical design parameters. After partitioning the circuit into functional modules, the width W and length L of the transistors in each module are determined using the g_m/I_D design method. Once the circuit parameters are defined, performance metrics are extracted through simulation to generate a set of data sample. This process is repeated multiple times, each with varying design parameters, to build a comprehensive dataset.

The simulation data is encoded into an input dataset that conforms to the requirements of graph neural network. At the same time, the circuit netlist file is transformed into a graph representation using an adjacency matrix A. As shown in Fig. 2, a two-stage operational amplifier can be divided into five functional blocks. For the parameters of the transistor in the first functional blocks, it can be encoded as

$$[W, 0, 0, 0, 0, L, 0, 0, 0, 0, 0, 0, 1] \tag{5}$$

$[W, 0, 0, 0, 0]$ is the one-hot encoding vector of transistor width and $[L, 0, 0, 0, 0]$ is the one-hot encoding vector of transistor length. The last two elements represent the type of transistor, $[1, 0]$ corresponds to NMOS, and $[0, 1]$ corresponds to PMOS.

As shown in Fig. 4, the two-stage operational amplifier circuit is abstracted into an undirected graph, where each transistor is represented as a node and the electrical connections between them are represented as edges. To emphasize the direct relationships among transistors, non-transistor elements — such as power supply nodes (VDD), ground (GND), current sources, a nd capacitors — are excluded during graph construction. For example, if the drains of M7 and M5 and the gate of M8 are connected to the same net, this shared connectivity is represented by a hyperedge. In practice, hyperedges are decomposed into pairwise connections to fit the standard graph representation. For a two-stage operational amplifier consisting of 8 transistors, the adjacency matrix $A \in \{0, 1\}^{8\times8}$ encodes the pairwise connectivity.

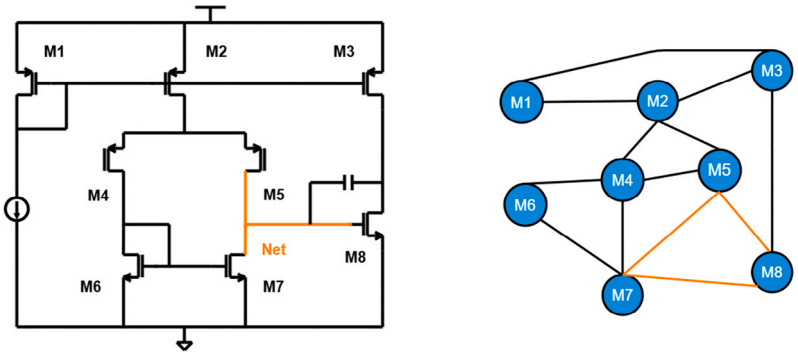


Fig. 4. The graph representation of a two-stage operational amplifier.

3.2. Train the GNN-EQL model

We propose a hybrid modeling framework that integrates Graph Neural Network (GNN) with Equation Learner Neural Network (EQL) to predict the performance metrics of analog integrated circuits. The GNN component processes input graph data, which consists of a node feature matrix H and adjacency matrix A. The EQL component is designed to derive mathematical expressions from the input features generated by the GNN, effectively modeling the performance metrics of the analog circuit.

We use GraphSAGE [19] and DIFFPOOL [20] to extract the structural information of the graph. GraphSAGE can propagate information between layers and aggregate feature vectors from the neighbor nodes, the key idea of GraphSAGE is described as:

$$h_v^k = \sigma\left(W^k \cdot \mathrm{CONCAT}(h_v^{k-1}), \mathrm{AGR}_k(h_u^{k-1})\right) \tag{6}$$

h_v^k is the vector embedding of node v in the k layer. The node u is the neighborhood of the node v. W^k is a weight matrix. AGR_k is aggregation function, it can help node v aggregate information from neighbor nodes. After aggregating the neighborhood vectors, GraphSAGE concatenates them with the current feature vector h_v^{k-1}, and passes through the non-linearity activation function $\sigma(\cdot)$ to obtain a new feature vector h_v^k. DIFFPOOL is used as a pooling operation and can aggregate node features by mapping the nodes to the soft cluster.

The final output features are flattened and sent to the EQL Neural Network. EQL is a totally differential model, and the function can be described as ψ. EQL ith layer is composed of two parts, firstly linear mapping and then non-linear transformations come into action. Linear mapping describes as

$$z^{(i)} = W^{(i)} y^{(i-1)} + b^{(i)} \tag{7}$$

where $W^{(i)}$ is the weight matrix and $b^{(i)}$ is the bias vector $y^{(i-1)}$ is the output of the previous layer and $y^{(0)} = h$, which is the flattened features. The output of linear mapping are performed nonlinear transformations based on different units, unary units such as *square* can help directly get the answer z^2 and binary units such as *mul* can take two variables as input and give the output of multiplication.

EQL can be trained with by minimizing the loss L:

$$L = \frac{1}{N}\sum_{i=1}^{N} \|\psi(x_i) - y_i\|^2 + \lambda \sum_{l=1}^{L} \left|W^{(l)}\right|_1 \tag{8}$$

where loss L consists of L_2 loss and L_1 regularization term. L_1 regularization is used to encourage networks to have sparser connections, and λ is the coefficient of regularization. The regularization term encourages the network to learn simpler and more compact expressions, which can improve generalization and facilitate better extrapolation to unseen parameter ranges. To make a GNN-EQL gradient descent as soon as possible, in GNN layers we add BatchNorm layers to restrict the range of data and we use exponential linear unit (ELU) to introduce some kind of nonlinearity in GNN model. After combining the GNN module with the EQL module, GNN can learn the structure of the circuits, and the EQL model can help to get an analytical function from the features.

Table 1
Typical parameter configuration of two-stage operational amplifier.

Paramete	Current source	Voltage source	Bias voltage	Compensation capacitor	Load capacitance
Typical value	30 μA	1 V	200 mV	500 fF	2 pF

Table 2
The number of transistors, modules and dimensions of feature matrix.

Name	#Transistor	#Modules	Dimensions of feature matrix
Two-stage amplifier	8	5	8 × 12
Folded-cascode amplifier	27	9	27 × 20
Telescope amplifier	14	7	14 × 16

3.3. Few-shot learning GNN-EQL model

In analog integrated circuit design, few-shot learning techniques offer a powerful means of transferring knowledge across different modeling tasks, substantially reducing the amount of data required. For example, [21] demonstrated that a graph neural network pretrained to predict DC voltage of circuits can be effectively transferred to performance prediction tasks for circuits with different topologies, using only a small number of new samples. Similarly, [11] showed that transferring pretrained models to more advanced technology nodes reduces the need for extensive data collection.

Building upon these insights, this paper proposes a performance modeling framework for analog integrated circuits that leverages both zero-shot and few-shot learning strategies. Initially, the model is trained within a limited parameter space and then directly applied to circuit design tasks with an expanded parameter range through zero-shot learning. When the parameter space becomes significantly larger, a few-shot learning approach is adopted, wherein the pretrained model is fine-tuned with a small set of additional samples to adapt to the broader design space. This learning strategy effectively minimizes the need for large-scale training data while maintaining high prediction accuracy.

4. Experiment results

In this section, we present the experimental setup along with the corresponding results and analysis.

4.1. Experimental setup

We use a 65 nm CMOS process node to build three types of operational amplifier circuits, a two-stage amplifier (see Fig. 2), a folded-cascode amplifier, and a telescope amplifier with a common mode feedback (CMFB) circuit (see Fig. 5). Table 1 summarizes the typical

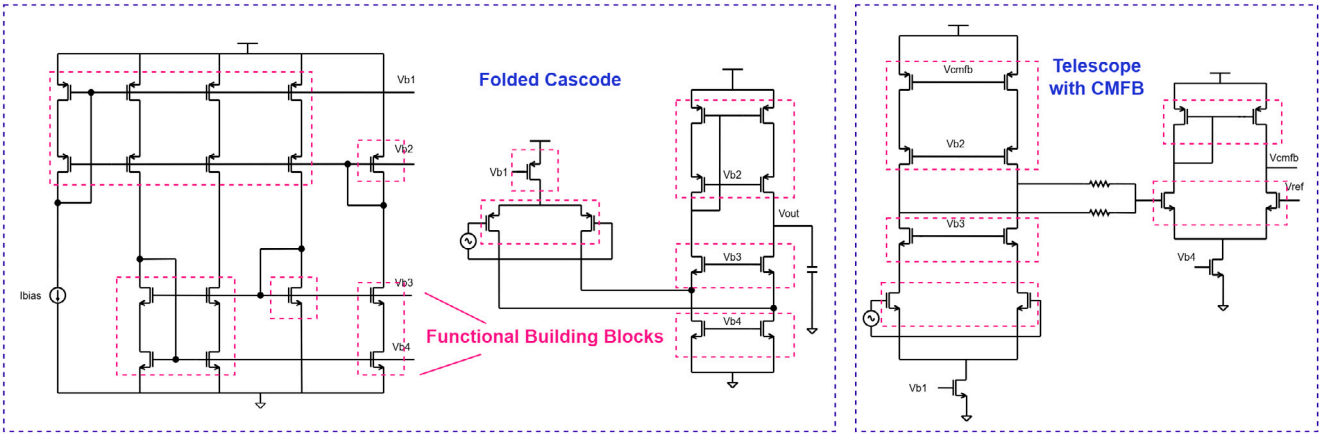


Fig. 5. Folded-Cascode operational amplifier (left) and Telescopic-Cascode operational amplifier with common-mode feedback (right).

Table 3
Model configuration.

Model	Layer configuration		#Trainable parameters
Feature extractor	#nodes	#features	
GNN layers	25	60	9700
	20	30	
	20	1	
Predictor	# hidden layer		
EQL layers	unit: [id, square, mul]		208
MLP with RELU	[64, 64, 32, 16, 1]		10 305

parameter configurations for two-stage operational amplifiers. The fundamental parameter settings for the other two amplifier are aligned with this configuration.

We divide the circuit into modules, and the geometry parameters of transistors in the same modules are the same. As shown in Table 2, the number of transistors and the number of modules that need to be set in three types of operational amplifiers are listed. And we can derive the dimensions of feature matrix. The EQL network consists of two hidden layers. Each layer includes units implementing identity, square, and multiplication operations. Additionally, L_0 regularization is applied to the output expressions of the EQL layers to control the complexity of the resulting symbolic representation. For the MLP model, the L and W of all transistors are directly taken as inputs, with input feature dimensions of 16, 54, and 28, respectively.

The normal neural network framework adopts PyTorch, and the graph neural network related modules are built based on PyTorch Geometric. Table 3 summarizes the configuration of the GNN-EQL model and MLP for the folded-cascode amplifier, the number of trainable parameters varies depending on the dimension of the feature vector.

In terms of sampling settings, the transistor length sampling range is set to [60 nm, 600 nm], the sampling interval is set to 10nm. And the sampling range of the g_m/I_D is set to 10 to 15, with an interval of 0.1. After determining the transistor width using the g_m/I_D method, the width value is rounded to $0.1\mu m$. Considering that the normalization range is unpredictable when the sample space is expanded, we adopt a fixed scaling method, scale the transistor parameters to a range close to [0,1] and maintain the scaling factor unchanged when expanding the sampling space to ensure consistency in the variable range.

We simulate OCEAN script to generate up to 10,000 samples for train and 1000 samples randomly for test. The relative error (RE) metric is used to evaluate the goodness of the model. It defines as:

$$RE = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{\mathbf{y}}_i - \mathbf{y_i}|_2}{|\mathbf{y_i}|_2} \times 100\% \tag{9}$$

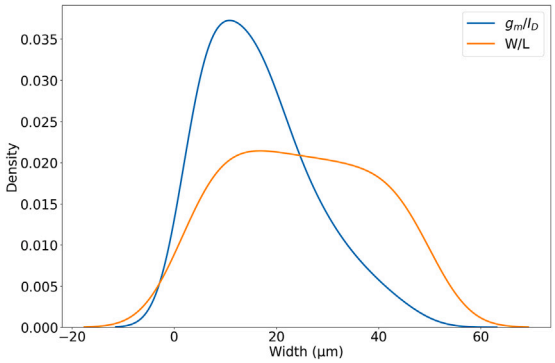


Fig. 6. $Width$ distribution obtained by two different sampling methods.

In the RE formula, N represents the number of samples, $\hat{\mathbf{y}}_i$ is the prediction result vector and $\mathbf{y_i}$ is the simulation result vector, and $|\cdot|_2$ represents the L_2 norm of the vector.

4.2. Comparison of sampling methods

We employ two methods to generate datasets for performance modeling: the W/L random sampling method and the g_m/I_D sampling method. In both approaches, the transistor length is sampled within the range [60 nm, 600 nm]. To ensure a fair comparison, the width sampling range in the W/L method is aligned with the width range derived from the g_m/I_D sampling method. As illustrated in Fig. 6, the x-axis denotes the transistor sampling range, while the y-axis indicates the distribution of transistor widths. It can be observed that the W/L random sampling method yields a relatively uniform distribution across the parameter space. In contrast, the g_m/I_D method produces a denser distribution, with samples more concentrated in the effective design region.

Based on the g_m/I_D sampling method and the W/L random sampling method, a total of 10,000 data samples are generated for each approach. These samples are simulated to obtain circuit metrics, including gain, gain-bandwidth product (GBW), and phase. As illustrated in Fig. 7, the distributions of various performance metrics are compared across the two sampling methods. It can be observed that the dataset generated using the g_m/I_D method results in a more concentrated distribution, effectively capturing the typical operating region.

Datasets generated via the g_m/I_D method, the W/L random sampling method, and the W/L Latin Hypercube Sampling (LHS) method are used to train a GNN-EQL model for gain prediction. Fig. 8 presents a comparison of prediction errors on the test set for these sampling strategies. When tested on the same parameter ranges, the blue curve

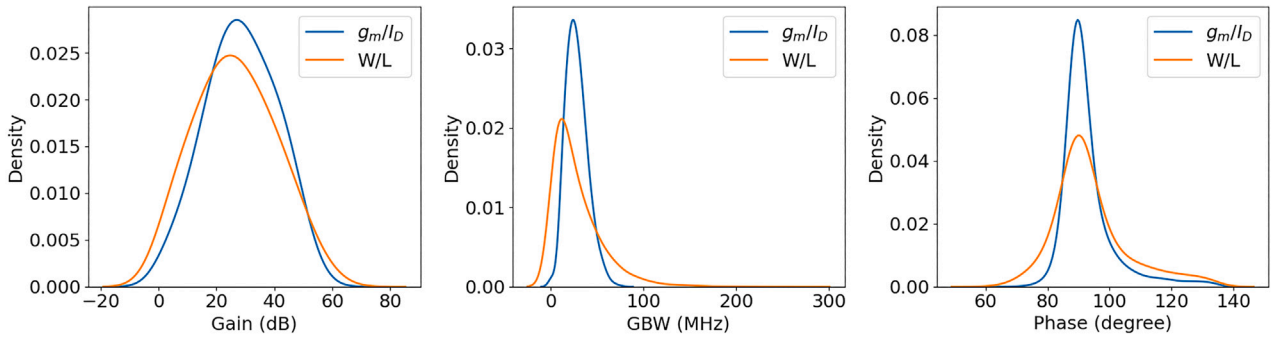


Fig. 7. Distribution of performance specifications obtained by two different sampling methods.

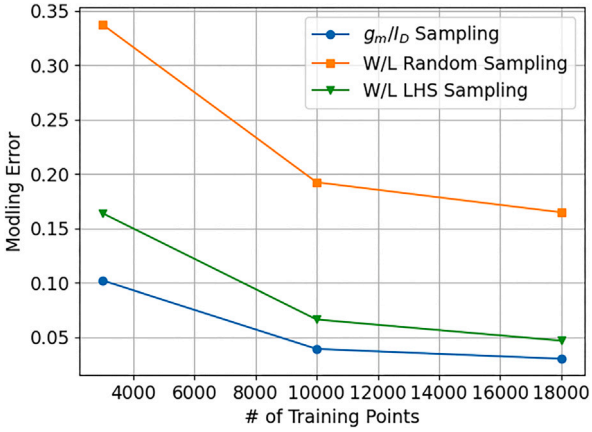


Fig. 8. Comparison between different sampling methods.

(model trained on g_m/I_D sampling method) achieves the lowest modeling error, followed by the green curve (model trained on W/L LHS samples), while the orange curve (model trained on W/L random samples) shows the highest error. Experimental results indicate that the g_m/I_D sampling method more effectively captures the valid and informative regions in the analog design space. As a result, it yields higher-quality training data and enables the model to achieve better predictive performance with fewer training samples, thereby enhancing the efficiency of the modeling process.

4.3. Zero-shot learning different models

The initial sampling range for transistor length is defined as [60 nm, 600 nm]. For each functional block within the operational amplifier circuit, transistor lengths L are randomly sampled within this range, The corresponding transistor widths W are then computed using the g_m/I_D method. For each amplifier configuration, a dataset comprising 10,000 training samples and 1,000 testing samples is generated. Using these datasets, three distinct model architectures — GNN-EQL, standard GNN, and MLP (Multi-Layer Perceptron) — are trained, each dedicated to predicting one of the target performance metrics. Once training is completed, the models are evaluated on the original test set to quantify prediction error, and their parameters are saved for subsequent zero-shot learning experiments.

The next phase involves evaluating the zero-shot generalization ability of the trained models by progressively expanding the parameter space. Specifically, the sampling range for transistor length is extended from the original [60 nm, 600 nm] to wider intervals: [60 nm, 700 nm], [60 nm,
800 nm], [60 nm, 900 nm], and ultimately [60 nm, 1 μm], respectively. For each newly defined interval, 1,000 additional test samples are generated using the same parameter sampling and simulation procedures as described earlier. Taking the interval [60 nm, 700 nm] as an example the extended test dataset enables evaluation of how well the pre-trained models, without any retraining, can predict performance metrics in a broadened design space.

Fig. 9 presents the zero-shot testing results across three amplifiers and three performance metrics. Specifically, subfigures (a) to (c) display results for the folded-cascode operational amplifier, reporting prediction errors on gain, GBW, and phase, respectively. Subfigures (d) TO (f) correspond to the two-stage operational amplifier, while subfigures (g) to (i) depict outcomes for the telescopic cascode operational amplifier. Taking subfigure (a) as an example, it illustrates the gain prediction error for the folded cascode op-amp. The x-axis represents the upper bound of the transistor length sampling range, which is incrementally expanded across five intervals: 600 nm, 700 nm, 800 nm, 900 nm, and 1 μm. The y-axis shows the corresponding model prediction error. Within the initial range [60 nm, 600 nm], three neural network models are trained and evaluated. The GNN-EQL model achieved the lowest prediction error, indicating higher accuracy in the baseline scenario. These trained models were then directly tested on expanded parameter spaces to evaluate their zero-shot generalization capabilities. The results show that while prediction error increases with broader design spaces for all models, the GNN-EQL model exhibits a significantly slower growth in error, consistently outperforming both GNN and MLP models. This highlights the transferability of GNN-EQL model in zero-shot scenarios.

4.4. Few-shot learning over broader range

This section focuses on the gain prediction task for the folded cascode operational amplifier, aiming to evaluate the few-shot learning capability of the proposed GNN-EQL model. As described in the preceding subsection, during the pretraining phase, transistor length parameters within each functional block of the circuit are randomly sampled from the range [60 nm, 600 nm], while the corresponding transistor widths are determined using the g_m/I_D method. In the few-shot learning stage, the parameter space is expanded to [60 nm, 1 μm]. Transistor lengths are uniformly sampled across this extended range, while widths continue to be derived via the g_m/I_D approach, preserving consistency in the design strategy.

Fig. 10 visualizes the parameter distributions under both sampling conditions. The left plot depicts the density distribution of transistor widths obtained from the g_m/I_D method. The blue curve corresponds to the original sampling range [60 nm, 600 nm], while the orange curve represents the extended range [60 nm, 1 μm]. It is evident that as the length sampling range increases, the resulting width distribution broadens accordingly. The right plot illustrates the sampling density of transistor lengths, comparing the uniform distributions obtained from both length intervals. The GNN-EQL model is used to learn the mapping from device parameters to performance metrics across these sampling ranges. By combining graph-based representations with symbolic regression capabilities, GNN-EQL can effectively capture the complex

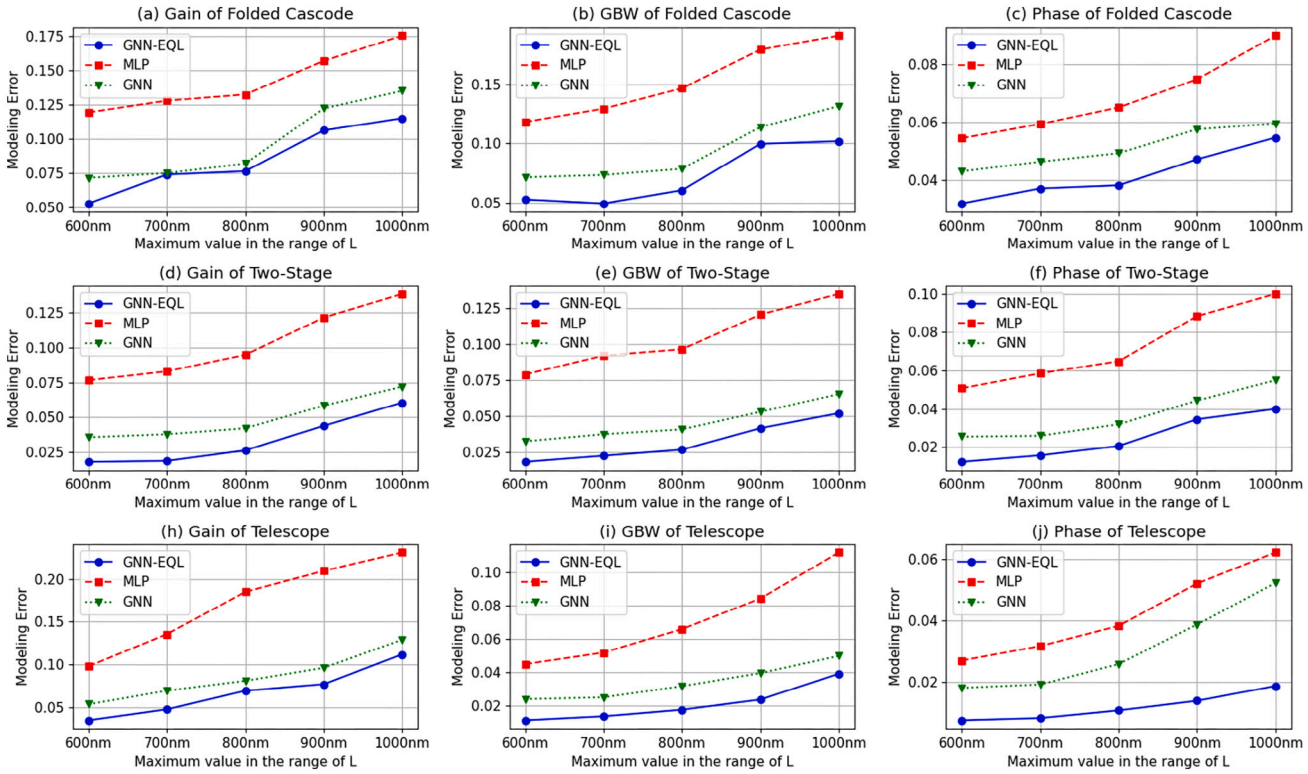


Fig. 9. The modeling error of pretrained model for three different amplifiers.

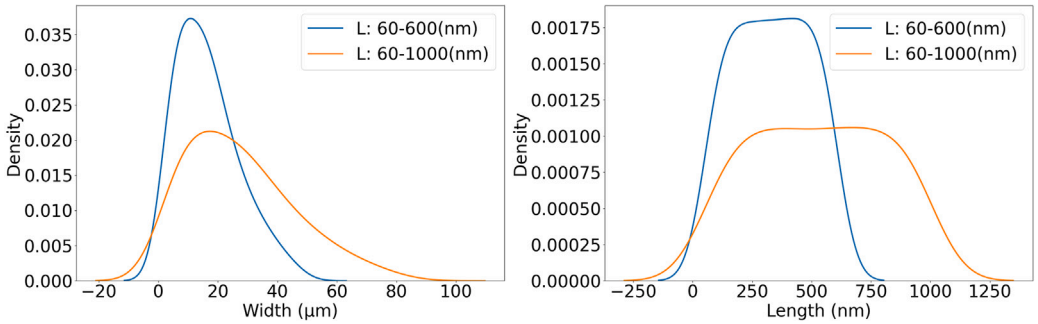


Fig. 10. Distribution of transistor parameters in different sampling spaces.

dependencies between parameters and improve prediction accuracy, even as the parameter space becomes larger and more diverse.

In the gain prediction task for the folded cascode operational amplifier, an initial dataset is first generated within the parameter space [60 nm, 600 nm], and used to train a baseline GNN-EQL model. Subsequently, the parameter space is expanded to [60 nm, 1 μm], and a small subset of samples is drawn from the new design space to perform fine-tuning on the pretrained model. Fig. 11 compares the prediction errors between the fine-tuned model and a model trained from scratch within the expanded parameter space. Experimental results indicate that with as few as 500 fine-tuning samples, the pretrained model achieves prediction accuracy comparable to that of a model trained from scratch using tens of thousands of samples. This outcome demonstrates that few-shot learning not only accelerates the model adaptation process but also reduces data requirements by approximately 20X, highlighting its practicality for analog integrated circuit performance modeling.

For the folded-cascode amplifier circuit, the workflow can first generate candidate configurations by sampling transistor parameters using the g_m/I_D method. The GNN-EQL model is then used to screen and predict feasible design points that meet performance targets. In practice, this approach significantly reduces manual effort, as selecting

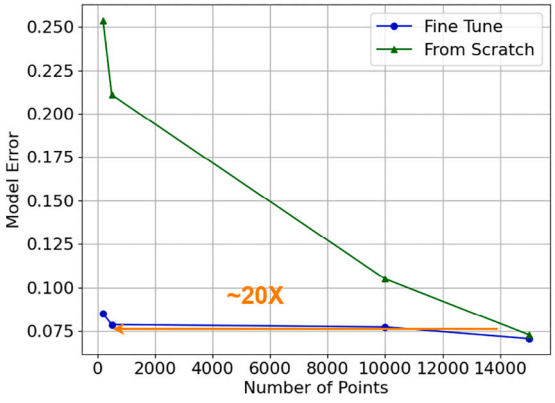


Fig. 11. Comparison between fine-tune model and train from scratch.

and validating comparable designs by hand typically requires several hours of iterative adjustment and simulation.

5. Conclusions

In this paper, we propose the GNN-EQL model for analog integrated circuit modeling, which chooses GNN model as a feature extractor and EQL model as predictor. Based on the g_m/I_D sampling method, we can find that GNN-EQL outperforms traditional model working when directly used on the broader range of datasets. And after few-shot learning, the pretrained GNN-EQL model can have the same accuracy as a new model, which helps to save lots of training data points.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62074100).

Data availability

Data will be made available on request.

References

[1] W. Daems, G. Gielen, W. Sansen, Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 22 (5) (2003) 517–534, http://dx.doi.org/10.1109/TCAD.2003.810742.

[2] T. Kiely, G. Gielen, Performance modeling of analog integrated circuits using least-squares support vector machines, in: Proceedings Design, Automation and Test in Europe Conference and Exhibition, vol. 1, 2004, pp. 448–453, http://dx.doi.org/10.1109/DATE.2004.1268887.

[3] V. Ceperic, A. Baric, Modeling of analog circuits by using support vector regression machines, in: Proceedings of the 2004 11th IEEE International Conference on Electronics, Circuits and Systems, 2004. ICECS 2004, 2004, pp. 391–394, http://dx.doi.org/10.1109/ICECS.2004.1399700.

[4] Almitra Pradhan, Ranga Vemuri, Regression based circuit matrix models for accurate performance estimation of analog circuits, in: 2007 IFIP International Conference on Very Large Scale Integration, 2007, pp. 48–53, http://dx.doi.org/10.1109/VLSISOC.2007.4402471.

[5] Dhruva Ghai, Saraju P. Mohanty, Garima Thakral, Fast analog design optimization using regression-based modeling and genetic algorithm: A nano-CMOS VCO case study, in: International Symposium on Quality Electronic Design, ISQED, 2013, pp. 406–411, http://dx.doi.org/10.1109/ISQED.2013.6523643.

[6] Tsau-Shuan Wu, Cengiz Alkan, Tom W. Chen, Complexity reduction for analog circuit performance models using random forests, in: 2009 17th IFIP International Conference on Very Large Scale Integration, VLSI-SoC, 2009, pp. 29–34, http://dx.doi.org/10.1109/VLSISOC.2009.6041326.

[7] Yaping Li, Yong Wang, Yusong Li, Ranran Zhou, Zhaojun Lin, An artificial neural network assisted optimization system for analog design space exploration, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 39 (10) (2020) 2640–2653, http://dx.doi.org/10.1109/TCAD.2019.2961322.

[8] Mohsen Hassanpourghadi, Shiyu Su, Rezwan A. Rasul, Juzheng Liu, Qiaochu Zhang, Mike Shuo-Wei Chen, Circuit connectivity inspired neural network for analog mixed-signal functional modeling, in: ACM/IEEE Design Automation Conference, DAC, 2021, pp. 505–510.

[9] Zhikai Wang, Jingbo Zhou, Xiaosen Liu, Yan Wang, An efficient transfer learning assisted global optimization scheme for analog/RF circuits, in: 2024 29th Asia and South Pacific Design Automation Conference, ASP-DAC, 2024, pp. 417–422, http://dx.doi.org/10.1109/ASP-DAC58780.2024.10473798.

[10] Hongjian Zhou, Yaguang Li, Xin Xiong, Pingqiang Zhou, A transferable GNN-based multi-corner performance variability modeling for analog ICs, in: 2024 29th Asia and South Pacific Design Automation Conference, ASP-DAC, 2024, pp. 411–416, http://dx.doi.org/10.1109/ASP-DAC58780.2024.10473858.

[11] Juzheng Liu, Mohsen Hassanpourghadi, Qiaochu Zhang, Shiyu Su, Mike Shuo-Wei Chen, Transfer learning with Bayesian optimization-aided sampling for efficient AMS circuit modeling, in: 2020 IEEE/ACM International Conference on Computer Aided Design, ICCAD, 2020, pp. 1–9.

[12] F. Silveira, D. Flandre, P.G.A. Jespers, A g_m/I_D based methodology for the design of CMOS analog circuits and its application to the synthesis of a silicon-on-insulator micropower OTA, IEEE J. Solid-State Circuits 31 (9) (1996) 1314–1319, http://dx.doi.org/10.1109/4.535416.

[13] G. A. Jespers Paul, Boris Murmann, Introduction, in: Systematic Design of Analog CMOS Circuits: Using Pre-Computed Lookup Tables, Cambridge University Press, Cambridge, 2017, pp. xii–xiv.

[14] Razavi, Design Of Analog Cmos Integrated Circuit, McGraw Hill Higher Education, 2017.

[15] Jespers, The g_m/I_D Methodology, a Sizing Tool for Low-Voltage Analog CMOS Circuits: The Semi-Empirical and Compact Model Approaches, Springer, New York, ISBN: 978-0-387-47100-6, 2009, http://dx.doi.org/10.1007/978-0-387-47101-3.

[16] D. M. Binkley, Tradeoffs and optimization in analog CMOS design, in: 2007 14th International Conference on Mixed Design of Integrated Circuits and Systems, 2007, pp. 47–60, http://dx.doi.org/10.1109/MIXDES.2007.4286119.

[17] Minjeong Choi, Youngchang Choi, Kyongsu Lee, Seokhyeong Kang, Reinforcement learning-based analog circuit optimizer using gm/ID for sizing, in: 2023 60th ACM/IEEE Design Automation Conference, DAC, 2023, pp. 1–6, http://dx.doi.org/10.1109/DAC56929.2023.10247739.

[18] Haochang Zhi, Jintao Li, Yun Li, Weiwei Shan, Analog circuit transfer method across technology nodes via transistor behavior, in: Proceedings of the 30th Asia and South Pacific Design Automation Conference, Association for Computing Machinery, New York, NY, USA, ISBN: 9798400706356, 2025, pp. 197–203.

[19] William L. Hamilton, Rex Ying, Jure Leskovec, Inductive representation learning on large graphs, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 1025–1035.

[20] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, Jure Leskovec, Hierarchical graph representation learning with differentiable pooling, 2018, CoRR, abs/1806.08804, URL http://arxiv.org/abs/1806.08804.

[21] Kourosh Hakhamaneshi, Marcel Nassar, Mariano Phielipp, Pieter Abbeel, Vladimir Stojanovic, Pretraining graph neural networks for few-shot analog circuit modeling and design, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 42 (7) (2023) 2163–2173, http://dx.doi.org/10.1109/TCAD.2022.3217421.