

GE-LLM: Graph-Enhanced Large Language Models for Efficient Transistor-Level Circuit Simulation

1st Chao Wang 2nd Dan Niu 3rd Yichao Dong 4th Dekang Zhang 5th Changyin Sun
School of Automation *School of Automation* *School of Automation* *School of Automation* *School of Artificial Intelligence*
Southeast University *Southeast University* *Southeast University* *Southeast University* *Anhui University*
 Nanjing, China Nanjing, China Nanjing, China Nanjing, China Hefei, China
 230258263@seu.edu.cn danniul@163.com 230238503@seu.edu.cn 230258262@seu.edu.cn 3851822122@qq.com

6th Zhou Jin
College of Integrated Circuits
Zhejiang University
 Zhejiang, China
 z.jin@zju.edu.cn

Abstract—DC analysis holds critical importance in nonlinear circuit simulation, providing the essential precondition for transient and AC analyses. While Pseudo-Transient Analysis (PTA) and its variants excel in DC analysis, selecting the optimal PTA method for specific circuits remains challenging. To address this, we propose GE-LLM, a novel framework for optimal PTA method selection, which integrates Graph Neural Networks (GNNs) with Large Language Models (LLMs). The framework first converts circuit netlists into graph representations and employs a GNN-based graph encoder to capture essential circuit topologies. Subsequently, a novel text-graph alignment strategy bridges circuit topologies and textual descriptions, enabling the LLM to effectively comprehend multimodal information. Finally, we introduce a multi-perspective few-shot prompt that mitigates data scarcity by enabling effective in-context learning from limited circuit examples. Experimental results demonstrate that GE-LLM achieves a high selection accuracy of 0.9714 and improves the efficiency of DC analysis, yielding an average speedup of $2.89\times$ in PTA steps (up to $12.14\times$) and $3.45\times$ in Newton-Raphson iterations (up to $30.39\times$) compared to a commercial SPICE-like simulator.

Index Terms—LLMs, GNNs, PTA, DC analysis

I. INTRODUCTION

DC analysis is a fundamental step in circuit verification, serving not only as an initial check for circuit operation but also as an essential prerequisite for various circuit analyzes in SPICE-like simulators [1]. It provides an initial solution for transient analysis and also determines small signal model parameters of nonlinear devices in AC analysis. In DC analysis, several continuation methods, such as Source Stepping Method [2], Pseudo-Transient Analysis (PTA) [3], Gmin Stepping Method [4], and Homotopy Method [5] [6], have been developed to improve convergence. Among these, PTA and its variants, such as Damping PTA (DPTA), Ramping PTA (RPTA), and Compound Element PTA (CEPTA), have been proved to be the quite promising DC analysis algorithms in the industry due to its high continuity and easy implementation [7] [8]. It should be noted that different PTA methods can exhibit significantly varying simulation efficiency for the same circuit. In addition, for a given circuit, there are no effective rules to guide how to select appropriate PTA method. Currently, in PTA method

selection, the ad-hoc manual selections based on the circuit types, topologies and expert experiences are commonly used. However, this selection approach is not automatic and hinders the PTA application in commercial simulators [1].

Recently, Large Language Models (LLMs), with their remarkable abilities in understanding and reasoning over textual information, have been increasingly applied in the Electronic Design Automation (EDA) domain, such as code generation [9] [10] and circuit design [11] [12]. However, directly applying the LLM to DC analysis poses several critical challenges.

- 1) The maximum input token length of the LLM is limited. In addition, in analog circuit simulation, circuit scales vary dramatically, with some industrial-scale circuits containing tens of millions of nodes. Therefore, directly feeding the full netlist into the LLM is impractical. Moreover, although LLMs excel at understanding textual information, they often overlook the rich topological information embedded in circuits. In order to address the limitations of LLMs in handling restricted input token length and their inability to effectively capture topological information, we employ a graph encoder to extract essential topological features from the circuits. However, this approach introduces a core research question: *how to enable the LLM to effectively comprehend and integrate multimodal information, both textual descriptions and graph-level topological representations.*
- 2) Achieving high accuracy in PTA method selection with the LLM requires a diverse and representative set of circuit samples covering various scales and topological structures. However, in practice, available circuits are often limited in both quantity and diversity, posing a significant challenge: *how to enable the LLM to achieve high accuracy in PTA method selection under low-data conditions.*

In this paper, we present GE-LLM, a novel framework for PTA method selection that integrates the LLM with Graph Neural Networks (GNNs), aiming to enhance the efficiency of DC analysis for transistor-level circuits. We propose a netlist-to-graph transformation method and employ the GNN to encode

key topological features. To enable the LLM to effectively utilize multimodal information, a text–graph alignment strategy is developed. Furthermore, to address the challenge posed by limited circuit data, we design a specialized few-shot prompt that embeds representative circuit examples to guide the LLM toward producing accurate PTA method. The learned topological features are meaningful and invariant across tasks, indicating that the approach can generalize beyond DC analysis, such as selecting numerical integration methods in transient analysis. The main contributions of this work are as follows:

- 1) To the best of our knowledge, this work represents the first effort to leverage LLMs for PTA method selection in DC analysis, aiming to enhance simulation efficiency.
- 2) To address the limitation that the LLM cannot effectively comprehend circuit topology, we propose a GNN-based method for extracting essential topological features from circuits. Furthermore, we introduce a multimodal alignment strategy that integrates these structural representations with textual descriptions, significantly enhancing the LLM’s understanding of circuit characteristics.
- 3) To tackle the challenge of insufficient learning by the LLM under limited circuit data, we employ in-context learning. A multi-perspective prompt format is specifically designed for this purpose, which comprehensively incorporates functionality, device composition, and topology features of circuits.
- 4) Extensive experiments demonstrate that our GE-LLM framework achieves a high PTA method selection accuracy of 0.9714. Compared to a commercial SPICE-like simulator, it consistently reduces both the number of PTA steps and NR iterations across most circuits, yielding an average speedup of $2.89\times$ in PTA steps (with a maximum of $12.14\times$) and $3.45\times$ in NR iterations (with a maximum of $30.39\times$).

II. PRELIMINARY

A. PTA Algorithm

For a circuit, its nonlinear algebraic equations can be formulated using Modified Nodal Analysis (MNA) as $\mathbf{F}(\mathbf{x}) = \mathbf{0}$, where $\mathbf{x} = (\mathbf{v}, \mathbf{i})^T \in \mathbb{R}^m$, with $m = N + M$. The vector $\mathbf{v} \in \mathbb{R}^N$ represents node voltages, and $\mathbf{i} \in \mathbb{R}^M$ denotes the branch currents of independent voltage sources [13] [14]. When applying the PTA method by inserting pseudo-elements into the initial DC circuit, the system of ordinary differential equations (ODEs) becomes $\mathbf{P}(\mathbf{x}(t), \dot{\mathbf{x}}(t), t) = \mathbf{F}(\mathbf{x}) + \mathbf{D}\dot{\mathbf{x}}(t) = \mathbf{0}$, where $\dot{\mathbf{x}}(t) = (\dot{\mathbf{v}}(t), \dot{\mathbf{i}}(t))$ denotes the time derivative of \mathbf{x} , and \mathbf{D} represents the incidence matrix of the inserted pseudo-elements. It is important to note that different PTA methods vary in both the insertion locations and the parameter values of these pseudo-elements [15] [16], which ultimately affect the efficiency of the DC simulation.

B. Large Language Models

Large Language Models (LLMs), such as OpenAI’s GPT series [17], Meta’s LLaMA family [18], and DeepSeek’s models [19], have demonstrated remarkable capabilities in natural language understanding, reasoning, and generation across a variety of domains. These models are typically pre-trained on

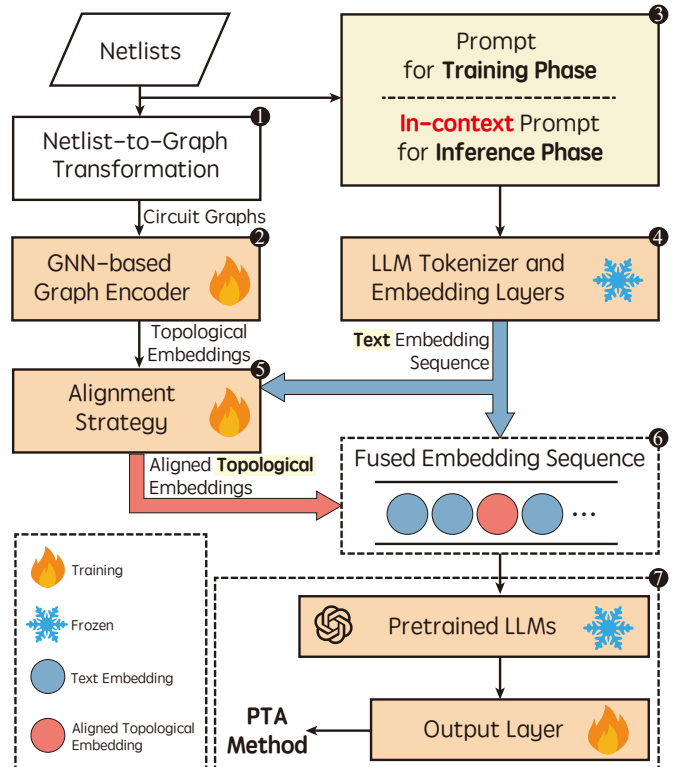


Fig. 1. The Workflow of GE-LLM

large-scale text corpora and can be adapted to downstream tasks through fine-tuning [20] [21]. However, LLMs inherently lack the ability to adequately comprehend and reason about structured data representations such as graphs and topological features [22]. This limitation becomes particularly salient in circuit simulation, where circuit functionality and performance are deeply influenced by complex connectivity.

III. PROPOSED METHOD

A. Overview

In this section, we introduce GE-LLM, a framework integrating LLMs and GNNs to automatically select PTA methods, thereby improving the efficiency of DC analysis. Fig. 1 illustrates the workflow of GE-LLM. Given a netlist dataset, GE-LLM selects the optimal PTA method for unseen circuits.

All circuits in the dataset are simulated using a commercial SPICE-like simulator (C-SPICE) to perform PTA simulations, where the PTA method labels are assigned based on the number of NR iterations. The GE-LLM workflow begins with transforming the circuit netlists into graph representations (Step 1). Then, a GNN-based graph encoder is employed to extract topological embeddings from each circuit graph (Step 2). Meanwhile, we design a specialized prompt that incorporates the circuit name, device composition statistics, and a description of the circuit topology. Specifically, during the inference phase, in-context examples are explicitly incorporated into the prompt covering all PTA method categories, thereby maximizing the diversity of contextual information and facilitating in-context learning (Step 3). The prompt is mapped to a sequence of text

embeddings through the tokenizer and embedding layers of the LLM (**Step 4**). To better integrate the text embeddings with the topological embeddings, an alignment strategy is applied (**Step 5**). The aligned topological embeddings are then inserted into the text embedding sequence (**Step 6**). Finally, the fused embedding sequence is fed into the LLM, and the result is generated through the output layer (**Step 7**).

During training, to reduce computational cost, the LLM parameters are kept frozen, while only the parameters of the graph encoder, alignment module, and output layer are updated.

B. Graph-Based Circuit Topology Extraction for the LLM

1) *Netlist-to-Graph Transformation*: To extract the circuit topological features, we propose a netlist-to-graph transformation framework. Currently, there is still no unified descriptive method to represent transistor-based circuits as graphs that can uniquely represent all possible topological configurations. In this work, each circuit is represented as an undirected graph $G(V, E)$, where the circuit nodes correspond to graph nodes V and the devices correspond to edges E , since the circuit nodes are already explicitly defined in the netlist. Given a graph $G = (V, E)$, each node $n_i \in V$ is assigned a feature vector $\mathbf{h}_{n_i} \in \mathbb{R}^{d_n}$, where d_n represents the counts of connected devices by type, such as resistors, independent voltage and current sources. For edge representation, each edge $e_{ij} \in E$ is associated with a feature vector $\mathbf{h}_{e_{ij}} \in \mathbb{R}^{d_e}$, where d_e denotes the dimension of the edge feature vector and uniquely identifies the corresponding device type. This edge representation captures both the device type and the number of parallel components between nodes.

In circuits, multi-terminal nonlinear devices such as mosfets and bjts substantially increase the topological complexity of circuits. For these devices, the added complexity manifests as hyperedges in the graph representation, which complicate the topological connectivity and make graph construction more challenging. To address this difficulty, we replace all multi-terminal devices with T-equivalent small-signal models, following small-signal circuit principles [23]. As illustrated in Fig. 2, this approach simplifies the circuit graph.

2) *Graph Encoder Based on GNNs*: In this work, we employ the GNN as the graph encoder and propose an effective aggregation strategy named **EdgeAware** to leverage edge information effectively. This approach enhances the expressive ability of the graph encoder by incorporating edge features into the message-passing process. Specifically, edge features are concatenated with the features of their corresponding connected nodes prior to aggregation, thereby ensuring the explicit integration of edge attributes during node feature updating. Formally, the EdgeAware aggregation is defined as follows:

$$\mathbf{h}_{n_i}^{(k)} = \sum_{n_j \in \mathcal{N}(n_i)} f_{\text{agg}} \left(\mathbf{h}_{n_i}^{(k-1)}, \mathbf{M} \left(\mathbf{h}_{n_j}^{(k-1)}, \mathbf{h}_{e_{ij}}, \mathbf{h}_{n_i}^{(k-1)} \right) \right) \quad (1)$$

where k is the current GNN layer, $\mathcal{N}(n_i)$ is the neighborhood of node n_i , f_{agg} is the aggregation function, and \mathbf{M} denotes the concatenation operation followed by a multilayer perceptron (MLP) that reduces dimension from $\mathbb{R}^{2d_n + d_e}$ to \mathbb{R}^{d_n} .

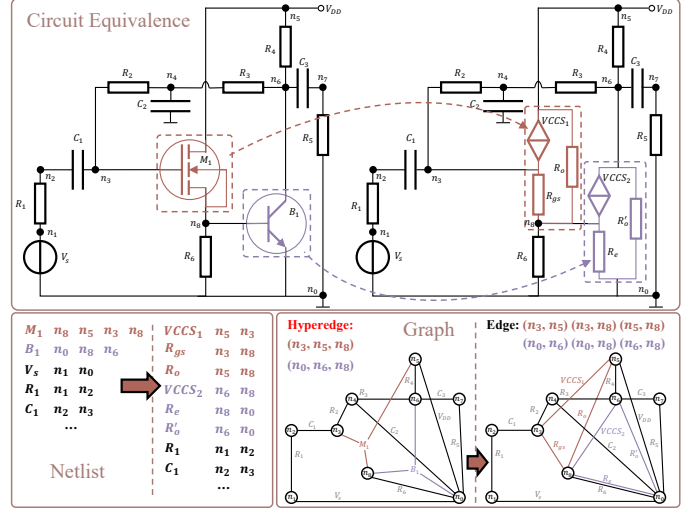


Fig. 2. Netlist-to-Graph Transformation

In graph classification, the readout function plays a crucial role in aggregating node-level embeddings into a graph-level representation. Although traditional readout strategies (sum, max, or mean) are computationally efficient, they inevitably result in partial loss of topological information. To address this limitation, we adopt a gated mechanism that adaptively aggregates node-level embeddings through a learnable weighting scheme. The graph-level embedding \mathbf{h}_g is computed as:

$$\mathbf{h}_g = \sum_{n_i \in V} \sigma(\text{gate}(\mathbf{h}_{n_i}) \cdot \mathbf{h}_{n_i}) \quad (2)$$

where \mathbf{h}_g denotes the final graph-level embedding, σ is a nonlinear activation function, and gate represents a learnable MLP-based gating function. This mechanism adaptively emphasizes informative nodes while suppressing less relevant ones, resulting in more expressive graph-level representations.

C. In-context Learning for the LLM

1) *Prompt Engineering*: In our setting, the number of available transistor circuits is limited, yet their structural complexity remains notably high. These characteristics present substantial challenges for conventional Supervised Fine-Tuning (SFT), which is prone to overfitting and often fails to generalize robustly. To mitigate this issue, we adopt an *in-context learning* paradigm, where task-specific instructions and representative examples are deliberately embedded into the prompts. This approach enables the LLM to effectively adapt to domain-specific tasks under low-data constraints. The designed prompts guide the model to leverage its pretrained knowledge while aligning it with the PTA method selection task, thereby enhancing performance under data scarcity.

The proposed prompt format is illustrated in Fig. 3. It is important to note that the prompt design differs between the training and inference phases. During training, the prompt for each circuit consists solely of a *Task* description and the *Target* instance, without any in-context examples. However, during inference, the few-shot prompts are employed, which

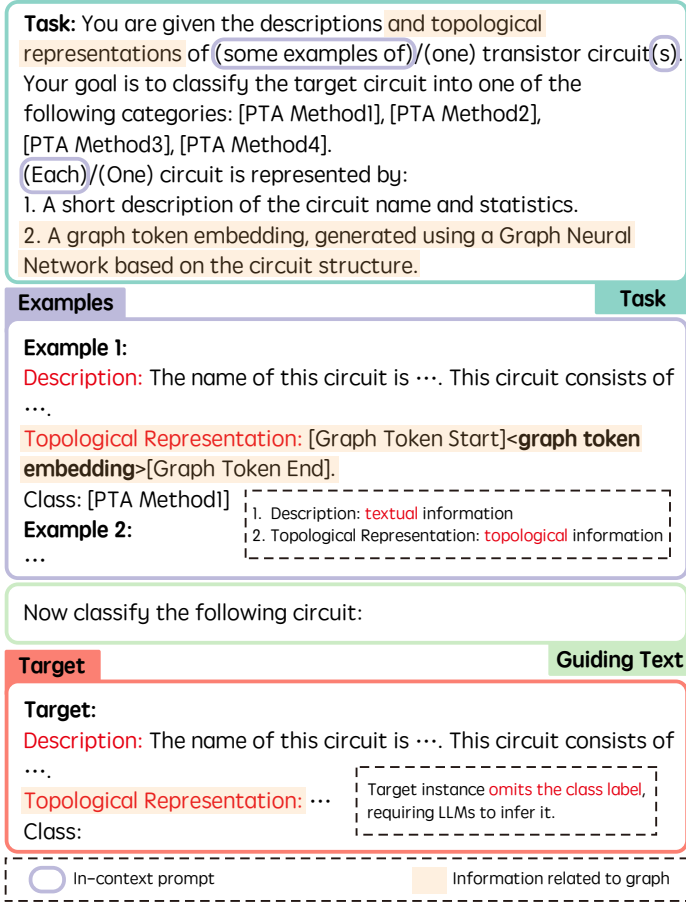


Fig. 3. Prompt Engineering

incorporates multiple examples to facilitate in-context learning without updating the model parameters. Each prompt consists of a *Task* description, several *Example* instances, and one *Target* instance. The *Task* description specifies the overall context and objective of the PTA method selection task. Each instance is represented by two complementary components:

- Description** provides a concise, human-readable summary of the circuit name and basic statistics, e.g., *The name of this circuit is add20. This circuit consists of 42 voltage sources, 3243 capacitors, and 958 mosfets.*
- Topological Representation** is represented as a placeholder in the format *[Graph Token Start][Graph Token End]*, where *[Graph Token Start]* and *[Graph Token End]* are specially designed marker tokens. The circuit topological representation is inserted between these markers.

In each example, the instance is followed by its corresponding **Class** label (one of the four PTA methods). The *Target* instance follows the same format as the examples but omits the **Class** label, requiring the LLM to infer it. The guiding text **Now classify the following circuit** is inserted between example block and target block to indicate the selection step. A critical component is the **selection of examples**. To expose the LLM to the complete set of possible outcomes while preventing data leakage, all in-context examples are exclusively selected

from the training set. Moreover, the examples are collectively chosen to ensure that all four PTA methods are represented. This coverage enables the LLM to learn from representative circuits of each class and capture their distinctive structural and functional characteristics, while strictly maintaining the separation between training and testing data.

Overall, this prompt design provides the LLM with both textual and topological information about the circuits, enabling it to leverage complementary modalities. In-context examples help the LLM learn the mapping between textual and topological information and labels without additional gradient updates.

2) *Alignment of Topology and Text Embeddings:* While the textual description can be directly processed by the LLM tokenizer, the graph-level embedding generated by the GNN is a continuous vector that does not directly align with the embedding space of the LLM text tokens. To enable the LLM to effectively comprehend circuit topological information, a learnable embedding-space mapping method is proposed.

Let $\mathbf{h}_g \in \mathbb{R}^{d_{\text{gnn}}}$ represent the graph-level embedding produced by the GNN, where d_{gnn} denotes the dimension of \mathbf{h}_g . We introduce a trainable linear projection layer:

$$\mathbf{h}_{\text{llm}} = \mathbf{W}\mathbf{h}_g + \mathbf{b}, \quad \mathbf{W} \in \mathbb{R}^{d_{\text{llm}} \times d_{\text{gnn}}} \quad (3)$$

where d_{llm} is the dimension of the LLM token embeddings. The resulting vector \mathbf{h}_{llm} is inserted between the special marker tokens *[Graph Token Start]* and *[Graph Token End]* in the prompt, serving as the **graph token embedding**.

This design preserves the original text embedding sequence while directly incorporating topological embeddings. Furthermore, it maps graph-level embedding into the embedding space of LLM text tokens, enabling cross-modal inference between textual and topological information.

3) *Output Layer:* In this work, we formulate PTA method selection as a graph classification problem. The fused embedding sequence that incorporates both textual and topological information of the circuits are fed into the frozen LLM to obtain graph-level representations. These representations are then flattened and passed through an output layer, implemented as a **linear projection**. Notably, the trainable parameters are primarily concentrated in the graph encoder, the alignment module, and the output layer, constituting only a negligible fraction compared to the parameter count of the LLM.

IV. EXPERIMENTS

A. Experimental Setup

In this work, the proposed GE-LLM is implemented in a commercial SPICE-like simulator (C-SPICE) and evaluated on dozens of benchmark circuits. All experiments are performed on an NVIDIA GeForce RTX 3090 GPU with 24 GB memory. The experimental dataset consists of 164 circuits, where 129 are used for training and the remaining 35 for testing. For each circuit, four PTA methods are evaluated, and the method that achieves the fewest NR iterations is assigned as the label.

In the evaluation, the primary metric is the reduction ratio (speedup) of NR iterations achieved by the selected PTA method relative to the default PTA method used in C-SPICE.

Since the PTA selection task is formulated as a graph classification problem, we also report four standard classification metrics: accuracy, F1-score, precision, and recall. The experiments are organized into three main parts: First, the PTA step and NR iteration speedup achieved by GE-LLM predictions is examined and compared against the default PTA method in C-SPICE. Second, we systematically evaluate the impact of different components, including LLMs (under both few-shot and zero-shot settings), graph encoders, and readout strategies, on PTA method selection performance. Third, ablation studies are conducted to validate the necessity of each module in the multimodal fusion framework. Moreover, we further analyze the representational capacity of the proposed EdgeAware.

B. PTA Step and NR Iteration Speedup Comparison

To evaluate the effectiveness of the proposed GE-LLM framework, extensive experiments are conducted on 35 transistor circuits with results summarized in Table I. For each circuit, the default PTA method in C-SPICE is compared against the GE-LLM predicted method in terms of PTA steps (**#Step**) and NR iterations (**#Iter**). The results show that GE-LLM consistently reduces both metrics across most circuits. On average, it achieves a $2.89\times$ speedup in PTA steps and $3.45\times$ in NR iterations, demonstrating significant acceleration in DC analysis. Maximum improvements reach $12.24\times$ in PTA steps (circuit *smult20*) and $30.39\times$ in NR iterations (circuit *invbjt*).

C. Impact of Different Components

1) Influence of Different LLMs and In-context Learning:

As shown in Table II, all experiments utilize EdgeAware as the graph encoder along with the gate readout strategy. We compare several LLMs, including GPT2-Large [17], LLaMA-2-7B [18], DeepSeek-R1-Distill-Qwen-7B [19], and Vicuna-7B-v1.5 [24], under the 8-shot in-context learning setting. Among these, LLaMA-2-7B achieves the highest performance. To further evaluate the importance of in-context examples, we compare the LLaMA-2-7B model under 8-shot and 0-shot conditions. The 8-shot condition yields significantly higher results, confirming that in-context examples are essential for guiding the LLM.

2) *Graph Encoders Performance Comparison:* The choice of graph encoder is evaluated by comparing EdgeAware, GCN [25], GAT [26], GraphSAGE [27], and GIN [28], as shown in Table III. All experiments presented in Table III are conducted using LLaMA-2-7B with 8-shot prompting. The results show that GCN and GIN perform significantly worse than the others, indicating their limitations in capturing circuit topological relations. In contrast, our EdgeAware achieves the highest performance by explicitly incorporating edge features into the message-passing process. Since edges in our graphs represent circuit components that mediate interactions between nodes, incorporating edge information enables the EdgeAware to capture richer device-level relationships and circuit topologies. These results validate the significant benefit of integrating edge features into the aggregation process.

3) *Different Readout Strategies:* Three commonly used readout strategies (mean, max, and sum) are compared against the proposed gated readout mechanism. As illustrated in Table III,

TABLE I
DC ANALYSIS EFFICIENCY COMPARISONS

Circuits	C-SPICE		GE-LLM (Ours)		Speedup	
	#Step	#Iter	#Step	#Iter	#Step	#Iter
6stageLimAmp	25	80	11	38	2.27×	2.11×
bias	62	362	15	76	4.13×	4.76×
nagle	53	324	19	104	2.79×	3.12×
optrans	203	2508	64	589	3.17×	4.26×
pump	21	22	7	8	3.0×	2.75×
reg0	22	43	8	16	2.75×	2.69×
hussamp	139	860	139	860	1.0×	1.0×
e1480	52	361	25	85	2.08×	4.25×
sqrt	41	164	41	164	1.0×	1.0×
voter	78	451	78	451	1.0×	1.0×
dac	139	988	55	690	2.5×	1.4×
pchip	45	241	45	241	1.0×	1.0×
smult20	612	3821	50	642	12.24×	5.95×
MOSAMP1	34	150	34	150	1.0×	1.0×
UA741	65	426	13	70	5.0×	6.09×
DIFFPAIR	30	128	15	62	2.00×	2.06×
Square_root	33	147	17	93	1.94×	1.58×
UA741PFBVINNEG	29	127	11	42	2.64×	3.02×
D1	98	1089	62	286	1.58×	3.81×
UA741PFBx5	142	1065	68	684	2.09×	1.56×
D21	27	80	9	37	3.0×	2.16×
D20	26	80	12	48	2.17×	1.67×
D30	23	61	10	38	2.3×	1.61×
Inv_RCL	23	51	9	22	2.56×	2.32×
UA741PF	29	127	11	42	2.64×	3.02×
t1	23	45	7	15	3.29×	3.0×
invbjt	99	851	9	28	11.0×	30.39×
HVRefRCL154	22	45	8	16	2.75×	2.81×
UA727-2	42	252	16	82	2.62×	3.07×
tc8	42	221	29	120	1.45×	1.84×
UA727-3	42	252	16	82	2.62×	3.07×
BIPInvchain15	31	95	11	31	2.82×	3.06×
c	97	1099	66	335	1.47×	3.28×
RCL84	26	77	11	32	2.36×	2.41×
tPTAV2	22	43	8	16	2.75×	2.69×
Average					2.89×	3.45×

TABLE II
INFLUENCE OF DIFFERENT LLMs AND IN-CONTEXT LEARNING

LLMs	Accuracy	F1-Score	Precision	Recall
LLaMa-2-7B (8-shot)	0.9714	0.9821	0.9792	0.9799
LLaMa-2-7B (0-shot)	0.8286	0.8125	0.8215	0.7899
Vicuna-7B-v1.5 (8-shot)	0.9429	0.9375	0.9599	0.9464
DeepSeek-R1-Distill-Qwen-7B (8-shot)	0.9714	0.9821	0.9500	0.9630
GPT2-Large (8-shot)	0.8857	0.8952	0.9167	0.8905

it can be observed that max strategy yields the lowest performance. In contrast, the proposed gate strategy achieves the highest accuracy, which adaptively learns the importance of each graph node and performing a weighted aggregation based on node features. It allows the graph encoder to dynamically emphasize nodes that significantly influence circuit simulation.

D. Ablation Experiment

Ablation experiments are conducted to separately evaluate the contribution of the LLM and the graph encoder, as shown in Table IV. When only the LLM is used (without graph features), the model relies solely on textual descriptions of circuits, ignoring topological information. Specifically, all information in prompt related to graph is removed, including mentions in the task description and the topological representation in the examples and the target. This leads to consistently poor performance across all four LLMs, demonstrating that textual

TABLE III
GRAPH ENCODERS PERFORMANCE COMPARISON

GNNs	Readout	Accuracy	F1-Score	Precision	Recall
GCN	gate	0.8286	0.8049	0.8215	0.8033
GAT		0.9429	0.9167	0.9615	0.9337
GraphSage		0.9143	0.8667	0.8792	0.8709
GIN		0.8286	0.8250	0.8538	0.8218
EdgeAware	mean	0.8857	0.9115	0.8292	0.8334
	sum	0.9429	0.9391	0.9615	0.9464
	max	0.7143	0.8006	0.6830	0.6324
	gate	0.9714	0.9821	0.9792	0.9799

TABLE IV
ABLATION EXPERIMENT

Models	Accuracy	F1-Score	Precision	Recall
LLaMa-2-7B (8-shot)	0.4286	0.3485	0.2917	0.2127
Vicuna-7B-v1.5 (8-shot)	0.4286	0.2299	0.3013	0.2558
DeepSeek-R1-Distill-Qwen-7B (8-shot)	0.6000	0.7351	0.5981	0.5462
GPT2-Large (8-shot)	0.4286	0.3185	0.3824	0.2857
EdgeAware	0.7143	0.6875	0.6490	0.6589
GCN	0.6286	0.5846	0.5282	0.5388
GAT	0.6857	0.7183	0.5990	0.6204
GraphSage	0.6857	0.5119	0.5715	0.5365
GIN	0.6286	0.5350	0.5298	0.5277

descriptions alone cannot capture the detailed interconnections and hierarchical dependencies among devices. On the other hand, when only the graph encoder is used (without the LLM), performance remains limited across all GNNs under the gate readout strategy, though the proposed EdgeAware achieves the best results (accuracy 0.7143). The reason is that circuit datasets exhibit substantial variation in scale and structure. Circuits range from small transistor-level designs to large integrated blocks, making it challenging for the GNN alone to generalize effectively. Furthermore, the limited number of layers in GNNs inherently restricts their capacity to fully understand complex and large-scale topological dependencies.

These results confirm the complementary strengths of both modalities: EdgeAware encodes fine-grained structural dependencies, while the LLM supplies high-level functional semantics. Their integration is necessary to achieve the highest performance, as evidenced by the full GE-LLM model.

E. Representational Capacity of EdgeAware

Fig. 4 illustrates the representation capability of the proposed EdgeAware graph encoder. Each scatter point represents the visualization of a graph-level embedding generated by EdgeAware for a test circuit, with distinct colors indicating different labels. The left subplot illustrates the graph-level embeddings output by EdgeAware during the inference phase of GE-LLM. The right subplot, in contrast, displays embeddings obtained when EdgeAware operates independently, which are used directly for PTA method selection.

As can be observed, the scatter points in the right subplot are relatively dispersed, and the boundaries between different categories are unclear, indicating weak class separation. By comparison, the scatter points in the left subplot form tighter clusters with more distinct inter-class boundaries. This contrast

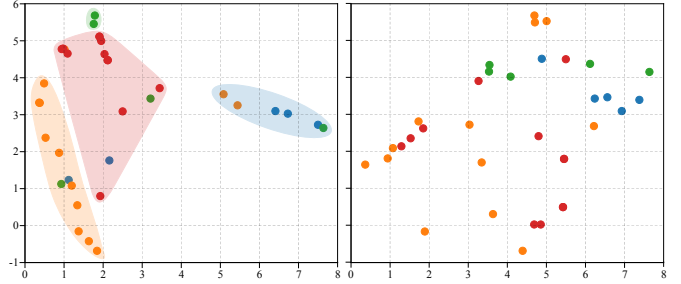


Fig. 4. Representational Capacity of EdgeAware. Each point corresponds to the graph-level embedding of a test circuit, with colors denoting different labels. **Left:** Embeddings generated by EdgeAware in GE-LLM during inference. **Right:** Embeddings produced by EdgeAware operating independently.

suggests that when EdgeAware is used together with the LLM, its output representations become more discriminative, thereby facilitating downstream classification.

These observations further inspire two implications. First, given that the features extracted by the GNN alone are less separable than those produced in conjunction with the LLM, a promising direction is to leverage the LLM to enhance GNN training while using only the GNN during inference. This paradigm could mitigate the GNN’s limitations in capturing circuit topologies while reducing the computational overhead introduced by the LLM during inference. Second, the left subplot further demonstrates that the GNN successfully learns meaningful topological features from the circuit. Importantly, the invariance of circuit topology across different simulation tasks (e.g., the consistency of a circuit topology during both DC and transient analysis) suggests that the proposed framework is not limited to PTA method selection in DC analysis. It can also be extended to other applications, such as selecting numerical integration methods in transient analysis.

V. CONCLUSIONS

This paper proposes GE-LLM, a novel framework that integrates Graph Neural Networks (GNNs) with Large Language Models (LLMs) for PTA method selection in DC analysis. We introduce a netlist-to-graph transformation method and a GNN-based encoder to extract topological features, along with a text-graph alignment strategy to enable multimodal reasoning. To mitigate data scarcity, we design a few-shot prompt incorporating in-context examples covering diverse PTA method categories, facilitating effective generalization under low-data conditions. Experimental results show that GE-LLM achieves an average speedup of $2.89\times$ in PTA steps and $3.45\times$ in NR iterations, with the classification accuracy of 0.9714. Notably, our experiments also indicate that the framework is generalizable and can be extended to other tasks, such as selecting numerical integration methods in transient analysis.

ACKNOWLEDGEMENT

Corresponding author is Dan Niu. This work is partially supported by NSFC (Grant No. 62374031), Natural Science Foundation of Jiangsu Province (BK20240173), NSFC (Grant No. 62204265, 62234010, 92473107, 62331009).

REFERENCES

- [1] J. Deng, K. Batselier, Y. Zhangn, and N. Wong, "An efficient two-level dc operating points finder for transistor circuits," in *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2014.
- [2] T. Najibi, "Continuation methods as applied to circuit simulation," *IEEE Circuits and Devices Magazine*, vol. 5, no. 5, pp. 48–49, 1989.
- [3] C. T. Kelley, D. E. Keyes, "Convergence Analysis of Pseudo-transient Continuation," *SIAM Journal on Numerical Analysis*, vol. 35, no. 2, pp. 508–523, 1998.
- [4] K. Kundert, "The Designer's Guide to SPICE and SPECTRE®.". *Springer Science & Business Media*, 2006.
- [5] K. Yamamura, W. Kuroki, "An Efficient and Globally Convergent Homotopy Method for Finding DC Operating Points of Nonlinear Circuits," in *2006 11th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 8, 2006.
- [6] D. Niu, G. He, Q. Ye, X. Wang, and X. Zhou, "An effective embed algorithm for newton fixed-point homotopy method in MOS transistor circuits," *Int. J. Inf. Electron. Eng.*, vol. 6, no. 3, pp. 32–36, 2016.
- [7] Z. Jin, H. Pei, Y. Dong, X. Jin, X. Wu, W. W. Xing, and D. Niu, "Accelerating Nonlinear DC Circuit Simulation with Reinforcement Learning," in *Proceedings of the 59th ACM/IEEE Design Automation Conference (DAC)*, pp. 619–624, 2022.
- [8] Z. Jin, W. Li, H. Pei, X. Zha, Y. Dong, X. Jin, X. Wu, D. Niu, and W. W. Xing, "ML-PTA: A Two-Stage ML-Enhanced Framework for Accelerating Nonlinear DC Circuit Simulation With Pseudo-Transient Analysis," *IEEE Transactions on Computers*, vol. 74, no. 10, pp. 3319–3331, 2025.
- [9] J. Niu, X. Liu, D. Niu, X. Wang, Z. Jiang, and N. Guan, "Rechisel: Effective Automatic Chisel Code Generation by LLM with Reflection," in *Proceedings of the 62nd ACM/IEEE Design Automation Conference (DAC)*, 2025.
- [10] S. Liu, W. Fang, Y. Lu, J. Wang, Q. Zhang, H. Zhang, and Z. Xie, "RTLCode: Fully Open-Source and Efficient LLM-Assisted RTL Code Generation Technique," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 44, no. 4, pp. 1448–1461, 2025.
- [11] Y. Lai, S. Lee, G. Chen, S. Poddar, M. Hu, D. Z. Pan, and P. Luo, "AnalogCoder: Analog Circuit Design via Training-Free Code Generation," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, vol. 39, no. 1, pp. 379–387, 2025.
- [12] Y. Yin, Y. Wang, B. Xu, and P. Li, "ADO-LLM: Analog Design Bayesian Optimization with In-Context Learning of Large Language Models," in *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, no. 81, pp. 1–9, 2025.
- [13] D. Niu, Y. Dong, Z. Jin, C. Zhang, Q. Li and C. Sun, "OSSP-PTA: An Online Stochastic Stepping Policy for PTA on Reinforcement Learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 11, pp. 4310–4323, 2023.
- [14] Y. Dong, D. Niu, Z. Jin, C. Zhang, Q. Li, and C. Sun, "Adaptive Stepping PTA for DC Analysis Based on Reinforcement Learning," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 1, pp. 266–270, 2023.
- [15] Z. Jin, X. Wu, D. Niu and Y. Inoue, "Effective Implementation and Embedding Algorithms of CEPTA Method for Finding DC Operating Points," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 96, no. 12, pp. 2524–2532, 2013.
- [16] X. Wu, Z. Jin, D. Niu and Y. Inoue, "A PTA Method Using Numerical Integration Algorithms with Artificial Damping for Solving Nonlinear DC Circuits," *Nonlinear Theory and Its Applications*, vol. 5, no. 4, pp. 512–522, 2014.
- [17] OpenAI, et al., "GPT-4 Technical Report," *arXiv preprint arXiv:2303.08774*, 2024.
- [18] A. Pandey, et al., "The Llama 3 Herd of Models," *arXiv preprint arXiv:2407.21783*, 2024.
- [19] DeepSeek-AI, et al., "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning," *arXiv preprint arXiv:2501.12948*, 2025.
- [20] X. Ren, J. Tang, D. Yin, and N. Chawla, and C. Huang, "A Survey of Large Language Models for Graphs," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 6616–6626, 2024.
- [21] R. Ye, C. Zhang, R. Wang, S. Xu, and Y. Zhang, "Language is All a Graph Needs," *Findings of the Association for Computational Linguistics: EACL 2024*, pp. 1955–1973, 2024.
- [22] S. Yu, Y. Wang, R. Li, G. Liu, Y. Shen, S. Ji, et al., "Graph2text or Graph2token: A Perspective of Large Language Models for Graph Learning," *CoRR*, 2025.
- [23] A. S. Sedra, and K. C. Smith, "Microelectronic Circuits 7th edition," *Oxford University Press*, 2007.
- [24] L. Zheng, et al., "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena," *arXiv preprint arXiv:2306.05685*, 2023.
- [25] T. N. Kipf, and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," *arXiv preprint arXiv:1609.02907*, 2017.
- [26] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," in *International Conference on Learning Representations (ICLR)*, 2018.
- [27] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," *arXiv preprint arXiv:1706.02216*, 2018.
- [28] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How Powerful are Graph Neural Networks?," in *International Conference on Learning Representations (ICLR)*, 2019.