electronics

Article

Multi-View Graph Learning for Path-Level Aging-Aware Timing Prediction

Aiguo Bu *, Xiang Li, Zeyu Li and Yizhen Chen

National ASIC System Engineering Research Center, School of Integrated Circuits, Southeast University, Nanjing 210023, China; xiang_l@seu.edu.cn (X.L.); 220236462@seu.edu.cn (Z.L.); 220236702@seu.edu.cn (Y.C.)
* Correspondence: 101010876@seu.edu.cn hl(A.B.)

Abstract: As CMOS technology continues to scale down, the aging effect—known as negative bias temperature instability (NBTI)—has become increasingly prominent, gradually emerging as a key factor affecting device reliability. Accurate aging-aware static timing analysis (STA) at the early design phase is critical for establishing appropriate timing margins to ensure circuit reliability throughout the chip lifecycle. However, traditional aging-aware timing analysis methods, typically based on Simulation Program with Integrated Circuit Emphasis (SPICE) simulations or aging-aware timing libraries, struggle to balance prediction accuracy with computational cost. In this paper, we propose a multi-view graph learning framework for path-level aging-aware timing prediction, which combines the strengths of the spatial–temporal Transformer network (STTN) and graph attention network (GAT) models to extract the aging timing features of paths from both timing-sensitive and workload-sensitive perspectives. Experimental results demonstrate that our proposed framework achieves an average MAPE score of 3.96% and reduces the average MAPE by 5.8 times compared to FFNN and 2.2 times compared to PNA, while maintaining acceptable increases in processing time.

Keywords: aging; NBTI; static timing analysis; graph learning; transformer

check for updates

Citation: Bu, A.; Li, X.; Li, Z.; Chen, Y. Multi-View Graph Learning for Path-Level Aging-Aware Timing Prediction. *Electronics* **2024**, *13*, 3479. https://doi.org/10.3390/electronics13173479

Received: 29 July 2024
Revised: 25 August 2024
Accepted: 29 August 2024
Published: 2 September 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In the process of advanced process to promote the chip feature size miniaturization to the nanometer level, chip reliability issues are increasingly prominent, which constitutes a serious test of the long-term stability and operating life of the chip [1]. Among them, the aging effect is the main cause of chip failure. Aging effect refers to the degradation of the electrical characteristics of devices in integrated circuits over time. The physical mechanism of aging effect mainly includes bias temperature instability (BTI), Hot Carrier Injection (HCI), Time-Dependent Dielectric Breakdown (TDDB), Electromigration (EM) and Self-heating Effect (SHE) [2,3]. Under advanced technology, negative bias temperature instability (NBTI) occurring on a P-type Metal-Oxide-Semiconductor (PMOS) has become the main factor affecting device reliability.

NBTI effect can be divided into two stages, including the stress stage and the recovery stage. In stress stage, negative bias voltage applied on PMOS ($V_{gs} < 0$) can cause the degradation of the threshold voltage (ΔV_{th}). After the stress is removed ($V_{gs} = 0$), the threshold voltage shift will partially recover to the level before stress injection. Shifted threshold voltage degrades the delay performance of standard cells, thus increasing the likelihood of timing violations on critical paths. The degradation of NBTI is influenced by factors such as voltage, temperature, runtime and workload-related signal probability (SP), resulting in the non-uniform degradation of path timing. Then, the ranking of critical paths in the fresh state will change after aging, which is illustrated in Figure 1.

In order to mitigate the NBTI effect on timing closure, circuit designers generally set targeted timing margins at the early design phase to cover harsh timing scenarios after aging. Then, finding an optimal margin is significant since excessive margins can result in performance limitations through over-design, while insufficient margins risk under-design and potential chip failure [4]. Therefore, accurate aging-aware timing analysis is crucial for margin evaluation. Conventional aging-aware timing analysis flows based on the SPICE simulation or the aging-aware gate model cannot achieve high accuracy and efficiency at the same time [5,6].

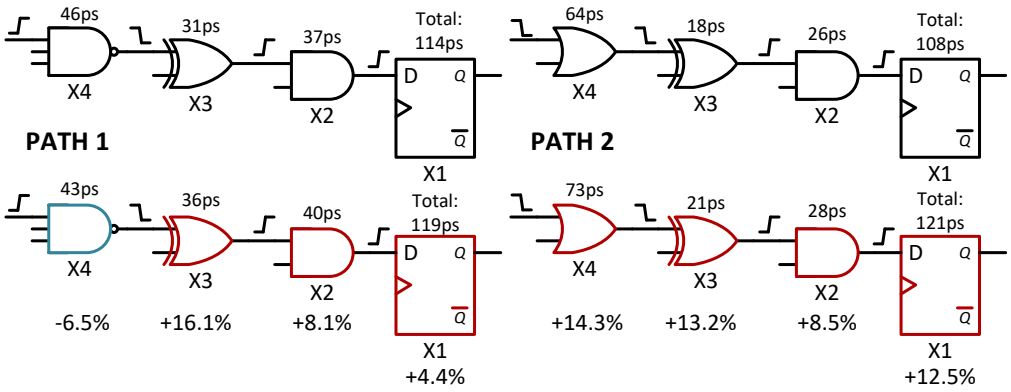


Figure 1. Example of ranking changes in timing paths after aging.

With the advancements in machine learning (ML) technology, ML approaches such as Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Bayesian, and probabilistic models have achieved remarkable results in various fields [7–9]. Researchers are increasingly inclined towards ML approaches in electronic design automation (EDA). Classical ML algorithm such as Support Vector Machines (SVMs), XGboost and convolutional neural networks (CNNs) have been proven to be effective in tackling EDA tasks with controllable runtime and great accuracy. Recently, graph learning techniques have emerged in the public eye due to their powerful ability to handle non-Euclidean data such as graphs. Several studies have shown that graph learning techniques can be used to solve complex problems in EDA, such as power estimation, timing prediction and placement optimization [10–12].

Inspired by existing graph neural network (GNN) applications in EDA, we propose an aging-aware path timing prediction framework based on multi-view graph learning. Timing paths are converted into graph form, fusing timing and workload information as features. We combine the STTN with GAT models to generate critical paths' joint representation for aging-aware timing prediction. The main contributions of this paper are summarized as follows:

- We have implemented an end-to-end aging-aware path time prediction framework based on multi-view graph learning, achieving a tradeoff between efficiency and accuracy.
- We customize a STTN-GAT model to improve the model's expressing ability and reduce the over-smoothing issues.
- The prediction accuracy and runtime of our model has been validated on multiple industrial designs.

The rest of this paper is organized as follows: Section 2 provides a brief overview of the prior work, including the NBTI mechanism and aging-aware timing analysis methods. Section 3 provides the background notions and the utilized learning models as the basis of our technique. Section 4 details our proposed aging-aware path timing prediction method. Section 5 presents the experimental setup and results. Section 6 concludes the paper.

2. Related Works

In recent years, many scholars have conducted research on aging-aware timing modeling, which is mainly divided into two categories: aging-aware SPICE simulation and

aging-aware STA. With the development of machine learning techniques, learning-based timing prediction methods have received widespread attention.

2.1. Existing Approaches for Aging-Aware Timing Modeling

2.1.1. Aging-Aware SPICE Simulation

The SPICE-based method relies on the transistor reliability model to obtain post-degradation transistor parameters, which are then back-annotated into the netlist to perform path-wise SPICE simulation. Although aging-aware SPICE simulation has high accuracy, the computational costs restrict its application in Very-Large-Scale Integrated Circuits (VLSI) with millions of instances. Therefore, circuit designers usually perform a SPICE simulation on a limited number of timing paths and determine the timing margin based on worst-case principles, which is added to STA for timing closure. However, this method may introduce over-design and limit the ultimate performance of chips.

2.1.2. Aging-Aware STA

Aging STA is implemented based on an aging-aware gate delay model and STA tools, which highly improves efficiency and enables full-chip reliability analysis. The aging-aware standard cell library will be re-characterized based on the traditional two-dimensional Look-Up Table (LUT), additionally considering aging-related factors [13–15]. However, gate delay degradation can vary significantly because of different mission profiles and workloads. The aging standard cell library based on a single aging scenario cannot accurately model the non-uniform degradation of path delay, and realizing timing analysis across diverse scenarios incurs substantial library characterization overhead. Recently, some researchers have adopted machine learning-based delay modeling methods to enhance accuracy performance and scalability [16,17]. In addition, commercial EDA companies also focus on the aging-aware library characterization. Synopsys's new-generation library characterization tool PrimeLib has taken reliability issues into account [18].

2.2. Frontier Machine Learning Techniques for Timing Prediction

Previous efforts have been made to apply machine learning techniques to timing prediction tasks such as pre-layout, interconnects and aging-aware timing estimation. Moreover, due to the large amount of topology and sequence information involved in timing prediction tasks, frontier machine learning algorithms led by GNNs and a Transformer have shown strong performance. Researchers presents a novel approach to predict path delays in circuit design by leveraging Transformer networks and residual learning to account for pre-routing and post-routing discrepancies [19]. In [11], a customized graph attention network method to estimate net length and timing before cell placement in integrated circuit design is proposed. In [20], wire timing predictions in large-scale circuit designs are significantly improved in speed and accuracy through the implementation of GNNTrans, a graph learning architecture that efficiently captures wire path representations. In [21], researchers develop a deep edge-featured graph attention network to predict path-based timing analysis results from graph-based timing analysis results, achieving fast and accurate timing prediction with reduced pessimism and runtime. In [22], a principal neighborhood aggregation (PNA)-based reliability assessment framework called GNN4REL enables designers to perform delay degradation estimation without accessing transistor models and standard cell libraries. In [23], researchers introduce an innovative aging-aware critical path selection methodology that employs graph attention networks to accurately identify critical cells in aged circuits, enhancing the reliability of timing analysis in advanced technology nodes.

3. Preliminaries

3.1. NBTI Degradation

Significant research has been carried out to build analytical models of NBTI. The two classic models for explaining the NBTI effect are the Reaction Diffusion (RD) theory

and the Trapping–Detrapping (TD) theory. Recent studies have demonstrated that the dynamic process of stress–recovery in NBTI originated from the TD theory, distinct from the threshold voltage degradation modeled by a power-law in the RD theory, wherein the TD theory yields a log-law model. The long-term model of dynamic NBTI under TD theory can be expressed as follows:

$$\begin{aligned}\Delta V_{th}(t) = \phi_1 \cdot [A + B\log(1 + C\alpha T_{clk})]\frac{1}{1-\beta_1\beta_2} + \\ \phi_2 \cdot [A + B\log(1 + C(1-\alpha)T_{clk})]\frac{\beta_1}{1-\beta_1\beta_2}.\end{aligned} \tag{1}$$

Here, β_1 and β_2 can be defined as follows:

$$\beta_1 = 1 - \frac{k + \log(1 + C\alpha T_{clk})}{k + \log(1 + C(t + C\alpha T_{clk}))}, \tag{2}$$

$$\beta_2 = 1 - \frac{k + \log(1 + C(1-\alpha) T_{clk})}{k + \log(1 + C(t + C\alpha T_{clk}))}, \tag{3}$$

where T_{clk}, α and t are the clock period, the signal probability, and the time, respectively. C and k are the fitting parameters of the TD theory.

However, the traditional single-stage log-law TD model is only effective in planar technology; it is no longer able to describe the dynamic behavior of NBTI in the full range for the latest process nodes such as FinFETs. In [24], a modified TD model is proposed, suggesting that both fast and slow traps jointly influence the NBTI aging effects in FinFET technologies. The modified NBTI closed-form model can be expressed as follows:

$$\Delta V_{th,fast} = M \cdot A_1 \exp(B_1 V_g) \exp(-\frac{E_{a1}}{k_B T/q}) \log(1 + C \cdot T \cdot DF), \tag{4}$$

$$\Delta V_{th,slow} = A_2 \exp(B_2 V_g) \exp(-\frac{E_{a2}}{k_B T/q}) \log(T \cdot DF)^{n_1}, \tag{5}$$

$$\Delta V_{th} = \Delta V_{th,fast} + \Delta V_{th,slow}, \tag{6}$$

where M is the modulation factor, T is the clock period, DF is the equivalent duty factor, and t is the time.

Currently, several commercial SPICE-based reliability application programing interfaces (APIs) have been developed to model NBTI effect, such as MOS Reliability Analysis (MOSRA) [25] and TSMC Modeling Interface (TMI) [26]. In this work, we will use MOSRA to conduct transistor NBTI degradation calculation, which is fully compatible with the HSPICE simulator.

3.2. Graph Neural Networks

In many practical application scenarios, data exists in the form of graphs, such as social networks, knowledge graphs, molecular structures, etc. These graph data have complex relationships and interdependence, and traditional neural network architectures such as convolutional neural networks (CNN) and recurrent neural networks (RNN) are not suitable for directly processing data in non-Euclidean space. Graph neural networks (GNNs) are specifically designed to process graph data, and they have achieved great success in various graph-based learning tasks, such as node classification, link prediction, graph classification, etc.

Usually, graph data structures are represented by $G = (V, E)$, where V represents the set of nodes in the graph, and E represents the set of edges in the graph. In this case, $v_i \in V$ can be represented as a node, and $e_{ij} = (v_i, v_j) \in E$ can be represented as a directed edge of node v_i pointing to node v_j. Furthermore, the set of neighboring nodes of node v can be defined as $N(v) = \{u \in V | (v, u) \in E\}$. If a graph has node attributes, the node attribute matrix is defined as $X \in \mathbb{R}^{n \times d}$, where the eigenvector of node v is represented as $x_v \in \mathbb{R}^d$;

Similarly, the edge attribute matrix is defined as $X^e \in \mathbb{R}^{m \times c}$, where the eigenvectors of edge ev, u are represented as $x^e_{v,u} \in \mathbb{R}^c$.

The graph convolutional neural network (GCN) is an important branch of graph neural network architecture, which is a successful extension of the traditional convolutional neural network for processing graph data, and the core idea is to produce a representation of the target node by aggregating the features of the target node with its neighboring nodes for downstream tasks. A message-passing mechanism is the core idea of GCN, allowing models to propagate and aggregate information through edges between nodes. The general expression for the message-passing function is as follows:

$$h_v^{(k)} = U_k(h_v^{(k-1)}, \sum_{u \in N(v)} M_k(h_v^{(k-1)}, h_u^{(k-1)}, x^e_{vu})), \tag{7}$$

where $h_v^{(k)}$ is the embedding of node v at the kth graph convolutional layer, x^e_{vu} is the embedding of edge $e_{v,u}$, and $U_k(\cdot)$ and $M_k(\cdot)$ are functions with learnable parameters. The existing graph convolutional neural network models, such as GCN, Graph Isomorphism Network (GIN), and graph attention network (GAT) [1,2], are mostly designed based on message-passing mechanisms, with differences only in the design of functions such as $U_k(\cdot)$ and $M_k(\cdot)$.

After exporting the hidden representation of each node, $h_v^{(k)}$ can be passed to the output layer to perform node-level prediction tasks, or to the readout function to perform graph-level tasks. The readout function generates a representation of the entire graph based on hidden node representations, typically represented as follows:

$$h_G = R(h_v^{(k)} | v \in G), \tag{8}$$

where $R(\cdot)$ represents a function with learnable parameters. The readout function is also known as graph pooling, and commonly used readout operations include mean, maximum, and sum.

3.3. Transformer Network

Traditional Natural Language Processing (NLP) algorithms often rely on handcrafted features and rule-based methods, which can be brittle and difficult to scale. They also struggle with long-range dependencies and complex linguistic phenomena. To address these limitations, researchers explored deep learning techniques such as RNNs and CNNs. However, RNNs are computationally expensive and have limited parallelization capabilities, while CNNs are better suited for tasks that involve local patterns and relationships [27].

The Transformer was introduced as a novel neural network architecture that leverages self-attention mechanisms to effectively capture long-range dependencies and complex linguistic phenomena, while also being more computationally efficient and parallelizable than RNNs [1]. The Transformer encoder consists of several identical layers, each of which contains two sub-layers: a self-attention layer and a feed-forward neural network (FFNN). The key component of the Transformer is the self-attention mechanism, which can be expressed as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{9}$$

where Q, K, and V are matrices of query, key, and value vectors, respectively, and d_k is the dimension of the key vectors. Multi-head attention is a variant of the self-attention mechanism used in the Transformer architecture, which can be expressed as follows:

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W_O \\ \text{where head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V), \end{aligned} \tag{10}$$

where W_i^Q, W_i^K, and W_i^V are matrices of query, key, and value projections, respectively, and W_O is a matrix of output projections. The second sub-layer is a feed-forward neural network, which serves as a pointwise non-linear transformation of the embeddings.

4. Aging-Aware Timing Prediction Model

4.1. Overview

Figure 2 shows the implementation process of the aging-aware path timing prediction model based on multi-scale graph learning. Firstly, considering that the fan-in and fan-out gates on the path affect its timing degradation, the static path 1-hop subgraph of the path is constructed, allowing the graph learning process to indirectly encode the features of the target node's neighbors. Furthermore, considering the continuous workload sequences loaded onto the netlist, the signal probability of all ports in the netlist will also change over time. Therefore, the static path 1-hop subgraph of the path needs to be extended along the time axis into a dynamic path 1-hop subgraph G1 to capture the aging and timing correlations between logic gates in both the temporal and spatial dimensions. For the dynamic path 1-hop subgraph, this paper adopts STTN to process the topological structure information and workload sequence information, capturing the attention coefficients between different nodes and time steps through the spatial–temporal self-attention mechanism to achieve effective graph representation. Additionally, the critical path is directly abstracted as a static path 1-hop subgraph G2, and GAT is used to capture the topological connections between logic gates on the critical path as well as timing features such as transition time and delay, supplementing the pre-aging timing information for the STTN model. Finally, the path timing information encoded by the STTN model and the GAT model is fused through a gated fusion network, which uses two learnable matrices as weights for the outputs of the two models, enabling the accurate prediction of the timing path delay changes in circuits under the NBTI effect.

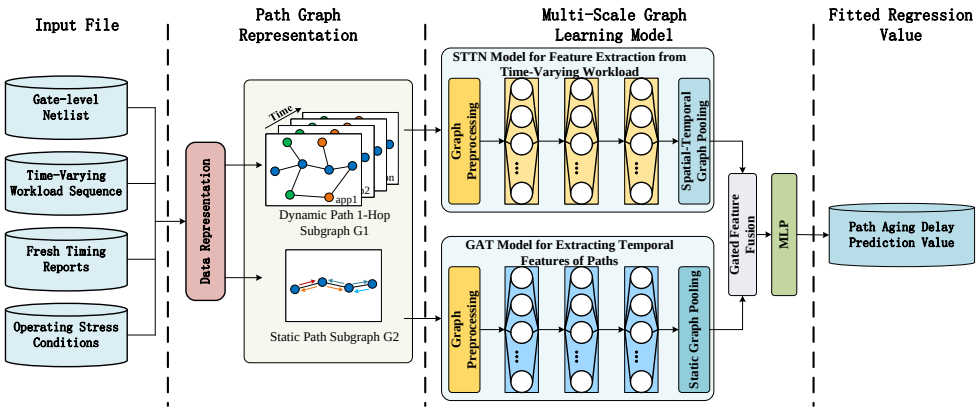


Figure 2. The overall process of the aging-aware timing prediction model.

4.2. Spatial–Temporal Transformer Network Design for Workload Features

4.2.1. Graph Representation and Workload Features

In this study, circuit netlists can be parsed and transformed into a graph data structure $G = (V, E)$, where V is a set of nodes $\{v_i \in V\}$ and E is a set of edges $\{e_{ij} \in E\}$. The nodes represent logic cells, primary inputs (PIs), and primary outputs (POs), while the edges represent the interconnections between the logic cells.

For circuit netlists in VLSI, directly modeling them as graph data structures for representation learning consumes substantial computational resources and carries the risk of overfitting. Furthermore, considering the time-varying characteristics of aging conditions,

learning on large-scale dynamic graphs becomes even more challenging. Therefore, a more efficient subgraph learning strategy is adopted in this paper. By targeting the critical paths extracted through STA, the original netlist graph is segmented and extracted to obtain a static path 1-hop subgraph structure based on timing paths. A static path 1-hop subgraph contains only the paths and the cells directly connected to the cells in the paths, without considering cells beyond the first hop.

Additionally, considering the impact of time-varying workloads, the static path 1-hop subgraph is extended to a dynamic path 1-hop subgraph. The proposed dynamic path 1-hop subgraph maintains the topological structure of the static path 1-hop subgraph, but the node features in the graph change over time.

When performing feature selection for the time-varying workload-sensitive dynamic path 1-hop subgraph, it is necessary to consider both the timing information within the subgraph and the aging-related features with dynamic characteristics. These features can be broadly categorized into transistor-level and cell-level features. As shown in Table 1, a series of features have been extracted from the circuit's timing report, gate-level netlist, and workload files.

Table 1. Original features for dynamic path 1-hop subgraph.

Type	Name	Description	Dimension
Node	cell_func	one-hot encoded cell type	8
	drive_strength	drive strength of cell	1
	wst_output_slack	worst slack of output pins	1
	wst_input_slack	worst slack of input pins	1
	max_input_slew	maximum slew of input pins	1
	max_output_slew	maximum slew of output pins	1
	tot_input_cap	total capacitance of input pins	1
	input_wst_sp	worst signal probability of input pins	1
	output_sp	signal probability of output pins	1
Global	op_temp	operation temperature	1
	op_voltage	operation voltage	1
	op_time	operation time range	1

4.2.2. Spatial–Temporal Embedding

When designing spatial–temporal graph models, it is necessary to consider the spatial positions of nodes within the graph and their dynamic changes over time. In the spatial dimension, attention should be given to the precursor units for signal inputs and the successor units for signal outputs of the target unit, thereby capturing the positional information of the nodes. In the temporal dimension, it is important to capture the temporal variation patterns of nodes to address the time-varying characteristics of workloads. Laplacian Positional Encodings (LPEs) are employed as spatial embeddings, extending the positional encoding used for sequence elements in Transformers to graphs [28,29]. The eigenvectors of the graph Laplacian matrix are utilized as an efficient representation of positional information in graph data, accommodating the nonlinear positional characteristics of graph structures. The calculation method for Laplacian eigenvectors is provided in the following formula:

$$\Delta = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = U^T \Lambda U, \tag{11}$$

where A is the $n \times n$ adjacency matrix, D is the degree matrix, and Λ and U correspond to the eigenvalues and eigenvectors, respectively. LPEs effectively capture distance-aware information, meaning that adjacent logic gates exhibit similar positional features, while distant logic gates have distinct positional features.

Through LPE, spatial embeddings offer static positional representations. Additionally, to capture the dynamic associations between logic gates across time steps, temporal embeddings are introduced, with each time step being encoded as a learnable vector to represent temporal positions. These spatial embeddings are then fused with the tempo-

ral embeddings to create spatial–temporal embeddings, enabling the capture of dynamic relationships between logic gates over time.

4.2.3. Spatial–Temporal Transformer Model

Algorithm 1 illustrates the process of the spatial–temporal transformer model proposed in this paper. Firstly, in Line 1 of the algorithm, the spatial–temporal attention layer is configured according to preset parameters. In Line 2, the node feature matrix X undergoes initial embedding through a Multilayer Perceptron (MLP) layer to obtain the initial embeddings for the first spatial–temporal attention layer. Lines 3–4 compute the spatial–temporal embedding.

Next, in Lines 5–6, the algorithm calculates the local spatial–temporal attention coefficients for the target node and its 1-hop neighboring nodes in both the time and space dimensions, meaning it considers only the direct neighbors of the target gate, not all gates in the entire graph.Futhermore, a learnable adaptive adjacency matrix A_{apt} is used to enhance local spatial–temporal attention, requiring no prior knowledge and allowing for end-to-end training and learning [30]. Its calculation formula is as follows:

$$A_{apt} = \text{softmax}\Big(\text{ReLU}(E_1 E_2^T)\Big), \tag{12}$$

where $E_1, E_2 \in \mathbb{R}^{N \times D}$, E_1 represents the original node embeddings, and E_2 represents the target node embeddings. The spatial dependency weights between the original and target nodes can be obtained through A_{apt}. To avoid high computational complexity, the Gumbel–Sigmoid method is used to generate a binary mask b from A_{apt} to limit node in-degrees and control the attention computation range. This mask ensures that the revised adjacency matrix A'_{apt} retains only a limited number of neighbors for each node through element-wise multiplication.

In Line 7, after computing the final spatial–temporal attention layer, the output node embeddings are aggregated along the time domain to generate static node embeddings. Finally, in Line 8, the global mean pooling of the full graph node embeddings is performed, and these are concatenated with the global graph features f_{g1} to generate the overall representation embedding for the dynamic path 1-hop subgraph.

Algorithm 1 Dynamic path 1-hop subgraph overall representation

Input: (1) Input graph: $G(V, E)$, (2) Node feature matrix: $X \in \mathbb{R}^{N \times T \times D}$, (3) Adjacency matrix: $A \in \mathbb{R}^{N \times N}$, (4) Number of STTN layers: L, (5) Number of STTN attention heads: K, (6) Sliding window width: window_width, (7) Historical sequence length: T
Output: Dynamic Path 1-Hop Subgraph overall representation R_{G1}

1: *initialize* $STAttBlock(L, K, T, window_{width})$
2: $X \leftarrow \text{MLP}_1(X)$
3: $STE \leftarrow STEmbedding(SE, TE)$
4: $STE_{his} = STE[:, :T]$
5: **for** each *net* in *STAttBlock* **do**
   X $\leftarrow$ net(X,STE$_{his}$)
6: **end for**
7: $H_S \leftarrow \text{Aggregate}_t(X, T)$
8: $R_{G1} \leftarrow \text{meanpool}(\text{H}_S)||f_{g1}$
9: **return** R_{G1}

4.3. Graph Attention Networks Design for Path Timing Features

4.3.1. Graph Representation and Timing Features

Path timing information can serve as supplementary features for aging-sensitive graph modeling proposed in Section 4.2. In this section, timing paths are selected to construct static path graph instead of static path 1-hop subgraph, because the graph attention network itself has the ability to aggregate neighborhood information.

When performing feature selection for a temporally sensitive path subgraph, the critical timing path topologies and timing information are retained, as shown in Table 2.

Table 2. Original features for dynamic path 1-hop subgraph.

Type	Name	Description	Dimension
Node	cell_func	one-hot encoded cell type	8
	drive_strength	drive strength of cell	1
	in_trans	transition time of input pins	1
	in_type	transition type of input pins	1
	out_trans	transition time of output pins	1
	cell_delay	cell delay	1
	cell_cap	cell load capacitance	1
	fanout	cell fanout number	1
Global	path_delay	fresh path delay	1
	path_depth	path depth	1

4.3.2. Graph Attention Model

The paper aggregates information from the other gates in a 3-hop neighborhood through three layers of GAT. The implementation process of the GAT layers, as shown in Figure 3, includes two parts: the calculation of normalized attention coefficients and feature weighting aggregation.

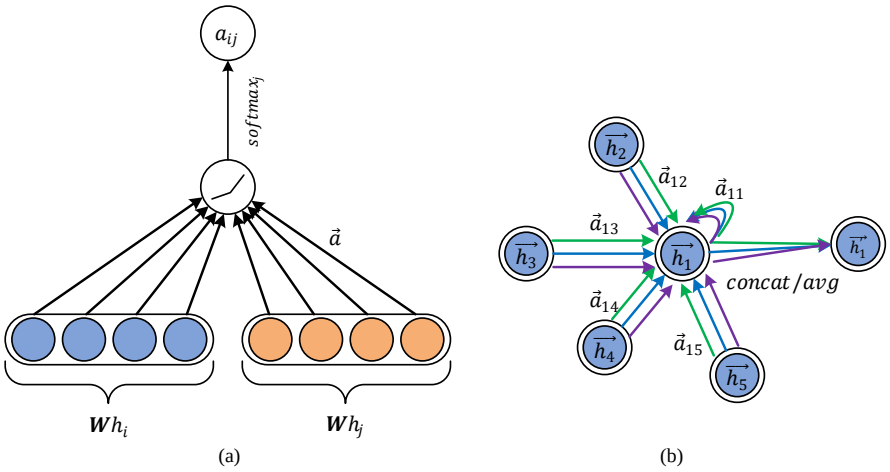


Figure 3. Diagram illustrating the implementation of a graph attention layer. (**a**) Attention Coefficient Calculation. (**b**) Feature Weighted Aggregation.

For a logic gate c_i and its 1-hop neighboring logic gate c_j, with corresponding node embeddings h_i and h_j, the attention score α_{ij} is given by:

$$e_{ij} = \text{LeakyReLU}(a^T[Wh_i \| Wh_j]), \tag{13}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}i} \exp(e_{ik})}, \tag{14}$$

where a represents a learnable weight vector, and W denotes a learnable weight matrix.

The new feature vector of the node logic gate c_i can be obtained by weighted aggregation of the features from neighboring nodes:

$$h_i' = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} W h_j\right). \tag{15}$$

To address the limitations in model expressiveness and stability of single-head attention in GAT, multi-head attention is introduced. The calculation formula for GAT using K attention heads in the multi-head attention mechanism can be expressed as:

$$h_i' = \|_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} W h_j\right). \tag{16}$$

After each GAT layer, a Batch Normalization (BN) layer is applied to accelerate the training process, and the output node embedding matrices from the first two BN layers are transmitted to the final BN layer via residual connections to reduce overfitting.

$$H^{l+1} = \mathrm{ReLU}(\mathrm{BN}(\mathrm{GAT}(H^l))), \tag{17}$$

$$H_{residual}^{l+1} = \mathrm{concat}(H^{l+1}, H^l, \ldots, H^0). \tag{18}$$

Then, each node's $H_{\mathrm{residual}}^{l+1}$ is aggregated through mean pooling, and concatenated with the global graph features f_{g2} to produce the graph-level representation R_{G2}.

$$R_{G2} = \frac{1}{N} \sum_{v \in V} (h_v) \| f_{g2}. \tag{19}$$

This representation is subsequently transformed into the final path graph embedding vector $\mathbf{x}_t$ containing the timing path information using a fully connected layer.

5. Experimental Result

5.1. Experiment Setup

The dataset generation process for this project was implemented on a Linux server with a 96-core CPU and 1.5 TB of RAM (Supermicro Storage A+ ASG-1115S-NE316R E3.S, Nanjing, Jiangsu, China), while the deep learning process was carried out on a Linux server equipped with a 28-core CPU, four NVIDIA V100 32 GB GPUs, and 256 GB of RAM (H3C UniServer R5300 G3, Nanjing, Jiangsu, China). The experiment utilized a commercial 16 nm FinFET process for both the standard cell library and the transistor model. To ensure that the dataset generation closely aligns with real chip operating scenarios, the operating voltages were set at 0.85 V, 0.9 V, 1.0 V, and 1.1 V. The operating temperatures were selected as 25 °C, 50 °C, 85 °C, and 125 °C, and the operating times were chosen as 1 year, 3 years, 5 years, and 10 years.

The benchmark circuits used in the experiment include the RISC-V core, FFT processor, and four OpenCores circuits, namely AC_97, AES_CORE, SYSTEMCDES, and WB_DMA. The RISC-V circuit uses the open-source Hummingbird E203 from Xilinx, which features a two-stage pipeline structure, including the Instruction Fetch Unit (IFU) and Execution Unit (EXU) stages. The FFT processor is a 256-point Fast Fourier Transform circuit design, implemented using a Radix-4 structure with single-path delay feedback technique, and has been verified through ASIC. As shown in Table 3, the RISC-V and FFT processors are used as the training set, representing known circuits, while the other four circuits are used as the test set, representing unknown circuits. From all the timing paths extracted from the training set circuits, 80% are randomly selected as training data for model training, and the remaining 20% of the timing paths are used to evaluate the prediction performance of the framework on known circuits.

Figure 4 illustrates the dataset generation flow. All circuit designs are synthesized using Synopsys Design Compiler. The "fresh" timing analysis and transistor level netlists generation of critical paths are performed by Synopsys PrimeTime. We use Synopsys Verification Compiler Simulator (VCS) simulator to conduct gate-level simulations with a specified workload. Based on the output waveforms, we implement cell-by-cell HSPICE MOSRA simulations to obtain ΔVth on PMOS transistors. Then, the aging information is back-annotated into the transistor-level netlists of paths for final aging-aware timing analysis to generate ground-truth labels.

Table 3. Testing circuit case statistical information description.

Design		#Cells	#FFs	#Train Paths	#Test Paths
Known	RISC-V	154,912	9829	10,385	2596
	FFT	102,226	9922	11,191	2238
Unknow	AC_97	12,787	2229	0	2129
	AES_CORE	16,424	530	0	659
	SYSTEMCDES	2258	190	0	445
	WB_DMA	4573	611	0	921

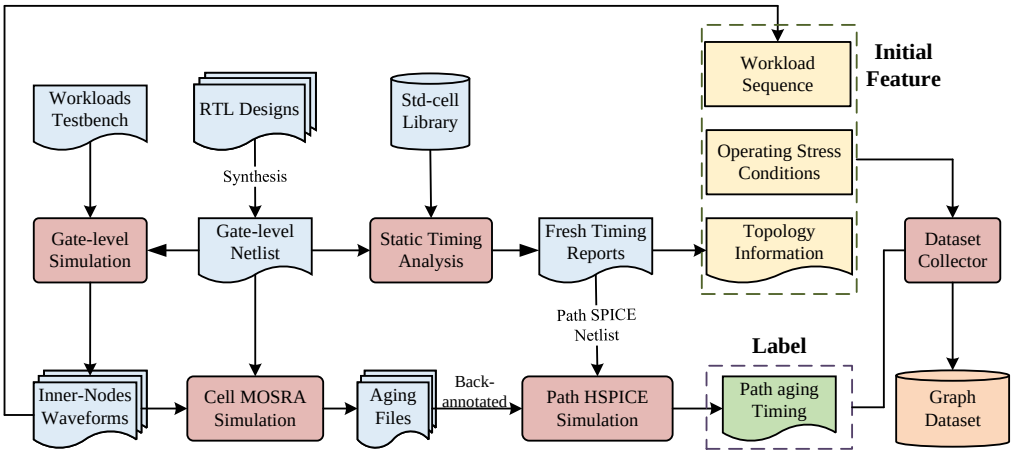


Figure 4. General flow for graph dataset generation.

For the training process, our model is trained with a gradient descent optimizer called Adam with a learning rate of 1×10^{-3}. We select Mean Squared Error (MSE) as the loss function of our regression task, represented in the following formula.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{20}$$

where N is the total number of timing paths in the overall dataset, y_i is the predicted result of our model, $\hat{y}_i$ is the ground-truth value.

5.2. Results and Comparison

5.2.1. Ablation Experiment

To validate the impact of each deep learning model on the prediction accuracy within the overall aging path timing prediction framework, this section conducts individual verification of the sub-models and quantitatively analyzes their contribution to the overall model's prediction accuracy.

Mean absolute percentage error (MAPE) and the coefficient of determination (R^2) are employed as the accuracy evaluation metrics for the deep learning model proposed in this study. The calculation formula for MAPE and R^2 is as follows:

$$\text{MAPE} = 100\% \times \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|. \tag{21}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{22}$$

In time-series forecasting tasks, the dimensional differences in delays between different paths can vary by up to three orders of magnitude. MAPE is very sensitive to situations where the actual values are close to zero, which can amplify the model's performance in

predicting small values. A smaller MAPE indicates a more accurate prediction performance by the model. R^2 provides a supplementary measurement of the model's ability to explain the overall variability in the dataset. It helps in understanding how well the model captures the inherent structure in the data and its superiority compared to simpler models.

Table 4 presents the comparison of prediction errors for each sub-model on the test circuit set. Here, Ma represents the sub-model with the STTN module removed, Mb represents the sub-model with the GAT module removed, Mc represents the sub-model where the gated fusion mechanism is replaced with a serial connection, Md represents the model that uses only local spatial–temporal attention without adaptive spatial–temporal attention, and M represents the complete model.

Table 4. Comparison of prediction errors for each sub-model.

Design		R^2*Score*/*MAPE* (%)				
		Ma	**Mb**	**Mc**	**Md**	**M**
Known	RISC-V	0.846/7.84	0.909/5.12	0.922/5.04	0.942/4.93	**0.974/3.90**
	FFT	0.850/5.87	0.914/4.43	0.938/5.14	0.973/4.67	**0.981/4.19**
Unknow	AC_97	0.910/6.09	0.916/4.97	0.952/4.92	0.962/4.72	**0.992/4.05**
	AES_CORE	0.811/10.27	0.908/4.83	0.948/3.30	0.968/4.06	**0.991/2.61**
	SYSTEMCDES	0.865/11.48	0.919/5.71	0.909/6.15	0.949/6.06	**0.962/5.77**
	WB_DMA	0.830/9.26	0.916/5.29	0.927/3.84	0.964/3.766	**0.986/3.21**
Average		0.852/8.468	0.914/5.083	0.933/4.732	0.959/4.701	**0.981/3.955**

The M model generally exhibits the best accuracy performance across all test circuits, indicating that each module has a positive impact on the model's performance. Futhermore, the MAPE score difference between the Ma sub-model and the complete model is the largest. Specifically, the M model reduces the average MAPE by 4.513% compared to the Ma model across all circuits, demonstrating that the STTN model significantly enhances the overall model's accuracy by effectively capturing time-varying workloads and path neighborhood information.

5.2.2. Comparison with Existing Models

In recent years, research on aging-aware timing prediction based on machine learning has mainly been divided into gate-level aging timing prediction and path-level aging timing prediction. Reference [16] constructed an aging-aware standard cell timing library using a deep learning model, replacing the original table lookup and interpolation steps in the STA process with model predictions. Reference [22] predicted the ratio of path delay degradation due to aging rather than obtaining the post-aging delay directly and could not effectively perceive workload information. Therefore, this section compares the accuracy of the proposed path delay aging prediction model with two aging timing prediction models based on FFNN [16] and the Principal Neighborhood Aggregation (PNA) graph neural network [22].

To ensure the fairness of model comparisons, the signal probabilities in the time-varying load sequences in this paper are averaged using summation, as data samples for the models in References [16,22], with data labels consistent with the proposed model.

Figure 5 presents a comparison of timing prediction errors for different models on the known and unknown circuit set. Compared to the FFNN model, the proposed aging path delay prediction framework reduces MAPE by factors of 4.0 and 6.6 on known circuits, and by factors of 4.9 to 7.4 on unknown circuits, with an overall reduction in the mean absolute percentage error by a factor of 5.8. Compared to the PNA model, the proposed aging path delay prediction framework reduces MAPE by factors of 2.7 and 1.8 on known circuits, and by factors of 1.8 to 3.6 on unknown circuits, with an overall reduction in the mean absolute percentage error by a factor of 2.2.

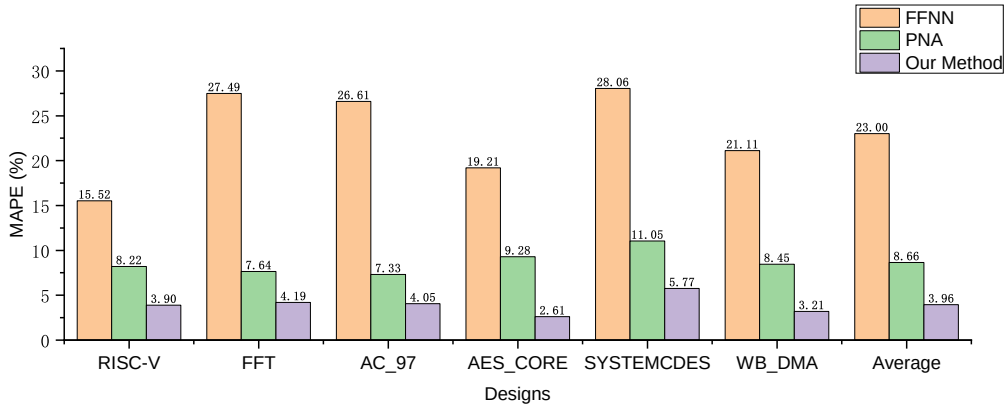


Figure 5. Timing prediction errors comparison among FFNN, PNA and our work.

Figure 6 presents a comparison of runtime for different models on the known and unknown circuit set. The runtime of the proposed prediction framework does not differ by more than an order of magnitude compared to the PNA-based method, indicating that the proposed model achieves better prediction accuracy while having a slightly longer runtime than existing methods.

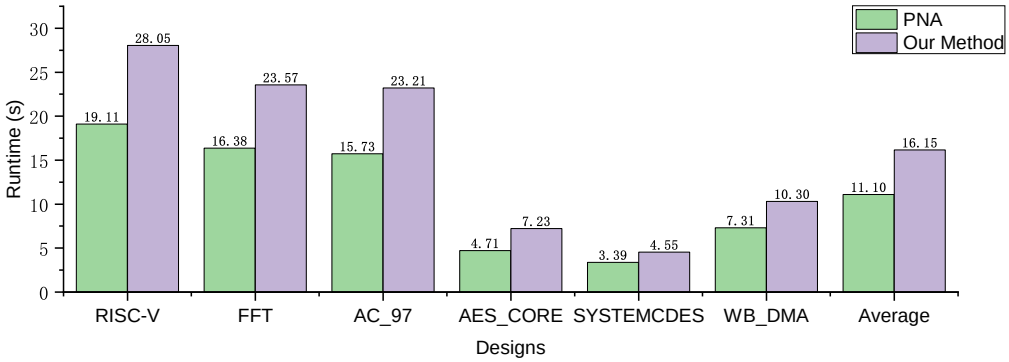


Figure 6. Runtime comparison between PNA and our work.

6. Conclusions

In this paper, an aging-aware path timing prediction framework is proposed based on muti-view graph representation learning. Our proposed framework adopts the spatial–temporal Transformer network and the graph attention network, harnessing the power of global attention mechanisms to create efficient and insightful timing path representations. The experimental results demonstrate that our work achieves an average MAPE of 3.96% on both known and unknown circuits, improving prediction accuracy by 5.8 times and 2.2 times, respectively, compared to existing approaches. Moreover, the runtime is on the same order of magnitude as methods based on PNA. Our proposed method can enable circuit designers to achieve more accurate aging-aware path timing prediction in the early stages of design, effectively reducing the timing margin.

There will be two significant improvements in our future work: first, while this paper has focused solely on path aging timing prediction, future research could explore how the prediction results can guide aging-aware optimization processes. Second, this study has not yet been tested on larger-scale circuits. Moving forward, we will investigate how techniques such as circuit partitioning can address memory and computational resource limitations, in order to explore and test the model's performance on larger-scale circuits.

Author Contributions: Conceptualization: A.B.; software: X.L.; data curation: A.B.; writing—original draft preparation: A.B.; writing—review and editing: Z.L.; supervision: Y.C.; project administration: A.B.; funding acquisition: A.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Key Research and Development Program of China (Grant No. 2019YFB2205004), and in part by the National Natural Science Foundation of China under Grant (62174031) and in part by the Jiangsu Natural Science Foundation (Grant No. BK20201233).

Data Availability Statement: Data are only available on request due to restrictions. The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Hill, I.; Chanawala, P.; Singh, R.; Sheikholeslam, S.A.; Ivanov, A. CMOS Reliability From Past to Future: A Survey of Requirements, Trends, and Prediction Methods. *IEEE Trans. Device Mater. Relib.* **2022**, *22*, 1–18. [CrossRef]
2. Kim, S.; Park, H.; Choi, E.; Kim, Y.H.; Kim, D.; Shim, H.; Chung, S.; Jung, P. Reliability Assessment of 3nm GAA Logic Technology Featuring Multi-Bridge-Channel FETs. In Proceedings of the 2023 IEEE International Reliability Physics Symposium (IRPS), Monterey, CA, USA, 26–30 March 2023; pp. 1–8.
3. Yasuda-Masuoka, Y.; Jeong, J.; Son, K.; Lee, S.; Park, S.; Lee, Y.; Youn Kim, J.; Lee, J.; Cho, M.; Lee, S.; et al. High Performance 4nm FinFET Platform (4LPE) with Novel Advanced Transistor Level DTCO for Dual-CPP/HP-HD Standard Cells. In Proceedings of the 2021 IEEE International Electron Devices Meeting (IEDM), San Francisco, CA, USA, 11 December 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 13.3.1–13.3.4.
4. Mishra, S.; Weckx, P.; Zografos, O.; Lin, J.Y.; Spessot, A.; Catthoor, F. Overhead Reduction with Optimal Margining Using A Reliability Aware Design Paradigm. In Proceedings of the 2021 IEEE International Reliability Physics Symposium (IRPS), Monterey, CA, USA, 21–25 March 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–7.
5. Tu, R.H.; Rosenbaum, E.; Chan, W.Y.; Li, C.C.; Minami, E.; Quader, K.; Ko, P.K.; Hu, C. Berkeley Reliability Tools-BERT. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **1993**, *12*, 1524–1534. [CrossRef]
6. Thirunavukkarasu, A.; Amrouch, H.; Joe, J.; Goel, N.; Parihar, N.; Mishra, S.; Dabhi, C.K.; Chauhan, Y.S.; Henkel, J.; Mahapatra, S. Device to Circuit Framework for Activity-Dependent NBTI Aging in Digital Circuits. *IEEE Trans. Electron. Devices* **2019**, *66*, 316–323. [CrossRef]
7. Wang, L.; Dernoncourt, F.; Bui, T. Bayesian Optimization for Selecting Efficient Machine Learning Models. *arXiv* **2020**, arXiv:2008.00386.
8. Jamhiri, B.; Xu, Y.; Shadabfar, M.; Costa, S. Probabilistic Machine Learning for Predicting Desiccation Cracks in Clayey Soils. *Bull. Eng. Geol. Environ.* **2023**, *82*, 355. [CrossRef]
9. Meng, C.; Xie, S.; Liu, L.; Wei, P.; Tang, Y.; Zhang, Y. Regional PM2.5 Concentration Prediction Analysis and Spatio-Temporal Mapping Incorporating ZWD Data. *Atmos. Pollut. Res.* **2024**, *15*, 102028. [CrossRef]
10. Guo, Z.; Liu, M.; Gu, J.; Zhang, S.; Pan, D.Z.; Lin, Y. A Timing Engine Inspired Graph Neural Network Model for Pre-Routing Slack Prediction. In Proceedings of the 59th ACM/IEEE Design Automation Conference, San Francisco, CA, USA, 10 July 2022; ACM: New York, NY, USA, 2022; pp. 1207–1212.
11. Xie, Z.; Liang, R.; Xu, X.; Hu, J.; Chang, C.-C.; Pan, J.; Chen, Y. Preplacement Net Length and Timing Estimation by Customized Graph Neural Network. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2022**, *41*, 4667–4680.. [CrossRef]
12. Shrestha, P.; Phatharodom, S.; Savidis, I. Graph Representation Learning for Gate Arrival Time Prediction. In Proceedings of the 2022 ACM/IEEE 4th Workshop on Machine Learning for CAD (MLCAD), Snowbird, UT, USA, 12 September 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 127–133.
13. Song Bian; Shintani, M.; Morita, S.; Hiromoto, M.; Sato, T. Nonlinear Delay-Table Approach for Full-Chip NBTI Degradation Prediction. In Proceedings of the 2016 17th International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, USA, 15–16 March 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 307–312.
14. Amrouch, H.; Khaleghi, B.; Gerstlauer, A.; Henkel, J. Reliability-Aware Design to Suppress Aging. In Proceedings of the 53rd Annual Design Automation Conference, Austin, TX, USA, 5 June 2016; ACM: New York, NY, USA, 2016; pp. 1–6.
15. Zhang, X.; Zhang, Z.; Lin, Y.; Ji, Z.; Wang, R.; Huang, R. Efficient Aging-Aware Standard Cell Library Characterization Based on Sensitivity Analysis. *IEEE Trans. Circuits Syst. II* **2023**, *70*, 721–725. [CrossRef]
16. Ebrahimipour, S.M.; Ghavami, B.; Mousavi, H.; Raji, M.; Fang, Z.; Shannon, L. Aadam: A Fast, Accurate, and Versatile Aging-Aware Cell Library Delay Model Using Feed-Forward Neural Network. In Proceedings of the 2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD), San Diego, CA, USA, 2–5 November 2020; ACM: New York, NY, USA, 2020; pp. 1–9.
17. Ye, Y.; Chen, T.; Wang, Z.; Yan, H.; Yu, B.; Shi, L. Fast and Accurate Aging-Aware Cell Timing Model via Graph Learning. *IEEE Trans. Circuits Syst. II* **2024**, *71*, 156–160. [CrossRef]

18. Synopsys, Inc. PrimeLib: Unified Library Characterization and Validation. [Online]. 2024. Available online: https://www.synopsys.com/implementation-and-signoff/signoff/primelib.html (accessed on 13 June 2024).
19. Yang, T.; He, G.; Cao, P. Pre-Routing Path Delay Estimation Based on Transformer and Residual Framework. In Proceedings of the 2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC), Taipei, Taiwan, 17 January 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 184–189.
20. Ye, Y.; Chen, T.; Gao, Y.; Yan, H.; Yu, B.; Shi, L. Fast and Accurate Wire Timing Estimation Based on Graph Learning. In Proceedings of the 2023 Design, Automation & Test in Europe Conference & Exhibition (DATE), Antwerp, Belgium, 17–19 April 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 1–6.
21. Ye, Y.; Chen, T.; Gao, Y.; Yan, H.; Yu, B.; Shi, L. Graph-Learning-Driven Path-Based Timing Analysis Results Predictor from Graph-Based Timing Analysis. In Proceedings of the 28th Asia and South Pacific Design Automation Conference, Tokyo Japan, 16 January 2023; ACM: New York, NY, USA, 2023; pp. 547–552.
22. Alrahis, L.; Knechtel, J.; Klemme, F.; Amrouch, H.; Sinanoglu, O. GNN4REL: Graph Neural Networks for Predicting Circuit Reliability Degradation. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2022**, *41*, 3826–3837. [CrossRef]
23. Ye, Y.; Chen, T.; Gao, Y.; Yan, H.; Yu, B.; Shi, L. Aging-Aware Critical Path Selection via Graph Attention Networks. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2023**, *42*, 5006–5019. [CrossRef]
24. Liu, J.C.; Mukhopadhyay, S.; Kundu, A.; Chen, S.H.; Wang, H.C.; Huang, D.S.; Lee, J.H.; Wang, M.I.; Lu, R.; Lin, S.S.; et al. A Reliability Enhanced 5nm CMOS Technology Featuring 5 th Generation FinFET with Fully-Developed EUV and High Mobility Channel for Mobile SoC and High Performance Computing Application. In Proceedings of the 2020 IEEE International Electron Devices Meeting (IEDM), San Francisco, CA, USA, 12 December 2020, IEEE: Piscataway, NJ, USA, 2020; pp. 9.2.1–9.2.4.
25. Fan, A.; Wang, J.; Aptekar, V. Advanced Circuit Reliability Verification for Robust Design. In Proceedings of the 2019 IEEE International Reliability Physics Symposium (IRPS), Monterey, CA, USA, 31 March–4 April 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–8.
26. Lee, W.-K.; Huang, K.; Hsu, L.C.; Huang, C.; Liang, J.; Chen, J.; Hsiao, C.; Su, K.-W.; Lin, C.-K.; Jeng, M.-C. A Unified Aging Model with Recovery Effect and Its Impact on Circuit Design. In Proceedings of the 2017 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD), Kamakura, Japan, 7–9 September 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 93–96.
27. Rezazadeh, N.; De Luca, A.; Perfetto, D. Unbalanced, Cracked, and Misaligned Rotating Machines: A Comparison between Classification Procedures throughout the Steady-State Operation. *J. Braz. Soc. Mech. Sci. Eng.* **2022**, *44*, 450. [CrossRef]
28. Dwivedi, V.P.; Joshi, C.K.; Luu, A.T.; Laurent, T.; Bengio, Y.; Bresson, X. Benchmarking Graph Neural Networks. *J. Mach. Learn. Res.* **2024**, *24*. [CrossRef]
29. Kreuzer, D.; Beaini, D.; Hamilton, W.L.; Létourneau, V.; Tossou, P. Rethinking Graph Transformers with Spectral Attention. In Proceedings of the 35th International Conference on Neural Information Processing Systems, Online, 6–14 December 2024; Curran Associates Inc.: Red Hook, NY, USA, 2024.
30. Wu, Z.; Pan, S.; Long, G.; Jiang, J.; Zhang, C. Graph Wavenet for Deep Spatial-Temporal Graph Modeling. In Proceedings of the 28th International Joint Conference on Artificial Intelligence, Macao, China, 10–16 August 2019; AAAI Press: Washington, DC, USA, 2019; pp. 1907–1913.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.