

TRACE: A Transferable Framework for Aging-aware Cell Delay Estimation

Muyan Jin¹, Chao Yang², Yunlin Liu², Zejian Cai², Pengpeng Ren^{1*}, Zhigang Ji¹

¹ National Key Laboratory of Science and Technology on Micro/Nano Fabrication

Shanghai Jiao Tong University, China

*Email: pengpengren@sjtu.edu.cn

² T-Head Semiconductor Co., Ltd., China

Abstract—With the continuous scaling of integrated circuits and the miniaturization of semiconductor devices, reliability issues have become increasingly critical. Aging delay prediction based on standard cells is essential for accurate circuit timing analysis. However, the growing diversity of process technology combinations poses significant challenges to the generalization capability of existing AI-based prediction methods. To address this, we propose a novel framework that first employs a graph neural network (GNN) to train a pre-trained model for delay prediction. Building upon this pre-trained model, we introduce a multi-task learning strategy combined with transfer learning to accelerate the training process and enhance adaptability across varying process conditions. This approach culminates in a unified model capable of accurate and efficient post-aging delay estimation. Experiments show that our method accelerates the simulation process by 17,025× compared to SPICE. At the same time, it achieves prediction accuracy comparable to the current state-of-the-art, while requiring 250× less data for training, substantially reducing computational resources.

Index Terms—aging delay, graph neural network, transfer learning, multi-task learning.

I. INTRODUCTION

As transistor dimensions shrink, modern integrated circuits become increasingly susceptible to aging, making long-term reliability a critical concern [1]. Accurate aging prediction is especially important in high-performance computing applications [2], [3], where strict power and performance requirements combine with complex workloads and adaptive voltage/frequency scaling (AVFS). Among major aging mechanisms, Bias Temperature Instability (BTI) [4], [5] and Hot Carrier Injection (HCI) [6] dominate long-term performance degradation. Negative BTI (NBTI) notably impacts PMOS transistors by increasing threshold voltage and slowing switching, which accumulates across cells and circuits, eventually causing timing violations.

While aging-aware SPICE simulations provide high accuracy, their computational cost limits applicability to large circuits. Machine learning methods, such as regression and feedforward networks, offer faster predictions but often fail to capture circuit topology and structural dependencies. In contrast, Graph Neural Networks (GNNs) model circuits as graphs, naturally capturing both local and global aging effects.

This work was supported by the National Natural Science Foundation of China under T2293704, the National Key Research and Development Program of China under 2023YFB4402204 and Beijing Outstanding Young Scientist Program (JWZQ20240101004).

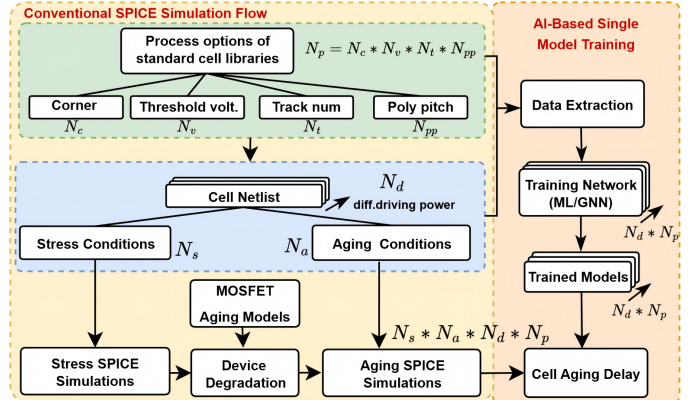


Fig. 1: SPICE simulation flow for conventional aging-aware cell delay and AI-Based Single-Model for prediction of cell aging delay.

By leveraging connectivity and device attributes, GNN-based approaches achieve a better trade-off between accuracy and scalability, providing a practical solution for aging analysis in advanced technologies.

However, existing methods face several limitations that hinder their practicality and accuracy. First, traditional machine learning models often treat circuits as black boxes, neglecting topological information and thus struggling to generalize across diverse cell types and configurations. Second, even graph-based approaches frequently lack refined representations to capture micro-architectural features such as switching activity, load conditions, and localized stress variations, resulting in suboptimal accuracy. Third, most models are trained under restricted operating conditions, leaving them fragile when encountering unseen voltage, temperature, or signal patterns. Finally, few methods support transfer across technologies or cell libraries, typically requiring costly retraining with large datasets when applied to new process nodes, which limits their industrial adoption.

As shown in Fig. 1, traditional methods rely on extensive SPICE simulations, while single-model AI approaches are limited to one specific scenario and require repeated training under complex operating conditions. We introduce TRACE, a unified aging-aware delay prediction framework that overcomes these limitations, offering improvements in accuracy, generalization, and practicality:

- Utilizes a graph neural network (GNN) with enhanced structural representation to model circuit netlists, capturing both local device properties and global interconnect dependencies, significantly outperforming traditional regression methods.
- Incorporates diverse operating conditions to enable robust predictions across previously unseen aging scenarios.
- Applies multi-task learning (MTL) with transfer learning to maximize parameter reuse, improving performance across cell types and aging mechanisms while accelerating adaptation to new technologies with limited data.
- Provides a unified model that directly predicts aging-induced delay shifts under arbitrary conditions, eliminating extensive SPICE characterization and reducing computational overhead without compromising accuracy.

The remainder of the paper is organized as follows: Section II reviews related work, Section III details the framework, Section IV presents experimental results, and Section V concludes the paper.

II. BACKGROUND AND RELATED WORK

A. Aging-Aware Timing Analysis

In conventional aging-aware timing analysis, SPICE-level simulations are commonly employed to evaluate device degradation [7], [8]. This flow typically follows a sequential two-stage process: a stress simulation to model electrical and thermal stress, followed by an aging simulation to estimate long-term parameter shifts (e.g., threshold voltage increase, carrier mobility degradation). While this reflects the physical dependence of aging on prior stress, it poses significant scalability challenges. Since simulations must be repeated for every distinct stress-aging combination, the computational cost becomes prohibitive under diverse PVT (process, voltage & temperature) and workload variations, motivating more efficient learning-based approaches.

B. Graph-Based Approaches for Aging Prediction

Recent advances in deep learning, particularly Graph Neural Networks (GNNs) [9], [10], have shown considerable promise for aging-aware analysis of integrated circuits [11]–[13]. For example, GNN4REL can predict aging-induced delay variations without relying on conventional static timing analysis [14]. Ye [15] proposed a spatio-temporal graph neural network model to capture both structural and temporal aspects of aging. The paper [10] combines graph neural networks with digital logic circuits to study the effects of aging. In addition, heterogeneous Graph Attention Network (GAT) has been employed to identify critical cells, serving as proxies for predicting cell aging-aware delay [16]. In summary, these methods each consider aging effects from their respective perspectives; however, given the complexity of the actual aging process, few studies have proposed a unified model capable of predicting the delays of aged standard cells across different scenarios.

C. Multi-task Learning & Transfer Learning

Multi-task learning (MTL) [17] and transfer learning (TL) [18] are two widely used strategies to enhance generalization in machine learning.

In MTL, a model is jointly optimized over multiple related tasks $\{T_1, T_2, \dots, T_K\}$ with losses $\{\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_K\}$:

$$\min_{\theta} \sum_{k=1}^K \alpha_k \mathcal{L}_k(\theta), \quad (1)$$

where θ denotes shared parameters and α_k the task weights. By exploiting task commonalities, MTL improves efficiency and robustness compared to training each task independently.

TL, in contrast, transfers knowledge from a source task T_S (with parameters θ_S) to a target task T_T , typically through fine-tuning:

$$\min_{\theta_T} \mathcal{L}_T(f(x; \theta_S, \theta_T)), \quad (2)$$

where θ_T are task-specific parameters. TL is particularly effective when the target domain has limited data.

In aging-aware timing analysis, where numerous process–voltage–temperature and workload scenarios exist, combining MTL and TL provides an efficient solution. The joint objective is:

$$\min_{\theta_S, \{\theta_{T_k}\}} \sum_{k=1}^K \alpha_k \mathcal{L}_{T_k}(f(x; \theta_S, \theta_{T_k})), \quad (3)$$

where θ_S represents shared knowledge and $\{\theta_{T_k}\}$ are task-specific parameters. This hybrid paradigm enables parallel transfer across multiple tasks while reducing redundancy, making it particularly suitable for large-scale aging-aware prediction.

III. METHODOLOGY

A. Introduction

In this section, we detail our proposed framework, as illustrated in Fig. 2. The overall workflow consists of two stages: the pretraining stage and the transfer learning stage. Let M

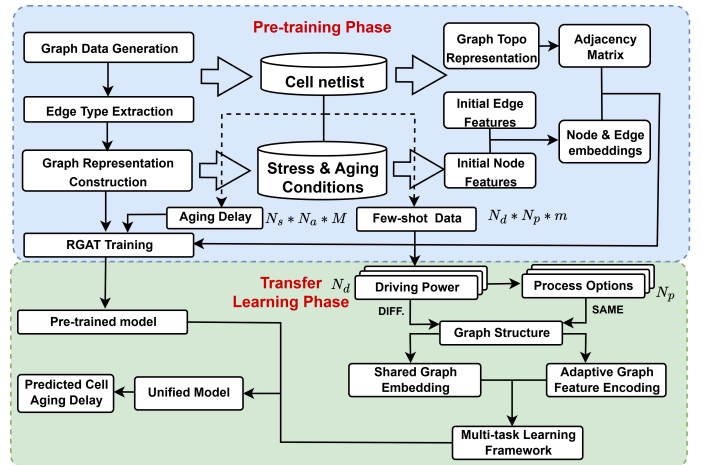


Fig. 2: Pipeline of the proposed framework, including the RGAT-Based Pre-Training and Transfer Learning modules.

denote the sample size required for pre-training the model, and m represent the sample size necessary for a set of transfer tasks. The following subsections discuss each component in detail.

B. Graph Structure

We formulate aging-induced delay prediction as a graph-level regression task at the cell level. In this setting, each standard cell is represented as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where each node $v_i \in \mathcal{V}$ corresponds to a transistor (MOSFET), a supply terminal (VDD/GND), or an input/output pin of the cell. Each directed edge $(v_i, v_j) \in \mathcal{E}$ denotes an electrical connection within the cell, such as a source–drain channel or a net connecting to a pin. The adjacency matrix $A \in \mathbb{R}^{n \times n}$ encodes the cell’s internal topology, where $A_{ij} = 1$ indicates a connection between v_i and v_j , and $A_{ij} = 0$ otherwise.

C. Graph Encoding

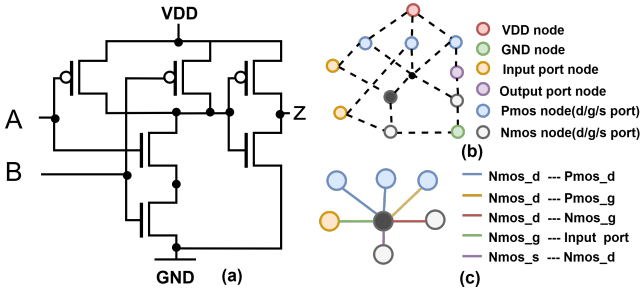


Fig. 3: AND2 Gate Representation: (a) Circuit Schematic, (b) Graph Model, and (c) Edge Feature Demonstration

Instead of encoding node types directly into node features, we use node types only to determine edge types. Specifically, each edge type $r \in \mathcal{R}$ is defined by the node-type pair of its endpoints. Although different cells exhibit diverse topologies, the number of edge types remains finite after this modeling. The adjacency matrix $A \in \mathbb{R}^{n \times n}$ encodes the cell’s internal connectivity. Fig. 3 illustrates the circuit schematic extracted from netlist. Based on their intrinsic characteristics, nodes are categorized into different types. Note that the drain, source, and gate of a MOS transistor are modeled as three distinct ports. Furthermore, each pair of adjacent ports forms an edge.

Each node v_i is initialized with a feature vector $\mathbf{h}_i^{(0)} \in \mathbb{R}^d$, which includes device- or port-specific attributes derived from simulation reports. Detailed feature specifications are provided in Section IV.

To incorporate edge-type information into message passing, we extend the vanilla GAT to a Relational GAT (RGAT) framework. For each edge type $r \in \mathcal{R}$, we associate a learnable weight matrix \mathbf{W}_r . The message aggregation for node i is given by:

$$\mathbf{f}'_i = \sigma \left(\sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_r(i)} \alpha_{ij}^r \mathbf{W}_r \mathbf{f}_j \right), \quad (4)$$

where $\mathcal{N}_r(i)$ denotes the neighbors of node i connected via edges of type r , and α_{ij}^r is the attention coefficient corresponding to relation r .

The attention coefficient is computed as:

$$\alpha_{ij}^r = \frac{\exp(\text{LeakyReLU}(\mathbf{a}_r^\top [\mathbf{W}_r \mathbf{f}_i \parallel \mathbf{W}_r \mathbf{f}_j]))}{\sum_{k \in \mathcal{N}_r(i)} \exp(\text{LeakyReLU}(\mathbf{a}_r^\top [\mathbf{W}_r \mathbf{f}_i \parallel \mathbf{W}_r \mathbf{f}_k]))}, \quad (5)$$

where \mathbf{a}_r is a learnable attention vector specific to relation r , and \parallel denotes vector concatenation.

After several RGAT layers, we aggregate node embeddings into a graph-level representation using a readout function (e.g., mean or sum pooling):

$$\mathbf{h}_G = \text{Readout}(\{\mathbf{f}'_i \mid v_i \in \mathcal{V}\}), \quad (6)$$

which is then fed into a regression head (MLP) to predict the cell aging delay.

D. Multi-task and Transfer Learning

After pretraining the cell-level RGAT model on a collection of conditions, we propose a hierarchical transfer learning framework to address heterogeneous tasks, where each target task corresponds to a specific circuit cell structure. The framework combines multi-task learning with three key components: parameter sharing, feature-wise modulation, and task-specific adaptation.

Formally, a target task is denoted as $\mathcal{T}_{i,j}$, where $i \in \{1, \dots, M\}$ indexes the cell structure type, and $j \in \{1, \dots, N_i\}$ indexes the task within structure i . For each structure type i , the corresponding task group is defined as:

$$\mathcal{S}_i = \{\mathcal{T}_{i,1}, \dots, \mathcal{T}_{i,N_i}\}, \quad (7)$$

and the complete task set is partitioned as:

$$\mathcal{S} = \{\mathcal{S}_1, \dots, \mathcal{S}_M\}. \quad (8)$$

Tasks within the same group \mathcal{S}_i share an identical graph topology but differ in global feature distributions, such as process corner information, which is constant across all MOS transistors within a standard cell. Tasks across different groups correspond to distinct cell topologies, extracted directly from the cell-level netlists.

The transfer learning procedure is hierarchical. *Intra-group learning* focuses on tasks within the same structure group, leveraging shared structural information via parameter sharing and adapting to task-specific global conditions through feature-wise modulation. *Inter-group adaptation* addresses tasks across different structure groups by introducing task-specific mechanisms that capture structural variations not represented by the shared encoder, enabling effective knowledge transfer across heterogeneous graph structures.

1) Intra-group Learning:

a) Parameter Sharing within Structure Groups: Let $\mathbf{X}_{i,j}$ denote the input features of task $\mathcal{T}_{i,j} \in \mathcal{S}_i$. Tasks within the same structure group share a common RGAT encoder f_θ to exploit their identical graph topology:

$$\mathbf{H}_{i,j} = f_\theta(\mathbf{X}_{i,j}), \quad \forall \mathcal{T}_{i,j} \in \mathcal{S}_i, \quad (9)$$

where $\mathbf{H}_{i,j}$ represents the node-level embeddings of task $\mathcal{T}_{i,j}$. Sharing the parameters θ across tasks in \mathcal{S}_i enforces a consistent representation of the shared graph structure, promotes knowledge transfer, and mitigates overfitting on individual tasks.

b) *Feature-wise Modulation (FiLM)*: Within each structure group \mathcal{S}_i , tasks share a common graph topology but may differ in global features such as operating conditions. We apply feature-wise linear modulation on the shared embeddings:

$$\tilde{\mathbf{H}}_{i,j} = \Gamma(\mathbf{g}_{i,j}) \odot \mathbf{H}_{i,j} + B(\mathbf{g}_{i,j}), \quad (10)$$

where $\mathbf{H}_{i,j}$ are node embeddings from the shared encoder, $\mathbf{g}_{i,j}$ are task-specific global features, and Γ, B are learnable mappings. This adapts the shared representation to task-specific conditions while preserving structural consistency.

2) *Inter-group Adaptation*: For tasks across different structure groups, we introduce a structure-aware adapter with global and local branches.

Global branch: Modulates embeddings using a graph-level representation $\mathbf{s}_{i,j}$:

$$g_{\phi}^{\text{global}}(\tilde{\mathbf{H}}_{i,j}, \mathbf{s}_{i,j}) = \text{MLP}^{\text{global}}([\tilde{\mathbf{H}}_{i,j}; \mathbf{s}_{i,j}]). \quad (11)$$

Local branch: Exploits neighborhood structure via attention:

$$g_{\phi}^{\text{local}}(\tilde{\mathbf{H}}_{i,j}, \mathbf{A}_{i,j}) = \sigma\left(\sum_{k \in \mathcal{N}(v)} a_{vk} W \tilde{\mathbf{h}}_k\right), \quad (12)$$

where a_{vk} is the attention coefficient, W a learnable weight, and σ a nonlinear activation.

Fusion: Outputs combined with a learnable weight α :

$$\mathbf{H}_{i,j}^{\text{adapt}} = \alpha \cdot g_{\phi}^{\text{global}} + (1 - \alpha) \cdot g_{\phi}^{\text{local}}. \quad (13)$$

The adapted embeddings are passed to the shared readout h_{ϕ} to produce task-specific outputs:

$$\mathbf{Z}_{i,j} = h_{\phi}(\mathbf{H}_{i,j}^{\text{adapt}}), \quad (14)$$

allowing effective inter-group knowledge transfer while retaining shared encoder benefits.

3) *Prediction and Loss Function*: For each task $\mathcal{T}_{i,j}$, the model predicts a multi-dimensional delay vector for the target cell. Specifically, the node embeddings $\mathbf{H}_{i,j}^{\text{adapt}}$ produced by the hierarchical transfer framework are aggregated via a readout function, and then mapped through a multi-layer perceptron (MLP) to obtain the final graph-level prediction:

$$\hat{\mathbf{y}}_{i,j} = \text{MLP}(h_{\phi}(\mathbf{H}_{i,j}^{\text{adapt}})) \in \mathbb{R}^{1 \times n}, \quad (15)$$

where n denotes the number of delay dimensions to predict.

Since the different delay dimensions may vary by several orders of magnitude, we adopt the Huber loss to balance sensitivity to outliers and robustness across scales. Let $\mathbf{y}_{i,j} \in \mathbb{R}^{1 \times n}$ be the ground-truth delay vector for task $\mathcal{T}_{i,j}$, and let δ be the Huber threshold, then the Huber loss for a single task is defined as

$$\mathcal{L}_{\delta}(\hat{\mathbf{y}}_{i,j}, \mathbf{y}_{i,j}) = \sum_{k=1}^n \begin{cases} \frac{1}{2}(\hat{y}_{i,j}^{(k)} - y_{i,j}^{(k)})^2, & \text{if } |\hat{y}_{i,j}^{(k)} - y_{i,j}^{(k)}| \leq \delta, \\ \delta |\hat{y}_{i,j}^{(k)} - y_{i,j}^{(k)}| - \frac{1}{2}\delta^2, & \text{otherwise.} \end{cases} \quad (16)$$

Finally, the overall loss for all tasks in all structure groups is computed by averaging over tasks:

$$\mathcal{L} = \frac{1}{\sum_{i=1}^M N_i} \sum_{i=1}^M \sum_{j=1}^{N_i} \mathcal{L}_{\delta}(\hat{\mathbf{y}}_{i,j}, \mathbf{y}_{i,j}). \quad (17)$$

This formulation ensures that the model effectively learns multi-dimensional delay predictions while being robust to varying scales across different delay components.

The overall procedure combines shared RGAT encoders, adaptive encoders, multi-task learning, and Huber loss. Below is a high-level pseudocode of our framework:

Algorithm 1 Pseudocode of TRACE

- 1: **Input**: Pretrained RGAT f_{θ} , datasets $\{(G_{i,j}, \mathbf{y}_{i,j})\}$, Huber threshold δ , max epochs O
 - 2: Initialize adapters $\{g_{\phi_{i,j}}\}$ for all tasks
 - 3: **for** epoch $o = 1$ to O **do**
 - 4: **for** structure group $i = 1$ to M **do**
 - 5: **for** task $j \in \mathcal{S}_i$ **do**
 - 6: Encode nodes via RGAT: $\mathbf{H}_{i,j}$
 - 7: Apply FiLM modulation: $\tilde{\mathbf{H}}_{i,j}$
 - 8: Aggregate nodes into graph embedding: $\mathbf{h}_{G_{i,j}}$
 - 9: Adapt structure with adapter: $\mathbf{H}_{i,j}^{\text{adapt}}$
 - 10: Predict delay vector: $\hat{\mathbf{y}}_{i,j}$
 - 11: Compute Huber loss: $\mathcal{L}_{i,j}$
 - 12: **end for**
 - 13: Backpropagate and update parameters: $f_{\theta}, g_{\phi_{i,j}}$
 - 14: **end for**
 - 15: **end for**
 - 16: **Output**: Cell Aging-aware delay $\{\hat{\mathbf{y}}_{i,j}\}$
-

IV. EXPERIMENTAL RESULTS

A. Experimental Setup

This section presents a detailed description of the experimental setup.

TABLE I: SIMULATION CONDITIONS INCLUDING AGING, STRESS, PROCESS, AND DRIVING POWER

Parameter	Aging	Stress	Process
Rise Transition	✓	✓	—
Fall Transition	✓	✓	—
Period	✓	✓	—
Output Load	✓	✓	—
Temperature	✓	✓	—
Supply Voltage	✓	✓	—
Signal Period	✓	✓	—
Aging Time	✓	—	—
Corner	—	—	✓
Threshold Voltage	—	—	✓
Poly Pitch	—	—	✓
Track Number	—	—	✓
Driving Power	—	—	—

1) *Condition Combination Selection*: The simulation conditions are grouped into three categories. *Aging conditions* specify the time-dependent degradation parameters, *stress conditions* represent operating conditions like voltage, temperature, and signal activity, and *process conditions* describe the manufacturing variations including corner, threshold voltage, poly pitch, and track number as shown in Table I.

2) *Training Details and Computational Setup*: The hierarchical RGAT-based framework was trained using the Adam optimizer with an initial learning rate of 1×10^{-3} for adaptive encoders and 1×10^{-4} for the shared RGAT encoder. A mini-batch size of 64 graphs per update was adopted, and gradients were clipped to a maximum norm of 1.0 to stabilize training. The Huber loss with $\delta = 1.0$ was employed for multi-variable prediction. Training was conducted on a server equipped with 8 NVIDIA GPUs, each with 20 GB memory. To accelerate the training process, data parallelism was utilized across multiple GPUs: each GPU processes a subset of the mini-batch, and gradients are aggregated synchronously before parameter updates. This setup enabled efficient training over the diverse cell structures and experimental conditions while maintaining consistent convergence.

B. Experimental Results

To assess the effectiveness of the proposed framework, we conducted comprehensive experiments using 56 representative standard cells implemented in an advanced semiconductor technology node. The overall experimental pipeline consists of two main stages. In the first stage, we performed large-scale circuit-level simulations on a selected subset of cells to generate abundant training data, which were then used to build a robust pre-trained model. In the second stage, we evaluated the adaptability of the pre-trained model by considering unseen scenarios, including previously unobserved cells, process variations, and different driving conditions. For each such case, only a limited number of simulation samples $M = 400$ were collected, enabling us to perform few-shot transfer learning.

C. Results of Pre-trained Model

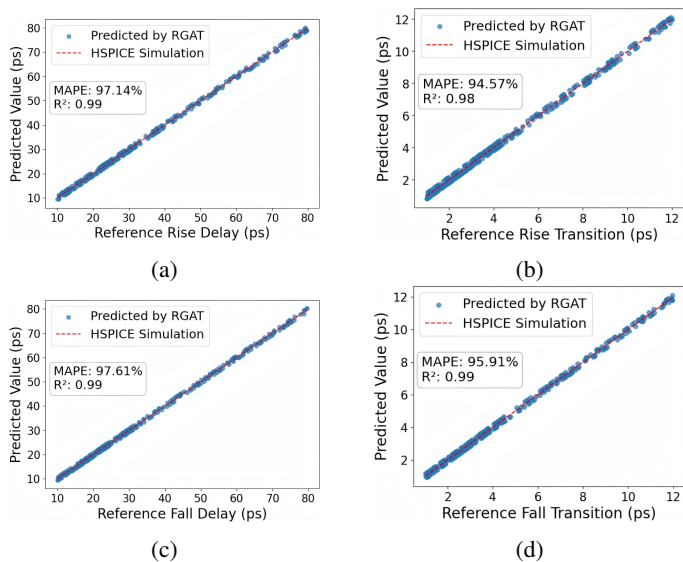


Fig. 4: Performance evaluation of the RGAT model pretrained on the AND2 cell. (a) Rise Delay, (b) Rise Transition, (c) Fall Delay, (d) Fall Transition.

To establish a reliable baseline, we first trained a pre-trained model using large-scale simulation data. As described

in Section III-C, we employed the RGAT-based architecture to capture the impact of stress-aging conditions on timing behavior. Specifically, we selected the two-input AND gate (AND2) as the representative circuit and simulated 100,000 stress-aging combinations. The trained model produces four timing metrics for each cell, namely rise transition, fall transition, rise delay, and fall delay as in Fig. 4. This pre-trained model serves as the foundation for subsequent transfer learning experiments. In this work, we evaluate the model prediction performance using two metrics: the Mean Absolute Percentage Error (MAPE) and the coefficient of determination (R^2).

MAPE measures the relative error between predicted and true values, and is defined as:

$$\text{MAPE} = 1 - \frac{100\%}{N} \sum_{i=1}^N \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \quad (18)$$

where y_i is the true value, \hat{y}_i is the predicted value, and N is the number of samples. The R^2 score quantifies the goodness of fit of the model, and is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^N (y_i - \hat{y}_i)^2}{\sum_{i=1}^N (y_i - \bar{y})^2}, \quad (19)$$

where \bar{y} is the mean of the true values. An R^2 closer to 1 indicates better model fit.

D. Results of Transfer Learning

Building on the pre-trained model, we evaluate the transfer learning framework from Section III-D, which efficiently adapts to new cell structures, process corners, and driving conditions via multi-task parallel training. Adaptive adapters in the graph encoder capture structural similarities while preserving discriminative features. Structurally similar cells are grouped and trained in parallel using few-shot samples; for example, an AND3 gate across three process corners, six driving strengths, and three threshold voltages results in 54 sub-tasks, which are predicted jointly by a unified model (Fig. 5). To handle variations in driving conditions, a structure adapter is introduced, while FiLM effectively handles process corner effects with minimal retraining, accelerating parallel adaptation.

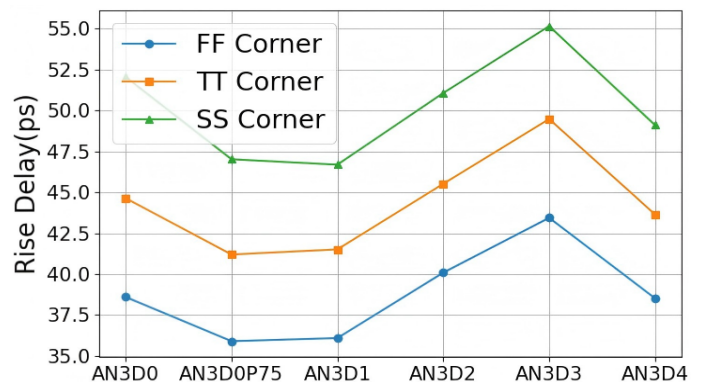


Fig. 5: Impact of different driving strengths and corners of AND3 on aging delay under a fixed set of stress-aging and process conditions.

TABLE II: EXPERIMENTAL COMPARISON OF THE PROPOSED METHOD VERSUS CONVENTIONAL STATIC TIMING ANALYSIS (STA) ON VARIOUS CIRCUIT DESIGNS. WNS AND TNS DENOTE WORST NEGATIVE SLACK AND TOTAL NEGATIVE SLACK, RESPECTIVELY. T0 REPRESENTS THE BASELINE (UNAGED), WHILE T10 REPRESENTS TEN YEARS OF AGING.

Design	Gate counts	WNS (ns)			TNS (ns)			Runtime (Hours)	
		STA (T0)	STA (T10)	This work	STA (T0)	STA (T10)	This work	STA (T10)	This work
DesignA	23K	-0.076	-0.112	-0.115	-0.259	-0.497	-0.501	10	<0.01
DesignB	70K	-0.014	-0.019	-0.019	-0.655	-0.731	-0.749	38	<0.01
DesignC	117K	-0.033	-0.041	-0.040	-0.721	-0.966	-0.958	66	<0.01
DesignD	>200 K	-0.007	-0.009	-0.009	-0.922	-1.136	-1.140	>100	<0.01

TABLE III: COMPARISON OF DIFFERENT METHODS ON GATE-LEVEL EVALUATION. WE USE THE ROOT MEAN SQUARED PERCENTAGE ERRORS (RE) AND R^2 TO MEASURE THE MODEL PERFORMANCE.

Gate	SPICE		Adam [19]		H-GAT [16]		Our framework	
	RE	R^2	RE	R^2	RE	R^2	RE	R^2
INV	0.00	1.00	0.45	0.95	0.03	0.98	0.03	0.98
NAND	0.00	1.00	0.42	0.96	0.02	0.99	0.05	0.99
OR	0.00	1.00	0.47	0.96	0.02	0.99	0.04	0.99
XNOR	0.00	1.00	0.42	0.96	0.03	0.99	0.04	0.98
XOR	0.00	1.00	0.25	0.97	0.04	0.98	0.03	0.99
OAI	0.00	1.00	0.81	0.95	0.04	0.99	0.04	0.99
Ave.	0.00	1.00	0.47	0.96	0.03	0.99	0.04	0.99

We present representative results for selected cells and benchmark them against state-of-the-art methods reported in the literature. The comparative performance is summarized in Table III. Our results closely match the state-of-the-art. However, unlike H-GAT, which requires retraining with 100,000 simulation samples for each new cell or condition, our framework achieves similar accuracy using only 400 few-shot samples per task, drastically reducing time and computational cost.

E. Runtime Analysis

In this subsection, we analyze the runtime performance of the different methods. Fig. 6(a) presents a comparison of the time consumption across the four approaches. It can be observed that AI-based methods, whether ML- or GNN-related, achieve substantial reductions in prediction time compared to HSPICE simulations. Among them, our method exhibits the lowest runtime, achieving a remarkable 17,025 \times speedup over HSPICE. Fig. 6(b) illustrates the additional time required for data preparation and model training for AI-based approaches. Notably, the H-GAT method requires 100,000 training samples for each new condition. In contrast, our transfer learning framework only selects 400 samples for each new condition combination. This significantly reduces the data preparation effort, especially when the number of condition combinations is large.

Furthermore, it is worth noting that the time consumption for one process condition in Fig. 6(b) is significantly less than 100,000 times the single-cell simulation time presented in Fig. 6(a). This efficiency is driven by two primary factors. First, we optimized the simulation process for similar stress-aging or process condition combinations, meaning the total time for X groups of simulations is substantially lower than X times the individual cell simulation time. Second, we employed a 30-core

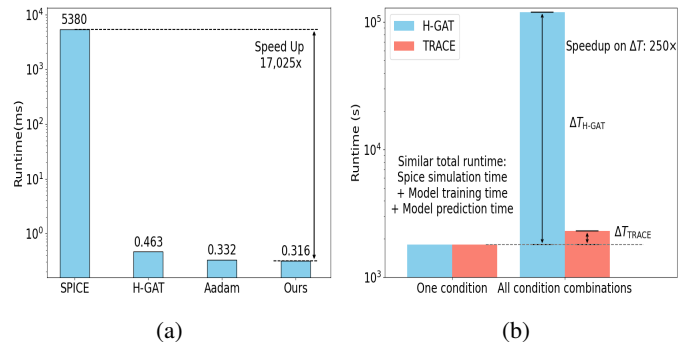


Fig. 6: Runtime analysis. Subfigure (a) compares the runtime of the proposed method with traditional approaches and other AI-based methods under a fixed process condition. Subfigure (b) reports two runtime comparisons with the H-GAT method: (i) prediction under a fixed process condition, and (ii) prediction across different process-condition settings.

CPU to enable multi-task parallel acceleration. By leveraging such industrial-grade approaches to compress simulation cycles, our framework aligns more closely with the requirements of practical industrial applications.

F. Circuit Static Timing Analysis

Table II demonstrates that our method can accurately capture the timing convergence of circuits under aging conditions while drastically reducing analysis time compared to traditional library-based STA. Unlike conventional approaches that require preparing separate libraries for different signoff scenarios—a process that is both time-consuming and resource-intensive—our approach eliminates this dependency without sacrificing accuracy. This enables rapid timing analysis and facilitates fast design iterations, making it particularly suitable for large-scale or frequently updated circuits.

V. CONCLUSION

This paper presents TRACE, a novel framework for cell-level aging-aware delay prediction, combining hierarchical multi-task and adaptive learning. Our approach delivers accurate predictions across diverse conditions and cell structures, while enabling effective knowledge transfer to new scenarios. Experimental results demonstrate superior performance and practical applicability, providing a scalable solution for aging analysis in IC design.

REFERENCES

- [1] K. Balaskas, G. Zervakis, H. Amrouch, J. Henkel, and K. Siozios, "Automated design approximation to overcome circuit aging," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 11, pp. 4710–4721, 2021.
- [2] L. Yip, R. Lin, C. Lai, and C. Peng, "Reliability challenges of high-density fan-out packaging for high-performance computing applications," in *2022 IEEE 72nd Electronic Components and Technology Conference (ECTC)*, pp. 1454–1458, 2022.
- [3] L. Wang, R. Thiagarajan, S. Jin, and Z. Zhang, "Accelerating simulation for high-fidelity pv inverter system reliability assessment with high-performance computing," in *2022 IEEE 49th Photovoltaics Specialists Conference (PVSC)*, pp. 0178–0182, 2022.
- [4] C. Dilopoulou and Y. Tsiatouhas, "Bti aging influence in sram-based in-memory computing schemes and its mitigation," in *2023 30th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, pp. 1–5, 2023.
- [5] Z. Ghaderi and E. Bozorgzadeh, "Aging-aware high-level physical planning for reconfigurable systems," in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 631–636, 2016.
- [6] T. Kishida, R. Kishida, and K. Kobayashi, "Investigation of the time-dependent bti-induced degradation distribution for ring oscillators in ultra-long-term stress conditions," in *2025 IEEE International Reliability Physics Symposium (IRPS)*, pp. 1–4, 2025.
- [7] B. Tudor, J. Wang, C. Sun, Z. Chen, Z. Liao, R. Tan, W. Liu, and F. Lee, "Mosra: An efficient and versatile mos aging modeling and reliability analysis solution for 45nm and below," in *2010 10th IEEE International Conference on Solid-State and Integrated Circuit Technology*, pp. 1645–1647, 2010.
- [8] R. Kishida, T. Asuke, J. Furuta, and K. Kobayashi, "Extracting voltage dependence of bti-induced degradation without temporal factors by using bti-sensitive and bti-insensitive ring oscillators," *IEEE Transactions on Semiconductor Manufacturing*, vol. 33, no. 2, pp. 174–179, 2020.
- [9] X. Cheng, Y. Ye, G. He, Q. Song, and P. Cao, "Heterogeneous graph attention network based statistical timing library characterization with parasitic rc reduction," in *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 171–176, 2024.
- [10] Z. Xue, Y. Liu, Y. Ye, T. Chen, H. Yan, and L. Shi, "Aging-aware logic restructure acceleration based on heterogeneous graph learning," in *2024 2nd International Symposium of Electronics Design Automation (ISEDA)*, pp. 480–485, 2024.
- [11] L. Alrahis, J. Knechtel, and O. Sinanoglu, "Graph neural networks: A powerful and versatile tool for advancing design, reliability, and security of ics," in *Proceedings of the 28th Asia and South Pacific Design Automation Conference*, pp. 83–90, 2023.
- [12] T. Chen, Q. Sun, C. Zhan, C. Liu, H. Yu, and B. Yu, "Deep h-gcn: Fast analog ic aging-induced degradation estimation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 7, pp. 1990–2003, 2021.
- [13] T. Chen, Q. Sun, C. Zhan, C. Liu, H. Yu, and B. Yu, "Analog ic aging-induced degradation estimation via heterogeneous graph convolutional networks," in *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, pp. 898–903, 2021.
- [14] L. Alrahis, J. Knechtel, F. Klemme, H. Amrouch, and O. Sinanoglu, "Gnn4rel: Graph neural networks for predicting circuit reliability degradation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 11, pp. 3826–3837, 2022.
- [15] J. Ye, P. Ren, Y. Xue, H. Fang, and Z. Ji, "Fast aging-aware timing analysis framework with temporal-spatial graph neural network," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023.
- [16] Y. Ye, T. Chen, Z. Wang, H. Yan, B. Yu, and L. Shi, "Fast and accurate aging-aware cell timing model via graph learning," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, no. 1, pp. 156–160, 2024.
- [17] W. Shi, H. Wang, J. Gu, M. Liu, D. Z. Pan, S. Han, and N. Sun, "Robustanalog: Fast variation-aware analog circuit design via multi-task rl," in *2022 ACM/IEEE 4th Workshop on Machine Learning for CAD (MLCAD)*, pp. 35–41, 2022.
- [18] H. Liang, W. Fu, and F. Yi, "A survey of recent advances in transfer learning," in *2019 IEEE 19th International Conference on Communication Technology (ICCT)*, pp. 1516–1523, 2019.
- [19] S. M. Ebrahimipour, B. Ghavami, H. Mousavi, M. Raji, Z. Fang, and L. Shannon, "Adam: A fast, accurate, and versatile aging-aware cell library delay model using feed-forward neural network," in *Proceedings of the 39th International Conference on Computer-Aided Design*, pp. 1–9, 2020.