DeepSeq: Deep Sequential Circuit Learning

Sadaf Khan, Zhengyuan Shi, Min Li, Qiang Xu
The Chinese University of Hong Kong
{skhan, zyshi21, mli, qxu}@cse.cuhk.edu.hk

***Abstract*—In this work, we propose *DeepSeq*, a novel representation learning framework for sequential netlists. It employs a graph neural network (GNN) with customized propagation to capture temporal correlations. To ensure effective learning, we propose a multi-task training objective with two sets of strongly related supervision: logic probability and transition probability at each logic gate. A novel *dual attention* aggregation mechanism is introduced to facilitate learning both tasks efficiently. Experimental results validate DeepSeq's superiority over other GNN models in sequential circuit learning. It demonstrates accurate *reliability* and *power estimation* across diverse circuits and workloads.**

***Index Terms*—Representation Learning, Sequential Circuits, GNNs**

I. Introduction

Recently, circuit representation learning has emerged as a promising research direction in electronic design automation (EDA) field, where a pre-trained model is employed to solve multiple downstream tasks. For example, DeepGate family [1], [2] learns a generic gate-level representation of combinational circuits, that benefits solving test point insertion (TPI) and logic synthesis tasks. However, existing works in circuit representation learning mainly focuses on combinational netlists [1]–[3]. Due to the presence of memory elements (e.g., flip-flops – FFs) in sequential netlists, the circuit behavior is reflected as state transitions at each clock cycle. Consequently, capturing this behavior by learning effective embeddings for memory elements is an important and challenging problem to solve.

To this end, we propose DeepSeq, a novel representation learning framework based on graph neural networks (GNNs) for sequential netlists. For the combinational components of the sequential netlists, we convert them into an optimized and-inverter graph (AIG) format [1]. Given the sequential netlist in AIG format, DeepSeq employs a novel DAG-GNN architecture equipped with a customized propagation scheme and a dedicated aggregation mechanism named *Dual Attention* to effectively learn the temporal correlations between gates and FFs. Moreover, it utilizes a multi-task training objective, incorporating state transition probabilities and logic probabilities. This joint supervision scheme guides DeepSeq towards learning informative representations that reflect the true logical behavior of the underlying sequential netlists.

II. Methodology

Fig. 1 shows the overview of DeepSeq model. To prepare the sequential circuit dataset, we extract sub-circuits ranging from 150 to 300 nodes from ISCAS89, ITC99 and open-core benchmarks.The use of small-scale circuits accelerates training and DeepSeq can generalize to larger circuits due to GNNs' generalization capability (Section III).

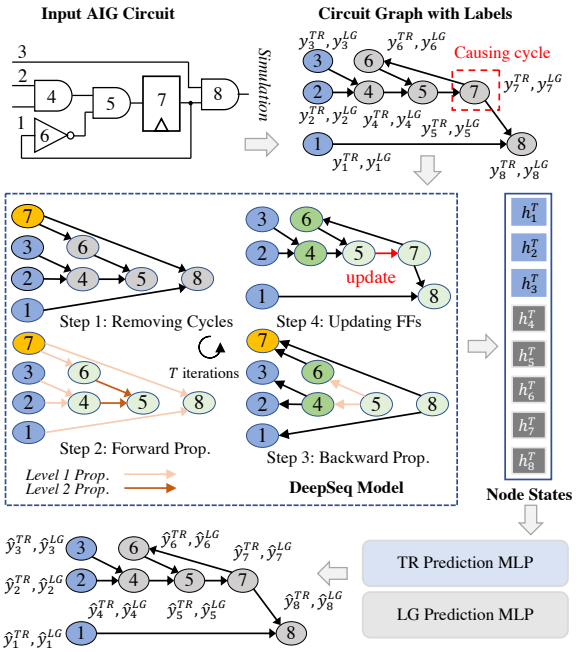

Fig. 1. The overview of DeepSeq framework for sequential circuit representation learning.

A. Training Objective: Multi-Task Learning

Given a sequential circuit in AIG format, DeepSeq utilizes multi-task learning to predict transition probabilities ($\mathcal{T}^{TR}$) and logic probability ($\mathcal{T}^{LG}$) simultaneously. Transition probabilities provides the temporal correlation information between gates and flip-flops in sequential circuits whereas logic probabilities reflects the structural and computational behavior of the combinational part of the sequential circuit. Therefore, they guide DeepSeq towards learning meaningful representation.

B. The Proposed Model

The directed logic propagation in sequential netlists makes DAG-GNN models, the suitable candidates for learning circuit representation. However, FFs can cause cycles in the circuit (Fig. 1). It is not straightforward to apply a DAG-GNN model to a directed cyclic graph. Therefore, we propose a novel DAG-GNN architecture to learn cyclic sequential netlists. We first map the input AIG circuit to a directed cyclic graph $\mathcal{G}$, with one-hot encoding of gate types as node features x_v for each node v. Every node v also has a embedding vector h_v that is learned during training to encode the representation of the sequential circuit.

During GNN propagation, we first remove the incoming edges of FFs to break cycles and make them psuedo primary inputs (PPIs). Then, we propagate the information sequentially

979-8-3503-4859-0/DATE24/© 2024 EDAA

from PIs/PPIs to POs. After that, we propagate the information through reverse topological graph to learn the implicit implications [1]. In the last step, we update FFs by directly copying the representations of their predecessors, emulating their behavior at each clock cycle. We iteratively repeat the above steps for T times to generate final hidden state h_v for every node v.

During forward and reverse propagation, we utilize the *dual attention* aggregation function to simultaneously learn $\mathcal{T}^{LG}$ and $\mathcal{T}^{TR}$. First, for a node v at an iteration t, we first compute the aggregated message $\mathbf{m}_v^{LG^t}$ from v's predecessors as follows:

$$\mathbf{m}_v^{LG^t} = \sum_{u \in \mathcal{P}(v)} \alpha_{uv}^t \mathbf{h}_u^t, \text{ where } \alpha_{uv}^t = \underset{u \in \mathcal{P}(v)}{softmax}(w_1^\top \mathbf{h}_v^{t-1} + w_2^\top \mathbf{h}_u^t) \quad (1)$$

where α_{uv}^t is a weighting coefficient that learns the impact of predecessors' information on node v, since different inputs have different impact on determining the output of a logic gate. Then, we perform another attention between $\mathbf{m}_v^{LG^t}$ and $\mathbf{h}_v^{t-1}$.

$$\mathbf{m}_v^{TR^t} = \alpha_{mv}^t \mathbf{m}_v^{LG^t}, \text{ where } \alpha_{mv}^t = softmax(w_1^\top \mathbf{h}_v^{t-1} + w_2^\top \mathbf{m}_v^{LG^t}) \quad (2)$$

where $\mathbf{m}_v^{LG^t}$ represents the logic computation result of node v at t^{th} iteration and $\mathbf{h}_v^{t-1}$ represents the computational state of node v at $t^{th}-1$ iteration. The intuition is that transition probabilities rely on the current and previous state of a node. We cancatenate the results from Eq. (1) and Eq. (2) and pass it to the GRU to update the hidden state h_v of node v. After T iterations, the final hidden states of all nodes are passed to two independent set of multi-layer perceptrons (MLPs) for prediction of $\hat{y}^{TR}$ and $\hat{y}^{LG}$.

III. Experiments

We compare DeepSeq with two baseline models, defined for directed graph: DAG-ConvGNN [4], [5] and DAG-RecGNN [6]. For both models, we try two different aggregation functions, i.e., convolutional sum and attention. Results show that DeepSeq surpasses the best-performing baseline model, i.e., DAG-RecGNN with attention as the aggregation function with a relative improvement of 20.00% and 15.79% for $\mathcal{T}^{TR}$ and $\mathcal{T}^{LG}$, respectively, in terms of average prediction error.

To demonstrate DeepSeq's generalizability, we evaluate it on power estimation and reliability analysis tasks. The test data comprises of six sequential circuits from open-core that are substantially larger and distinct from pre-training circuits.

A. Evaluation on Power Estimation

Pre-trained DeepSeq captures structural information and computational behavior of sequential circuits. However, we observe that the impact of workloads (distribution of transition probabilities) on large test circuits differs from pre-training circuits, due to low power design. Therefore, to learn the new distribution, we fine-tune DeepSeq on these large circuits. After fine-tuning with $1,000$ different workloads on a circuit, DeepSeq can generalize to arbitrary workloads for that circuit.

We use the transition probability obtained from logic simulation as the ground truth (GT). A probabilistic [7] and a learning-based (Grannite [8]) netlist-level power estimation methods are used as baseline. DeepSeq brings a significant improvement in error reduction, i.e., 80.49% over the probablistic baseline method and 62.38% over Grannite. (Table I).

TABLE I
The Results of Power Estimation on 6 Large-Scale Circuits

Design	# Nodes	GT	Prob.	Error.	Grannite	Error.	DeepSeq	Error.
noc_router	5,246	0.653	0.696	6.58%	0.641	1.85%	0.643	1.53%
pll	18,208	0.936	1.115	19.12%	0.829	11.41%	0.960	2.56%
ptc	2,024	0.247	0.184	25.55%	0.222	10.20%	0.239	3.24%
rtcclock	4,720	0.463	0.522	12.84%	0.437	5.72%	0.442	4.54%
ac97_ctrl	14,004	3.353	2.474	26.22%	3.943	17.60%	3.261	2.74%
mem_ctrl	10,733	1.365	1.471	7.77%	1.309	4.10%	1.303	4.54%
Avg.				**16.35%**		**8.48%**		**3.19%**

TABLE II
The Results of Reliability Analysis on 6 Large Scale

Design Name	GT	Prob.	Error	DeepSeq	Error
noc_router	0.9876	0.9607	2.72%	0.9814	0.63%
pll	0.9792	0.9501	3.95%	0.9857	0.35%
ptc	0.9970	0.9656	3.15%	0.9928	0.42%
rtcclock	0.9985	0.9812	1.73%	0.9969	0.16%
ac97_ctrl	0.9953	0.9704	2.50%	0.9943	0.10%
mem_ctrl	0.9958	0.9767	1.92%	0.9936	0.22%
Avg.			**2.66%**		**0.31%**

B. Evaluation on Reliability Analysis

For the reliability dataset, we conduct a fault-free and faulty (using the Monte Carlo method with a 0.05% error rate) simulations with 1,000 random sequential patterns. By comparing the output differences between these two simulations, we calculate the 0→1 and 1→0 error probabilities for each node as supervision. One-hot encoding of gate type is used as the node feature. We fine-tune DeepSeq on this dataset for 50 epochs using L1 loss. Reliability computed from simulation is used as ground truth (GT) wheras a probablistic method [9] servers as a baseline. Results show that DeepSeq brings a major improvement of 88.35% over the baseline method (Table II).

IV. Conclusion

In this work, we introduce *DeepSeq*, a novel representation learning framework for sequential netlist. Equipped with a customized propagation scheme and *dual attention* aggregation, it uses transition and logic probabilites to learn the accurate structural and computational circuit information. Despite showing promising results on power estimation and reliability analysis tasks, it exhibits a computational slowdown compared to commercial simulation tools due to its recursive sequential message passing approach. We leave it as future research work.

References

[1] M. Li *et al.*, "Deepgate: learning neural representations of logic gates," in *DAC*, 2022.

[2] Z. Shi *et al.*, "Deepgate2: Functionality-aware circuit representation learning," *arXiv preprint arXiv:2305.16373*, 2023.

[3] Z. Wang *et al.*, "Functionality matters in netlist representation learning," in *DAC*, 2022.

[4] M. Zhang *et al.*, "D-vae: A variational autoencoder for directed acyclic graphs," 2019.

[5] V. Thost and J. Chen, "Directed acyclic graph neural networks," in *ICLR*, 2021.

[6] S. Amizadeh, S. Matusevych, and M. Weimer, "Learning to solve circuit-SAT: An unsupervised differentiable approach," in *ICLR*, 2019.

[7] A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of average switching activity in combinational and sequential circuits," in *DAC*, 1992.

[8] Y. Zhang, H. Ren, and B. Khailany, "Grannite: Graph neural network inference for transferable power estimation," in *DAC*, 2020.

[9] H. Jahanirad, "Efficient reliability evaluation of combinational and sequential logic circuits," *J. Comput. Electron.*, 2018.