Critical Paths Prediction under Multiple Corners Based on BiLSTM Network

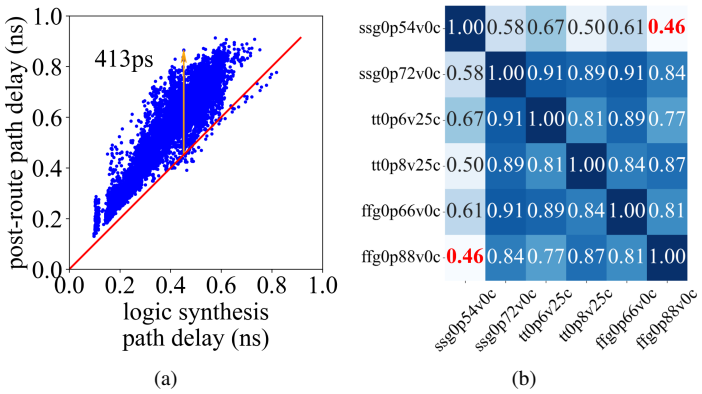
***Abstract*—Critical path generation poses significant challenge to integrated circuit (IC) design flow in terms of huge computational complexity and crucial impact to circuit optimization, whose early prediction is of vital importance for accelerating the design closure, especially under multiple process-voltage-temperature (PVT) corners. In this work, a post-routing critical path prediction framework is proposed based on Bidirectional Long Short-Term Memory (BiLSTM) network and Multi-Layer Perceptron (MLP) network to learn from the sequential features and global features at logic synthesis stage, which are extracted from the timing and physical information of cell sequences and operation conditions for circuit respectively. Experimental results demonstrate that with the proposed framework, the average prediction accuracy of critical paths achieves 95.0% and 93.6% for seen and unseen circuits in terms of F1-score for ISCAS'89 benchmark circuits under TSMC 22nm process, demonstrating an increase of 10.8% and 13.9% compared with existing learning-based critical paths prediction method.**

***Index Terms*—critical path prediction, corners, BiLSTM network**

I. Introduction

Static timing analysis (STA) plays an important role in integrated circuits (IC) design flow with heavy computation overhead and long runtime, which is repeatedly performed during logic synthesis and physical implementation to ensure the convergence of design. As a central task of STA, critical path generation takes a significant amount of time [1] and is of great importance for the guidance of design flow. Although plenty of works were devoted to reduce the runtime of critical path generation algorithm by constructing hardware-friendly data structure and increase parallelism [1–3], few of prior researches took the impact of later design stages and process-voltage-temperature (PVT) corners into consideration when identifying the potential critical paths, which may cause unnecessary iterations looping back to earlier stages to perform circuit optimization [4, 5].

The significant divergence of path delays between different design stages and PVT corners could be demonstrated in Fig. 1. As shown in Fig. 1(a), the path delays marked as blue dots that the increase between synthesis and routing stages varies dramatically for different paths and peaks at over 400ps, indicating that the paths with relatively small post-synthesis delay may even be more possible to be critical at routing stage. Therefore, it would be crucial to predict the critical path at early stage of design so that the designer can perform effective optimization to speed up design closure [6]. Moreover, due to the combination of process, voltage and temperature, the number of analysis corners grows explosively at the cost of substantial amount of computing resources for critical path



(a) (b)

Fig. 1. Divergence of critical path delay (a) between synthesis and routing stages (b) among PVT corners.

generation for each corner. Unfortunately, the impact of PVT corners poses great challenge to identify the critical paths [7]. In order to investigate the mis-correlation of path delays among corners, the radio of the common paths among the top 10% critical paths at two corners is quantitatively illustrated as matrix in Fig. 1(b). It can be seen that significant difference is manifested in the set of critical paths at different corners and the ratio of the common part could be as low as 46% between FF corner at 0.88V and SS corner at 0.54V.

To overcome the challenge induced by different design stages and PVT corners, a critical path prediction framework is proposed in this work at synthesis stage based on Bidirectional Long Short-Term Memory (BiLSTM) and Multi-Layer Perceptron (MLP) for multi-corners to predict whether the path is critical or not at routing stage. Sequential features are extracted to capture the cell sequence information along path and learned with BiLSTM network while global features including timing and PVT information are concatenated to be used by MLP network.

The main contributions are summarized as follows.

- To the best of our knowledge, this is the first study to predict the post-routing critical paths at synthesis stage under multiple corners, which enables the front-end engineers to employ the predicted criticality of post-routing paths as guidance of timely adjustments to reduce design iterations and accelerate design convergence.
- Owing to BiLSTM network, the post-routing criticality of paths is learned by capturing the timing and physical information from cell sequences as sequential features, where the impact of the fanin and fanout for each cell

is learned by LSTM network bidirectionally.
- The impact of PVT corner information to the path criticality is taken into account as global features by an MLP network, which improves the prediction accuracy for critical path under multiple corners.

The rest of the paper is organized as follows. Section II provides related works on critical path generation and timing prediction. Section III presents the related preliminaries. We give details of our proposed prediction framework in Section IV. Experimental results and discussion are given in Section V, followed by conclusion in Section VI.

II. RELATED WORKS

In recent years, the target of reducing the runtime of STA during design flow was achieved through two different ways. On the one hand, the efficient critical path generation algorithms were studied as key task of STA by reconstructing hardware-friendly data structure and utilizing heterogeneous computing. On the other hand, learning approaches were extended to the application of timing prediction across design stages and PVT corners, which was beneficial to the reduction of design iterations and acceleration of design convergence. Past works can be divided into the following categories.

Critical path generation algorithm. The critical path generation algorithms were studied thoroughly to accelerate the runtime of STA and shorten the design time. The novel algorithms proposed in [2, 3] can handle extensive path constraints and report any number of critical paths quickly. The former is sequential algorithm and target at CPU architectures. Depending on CPU multi-thread parallelism, it can only achieve $3\times$ speed-up at maximum. The latter constructs GPU-efficient data structures of suffix forest and prefix forest and leverages the power of GPU parallelism to accelerate critical path generation, achieving up to $102\times$ speed up. Instead of enumerating all path deviations iteratively, [1] uses mergeable heap to precompute the path deviations and applies a group of deviations to a path in a near-constant time, $1.8\times$ faster compared to path search algorithm based on suffix forest. Although remarkable acceleration is achieved by constructing hardware-friendly data structure and utilizing parallelism, the critical path is generated for single STA graph while the impact of later design stages and PVT corners is not considered in these works to reduce design iterations.

Timing prediction model across design stages. Some researches are devoted to the study of early timing prediction in the design flow with learning algorithms, which could be utilized to guide engineers to make corresponding adjustments in advance to reduce the iterations of the design flow. A net-based delay/slew model was introduced in [8] to predict the pre-routing path delay based on random forest algorithm to bridge the gap between the placement and routing stages. Different from [8], a path-based pre-routing timing prediction framework was proposed in [5] by transformer network with significant prediction accuracy improvement and runtime speed up. In [9], the post-routing TNS was predicted in early placement and CTS stages by exploiting sequential flow with LSTM networks based on graph learning. In [10], a CNN-based model was proposed to predict the critical paths at routing stage with the timing information at logic synthesis stage, which was utilized to guide the circuit optimization more strictly.

Timing prediction model across PVT corners. The timing analysis acceleration for multiple corners has also been extensively studied with learning-based approaches to reduce the runtime of STA [11]. Due to strong correlation of timing information under different corners, feature selection was performed to determine the best set of observed corners [12], with which learning models were built to predict the timing information of other unobserved corners [4]. Considering the impact of topology information and data set distribution on path timing, [13] and [14] achieve higher accuracy in the prediction of path timing. However, the above studies on the acceleration of multi-corners timing analysis are all conducted out at the same stage of design flow, but timing prediction in the early stage of design is more important to reduce design iterations.

In spite of plenty of learning models for path delay prediction, few of them were devoted to the prediction of critical paths to bridge the gap between the front-end and back-end, especially considering the impact of multiple PVT corners, which comes to be the target of this work.

III. PRELIMINARIES

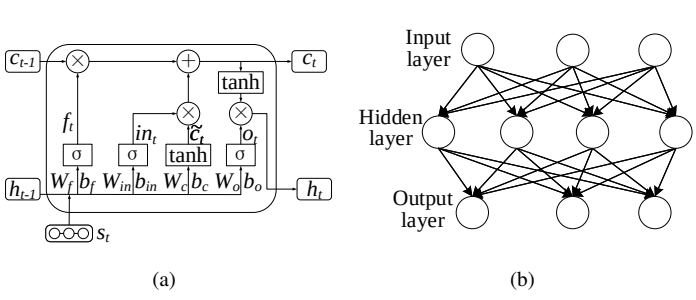
A. LSTM Network

Recurrent Neural Network (RNN) has a natural advantage in processing sequence problems due to its temporal memory function. LSTM [15] is a variant of RNN, which alleviates the problem of gradient descend/explosion that traditional RNN has when processing long sequences, and can effectively transfer and represent information in long sequences. The basic structure of LSTM layer, LSTM cell, is depicted in Fig. 2(a), which has an input gate(in_t), a forget gate(f_t) and an output gate(o_t) . Each gate uses the weights and bias in the figure to determine how much information needs to be saved from the previous state into the current state. For each time step in LSTM layer, h_t becomes the output of it, which is described by (1). For the last cell of the LSTM layer, T, h_T is the context vector of the whole sequence.

$$\begin{cases} in_t = \sigma\left(W_{in} \cdot [h_{t-1}, s_t] + b_{in}\right) \\ f_t = \sigma\left(W_f \cdot [h_{t-1}, s_t] + b_f\right) \\ o_t = \sigma\left(W_o \cdot [h_{t-1}, s_t] + b_o\right) \\ \widetilde{c}_t = \tanh\left(W_c \cdot [h_{t-1}, s_t] + b_c\right) \\ c_t = f_t * c_{t-1} + in_t * \widetilde{c}_t \\ h_t = o_t * \tanh\left(c_t\right) \end{cases} \tag{1}$$

B. MLP Network

MLP is an Artificial Neural Network (ANN) consisting of an input layer, an output layer and hidden layers as shown in Fig. 2(b). Each layer contains multiple neurons, and the neurons between layers are connected to each other. The existence of nonlinear activation function in neurons makes the MLP



(a) (b)

Fig. 2. Construction of (a) LSTM cell (b) MLP.

have strong fitting ability. At the same time, the error back propagation gradient descent algorithm makes the model can be optimized continuously. This is also an important reason for the wide application of MLP.

IV. CRITICAL PATH PREDICTION FRAMEWORK

A. Overview

The proposed critical path prediction framework is illustrated in Fig. 3 to identify whether a path is critical or not under multiple corners at routing stage based on the timing and physical information from post-logic synthesis netlist. Initially, STA is performed at logic synthesis stage as shown in Fig. 3(a). For each specific circuit path, the timing and physical information for every cell in the path are extracted and pre-processed into sequential features as shown in Fig. 3(c), where zero padding is performed to ensure that the length of feature sequence for each path is identical as that of the longest path. The sequential features for paths are processed by the BiLSTM network as shown in Fig. 3(e) for representation learning, which is consist of two LSTM layers to provide better context features considering both the fanin and fanout of each cell. Besides, due to the impact of clock cycle, PVT information and path delay, they are concatenated together to form the global feature as shown in Fig. 3(d), which are learned by an MLP in Fig. 3(f) to obtain the representation vector. Finally, in Fig. 3(g), the path sequence features represented by BiLSTM-based network and global features learned by MLP-based network are concatenated together and fed into a classification network. The probability of whether the path is critical or non-critical is generated by a $softmax$ function at the end of the classification network. With the labels from STA results for the post-routing netlist in Fig. 3(b), the whole framework is trained and the prediction accuracy is evaluated in terms of F1-score.

B. Feature Selection and Label Generation

Feature selection is crucial to the accuracy of machine learning-based model [16]. In this work, the path features used can be divided into sequential features and global features, as listed in Table I. The sequential features represent the local information of each cell in the path extracted from timing report at post-logic synthesis stage, including the cell type, cell delay, input transition, output load capacitance, and fanout, which contribute to the timing characteristics of sequential path

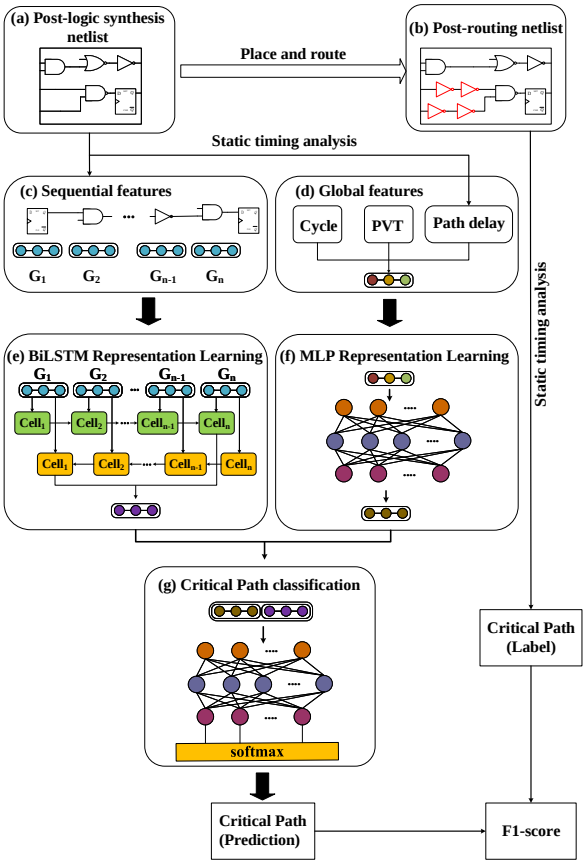


Fig. 3. Overview of the critical path prediction framework.

cells and are impacted by operation conditions at specific PVT corner. The global features are consist of the post-synthesis path delay, the clock cycle, and the PVT information, which are highly correlated with the criticality of paths at later design stages.

The labels of the critical path prediction framework are generated according to whether the path is critical after routing. During label generation, all paths are collected and matched between post-logic synthesis netlist and post-routing netlist according to their start-points and end-points. In the post-routing netlist, if the path is identified to be critical, its label is noted as 1 and if not, the label is noted as 0. In this work, the path is defined to be critical if its delay is no smaller than 90% of the target clock cycle time, which is compatible with the definition used in [10].

C. Sequential Feature Pre-processing

Motivated by the idea of text processing in the field of natural language processing, the timing and physical information of each cell along the path are preprocessed as sequential features, whose flow is demonstrated in Fig. 4. For non-numerical variables like cell type, the sequence is pre-processed by tokenizer to establish the mapping between the corresponding categorical variables and numbers, so as to convert the discrete non-numerical values to the discrete numerical values that can be recognized by the BiLSTM network. For continuous

TABLE I
SUMMARY OF FEATURES

Feature	Description	Data type
Sequence features		
Cell type	Cell type is related to the function and driven strength of the cell.	Enum
Cell delay	Cell delay is the delay between the input pin and the output pin of the cell, which is a direct part of path timing.	Float
Input tran.	Input transition is generally related to the cell delay and the output transition, which is further involved with the delay of the subsequent cell.	Float
Output load	Output load capacitance is generally related to the cell delay, which includes only pin capacitance at logic synthesis stage.	Float
Fanout	Fanout refers to the number of cells being driven, which is closely related to the load capacitance.	Int
Global features		
Path delay	Post-logic synthesis path delay provides a baseline for post-routing timing prediction.	Float
Clock cycle	Clock cycle is the timing constraint of a path, which is closely related to the criticality of the path.	Float
Process	One of circuit operation condition, including TT, FF, SS, and etc.	Enum
Voltage	One of circuit operation condition.	Float
Temperature	One of circuit operation condition.	Float

features including cell delay, input transition and output load capacitance, their sequences are discreted by binning operation to be mapped into different bins with the relative relationship of the data unchanged. For the discrete feature, fanout, its sequence is used without any additional pre-processing. After that, in order to keep the the dimensions of the input features consistent, zero padding is carried out to fill all sequences to an identical length, which is determined by the longest path in the data set.

D. BiLSTM Representation Learning

The process of BiLSTM representation learning is illustrated in Algorithm 1. First, the sequential features processed in Sec. IV-C go through the embedding layer in line 1, where the elements in the sequence are mapped as vectors. Then,

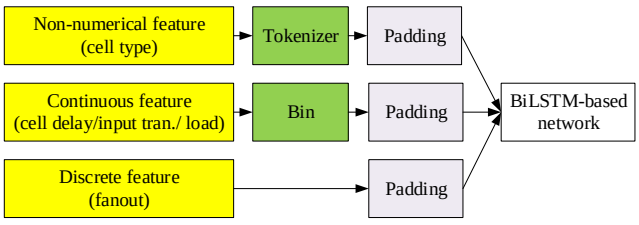


Fig. 4. Sequential feature preprocessing flow.

Algorithm 1 BiLSTM representation learning algorithm

Require: 1) P: set of paths,
2) K: length of data path in the design,
3) $s_p = \{s_p^1, s_p^2, ..., s_p^K\}$, $\forall p \in P$: feature sequence.
Ensure: LSTM representation vector s_p for all $p \in P$

1: $s_p \leftarrow \text{Embedding}\left(s_p\right)$
2: $s_p \leftarrow \text{Dropout}\left(s_p\right)$
3: **for** $k = 1, ..., K$ **do**
4: $\quad f_p^k, c_p^k \leftarrow \text{LSTM}_\text{f}\left(s_p^k, f_p^{k-1}, c_p^{k-1}\right)$ ▷ forwards
5: $\quad b_p^{K-k+1}, c_p^{K-k+1} \leftarrow \text{LSTM}_\text{b}\left(s_p^{K-k+1}, b_p^{K-k+2}, c_p^{K-k+2}\right)$ ▷ backwards
6: **end for**
7: $s_p \leftarrow \text{Contact}\left(f_p^K, b_p^1\right)$
8: $s_p \leftarrow \text{Dropout}\left(s_p\right)$
9: $s_p \leftarrow \text{Flatten}\left(s_p\right)$
10: $s_p \leftarrow \text{Dense}\left(s_p\right)$

in line 2, the dropout layer [17] randomly removes some neurons to reduce overfitting of the network. After that, in line 3-6, the feature are put into the BiLSTM network. The traditional LSTM network can only capture the relationship between elements in the sequence in one direction. However, in the path timing problem, the timing of the current cell will be affected by the timing of fanin and fanout cells, so the one-way logic will cause the loss of information, resulting in the decline of model accuracy. The BiLSTM network consists of two LSTM layers with opposite input feature sequences. For the forward LSTM layer ($LSTM_f$), the LSTM cell uses previous fanin cell state c_p^{k-1}, hidden state f_p^{k-1} and current input features s_p^k to update the current cell state c_p^k and hidden state f_p^k. For the backward LSTM layer ($LSTM_b$), the LSTM cell uses subsequent fanout cell state c_p^{K-k+2}, hidden state b_p^{K-k+2} and current input features s_p^{K-k+1} to update the current cell state c_p^{K-k+1} and hidden state b_p^{K-k+1}. From the aspect of network structure, BiLSTM transfers both forward and backward memory. For a data path, the influence of timing information such as input transition can be transferred to the fanin and fanout cells, which is more consistent with the signal propagation through the path. In line 7, the final output of BiLSTM is jointly determined by the output of the last LSTM cell in the forward LSTM layer (f_p^K) and the output of the first LSTM cell in the backward LSTM layer (b_p^1) . The dropout layer of line 8 is similar to that in line 2. The flatten layer in line 9 flattens the data to facilitate the combination with other features, and the dense layer in line 10 outputs the final representation of BiLSTM for sequence features.

E. MLP Representation Learning

Besides the sequential features, the global features are processed by MLP network to take the impact of timing constraint and PVT corner information into account with concatenated post-logic synthesis path delay, target clock cycle, and PVT information. The path delay at logical synthesis stage provides the basis for post-routing path timing prediction. The target clock cycle of design is one of the crucial factors whether

the path is critical or not. For multiple corners, the PVT information poses significant influence to the set of critical path, as investigated by the low ratio of common critical paths between different corners shown in Fig. 1(b). Owing the representation learning of MLP, the critical path prediction accuracy could be improved with the global features for the design.

V. EXPERIMENTAL RESULTS AND DISCUSSION

A. Experiment Setup

The proposed multi-corner critical path prediction framework was implemented in Python with the toolkits of keras and scikit-learn and compared with the prior learning approaches for critical path or path delay prediction [8, 10]. For the work proposed in [10], the features were identically selected as reported including the clock cycle, the cell type and the output load, where the load capacitance and the corresponding criticality of paths varied among different corners, so that the CNN model used in [10] was validated with multiple corners for comparison. By applying the model implemented with RF algorithm in [8], the predicted path delay is compared with the target clock cycle to determine whether a path is critical or not at routing stage. Due to the net-based prediction mechanism and corner-dependent features, the model in [8] could only be applied for single corner, leading to multiple training effort and storage cost. In this work, the ISCAS'89 benchmark circuits [18] were used for training and testing. All designs were synthesized with TSMC 22nm standard cell library by Synopsys Design Compiler and placed and routed by Synopsys IC Compiler. As summarized in Table II, three designs were used as seen circuits and the other two designs are unseen. We used Synopsys PrimeTime to report the path timing under 12 PVT corners as shown in Table III.

B. Results and Comparison

Table IV compares the average accuracy of the proposed BiLSTM-based framework for the critical path prediction with competitive RF-based [8] and CNN-based [10] models under all PVT corners in terms of recall, precision and F1-score. For seen designs, the F1-score of our proposed framework ranges from 93.5% to 96.1% with an average of 95.0%, which outperforms the CNN model with an increases by 10.8%. Although the average F1-score of RF model is slightly higher than that of this work, it can be seen that the proposed framework is superior to RF model in terms of recall. It is worthy noting that in the problem of critical path prediction, the prediction of non-critical path as critical path will cause unnecessary optimization effort but the prediction of critical path as non-critical path will cause unacceptable ignoring of timing violations, which means recall should be more of a concern than precision and the merit in recall is crucial for prediction model. For unseen designs, the F1-score of our framework ranges from 92.2% to 95.0% and is average at 93.6%, which demonstrates a satisfactory accuracy with an increase of 13.9% and 2.7% compared to CNN model and RF model respectively. Moreover, it should be noted that regardless of seen or unseen circuits, the RF model is required to be trained for each design corner, which is unable to predict critical path under multiple corners and suffers from exponential increase of training effort for all corners.

TABLE II
CIRCUIT STATISTICS

Ckt.	#Reg	#Cell	#Train	#Test	Type
s5378	163	750	25416	6360	seen
s13207	385	1152	51912	12984	
s38584	1201	5984	337956	84492	
s9234	145	606	0	37440	unseen
s15850	262	1065	0	45600	

TABLE III
CORNER STATISTICS

Corner	Process	Voltage (V)	Temperature (°C)
#1∼#4	SS	{0.54, 0.63, 0.72, 0.81}	0
#5∼#8	TT	{0.60, 0.70, 0.80, 0.90}	25
#9∼#12	FF	{0.66, 0.77, 0.88, 0.99}	0

In order to further demonstrate the prediction results of our proposed model under multi-corners, Fig. 5 shows the F1-score of the three models for an unseen circuit (s15850) under twelve corners separately. It can be seen that for the unseen circuit, our model achieves the F1-score between 91.0% and 96.5%, which is superior to the CNN-based framework in all corners and the RF model in most of them.

The high prediction accuracy of our framework is due to the following reasons. First, the BiLSTM network learns the criticality from sequential features by exploiting the timing and physical information of data path thoroughly under multiple corners. In contrast, the CNN model and the RF model learns from the path itself and a net of path respectively. Second, the carefully selected global features including PVT information increases the prediction ability for the critical path under multiple corners. Although the features used in CNN model are also impacted by PVT corner, it is validated by Table IV and Fig. 5 that its prediction accuracy is much poor than the proposed framework.

In addition, the path representation learned by our model is analyzed by plotting the tSNE [19] diagram of representation vector in Fig. 6. It can be clearly see the critical paths and non-critical paths can be divided into two distinct clusters, which demonstrates that our model has the ability to learn high-quality path representation, so that it can perform accurate critical path prediction.

VI. CONCLUSIONS

In this paper, a post-routing critical path prediction framework under multi-corners is proposed based on BiLSTM network and MLP network. Experimental results demonstrate that the proposed framework achieves significant precision improvement compared with the existing method. Even for a unseen design, F1-score is no less than 91.0%. The accurate classification results enable the designer to make adjustments in early stage to reduce the design iterations and accelerate the design convergence.

TABLE IV
COMPARISON OF CRITICAL PATH CLASSIFICATION ACCURACY WITH RECALL, PRECISION AND F1-SCORE RESULTS.

Type	Ckt.	CNN[10] (Multi-corner)			RF[8] (Single-corner)			BiLSTM (This work, multi-corner)		
		precision	recall	F1-score	precision	recall	F1-score	precision	recall	F1-score
seen	s5378	81.6%	87.6%	84.5%	99.1%	96.9%	98.0%	92.7%	98.5%	95.5%
	s13207	72.9%	86.0%	78.9%	93.9%	98.0%	95.9%	89.6%	97.7%	93.5%
	s38584	85.2%	93.3%	89.1%	97.4%	97.5%	97.4%	94.3%	98.0%	96.1%
	Average	79.9%	89.0%	84.2%	96.8%	97.5%	97.1%	92.2%	98.1%	95.0%
unseen	s9234_1	81.4%	70.9%	75.8%	88.8%	88.4%	88.6%	91.5%	93.0%	92.2%
	s15850	86.8%	80.7%	83.6%	92.2%	93.9%	93.1%	93.8%	96.2%	95.0%
	Average	84.1%	75.8%	79.7%	90.5%	91.2%	90.9%	92.7%	94.6%	93.6%

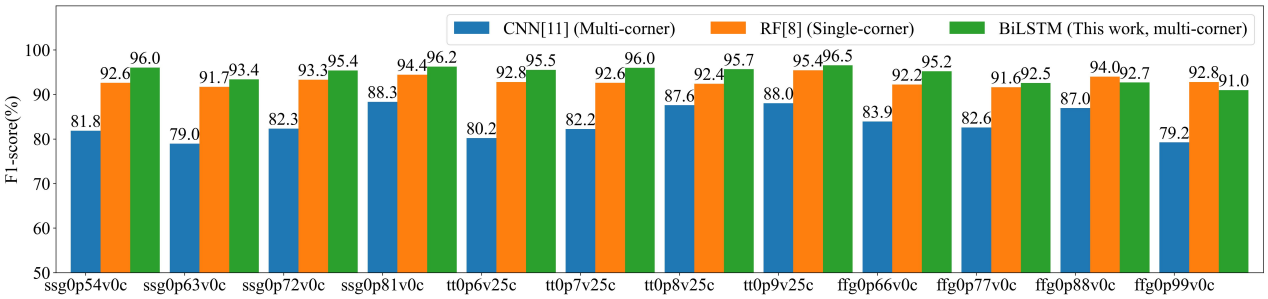
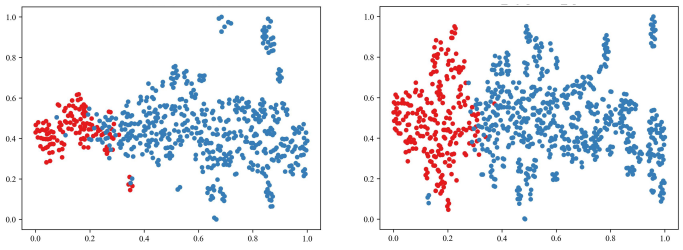


Fig. 5. F1-score of different models under multiple corners on an unseen design (s15850).



(a) (b)

Fig. 6. tSNE result of representation vector reported under FF corner at 0.66V and 0°C for paths from unseen designs (a) s9234 and (b) s15850. The red dots represent non-critical paths and the blue dots represent critical paths.

REFERENCES

[1] K. Zhou, Z. Guo, T. W. Huang, et al "Efficient Critical Paths Search Algorithm using Mergeable Heap," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2022, pp. 190-195.

[2] G. Guo, T. W. Huang, C. X. Lin, et al, "An Efficient Critical Path Generation Algorithm Considering Extensive Path Constraints," in *Design Automation Conference (DAC)*, 2020, pp. 1-6.

[3] G. Guo, T. W. Huang, Y. Lin, et al, "GPU-accelerated Critical Path Generation with Path Constraints," in *International Conference On Computer Aided Design (ICCAD)*, 2021, pp. 1-9.

[4] A. B. Kahng, U. Mallappa, L. Saul, et al, ""Unobserved Corner" Prediction: Reducing Timing Analysis Effort for Faster Design Convergence in Advanced-Node Design," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2019, pp. 168-173.

[5] T. Yang, G. He and P. Cao, "Pre-Routing Path Delay Estimation Based on Transformer and Residual Framework," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2022, pp. 184-189.

[6] B. Shook, P. Bhansali, C. Kashyap, C. Amin, et al, "MLParest: Machine Learning based Parasitic Estimation for Custom Circuit Design," in *Design Automation Conference (DAC)*, 2020, pp. 1-6.

[7] J. Kim, G. Lee, K. Choi, et al, "Adaptive delay monitoring for wide voltage-range operation,"in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2016, pp. 511-516.

[8] E. C. Barboza, N. Shukla, Y. Chen, et al, "Machine Learning-Based Pre-Routing Timing Prediction with Reduced Pessimism," in *Design Automation Conference (DAC)*, 2019, pp. 1-6.

[9] Y. C. Lu, S. Nath, V. Khandelwal, et al, "Doomed Run Prediction in Physical Design by Exploiting Sequential Flow and Graph Learning," in *International Conference On Computer Aided Design (ICCAD)*, 2021, pp. 1-9.

[10] W. L. Neto, M. Trevisan Moreira, L. Amaru, et al, "Read your Circuit: Leveraging Word Embedding to Guide Logic Optimization," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2021, pp. 530-535.

[11] M. Shoniker, O. Oleynikov, B. F. Cockburn, et al, "Automatic selection of process corner simulations for faster design verification", in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2017, vol. 37, no. 6, pp. 1312-1316.

[12] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection", *Machine Learning Research*, vol. 3, 2003, pp. 1157

[13] W. Bao, P. Cao, H. Cai, et al, "A learning-based timing prediction framework for wide supply voltage design," in *Great Lakes Symposium on VLSI (GLVLSI)*, 2020, pp. 309-314.

[14] P. Cao, W. Bao, K. Wang, et al, "A Timing Prediction Framework for Wide Voltage Design with Data Augmentation Strategy," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2021, pp. 291–296.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," in *Neural computation*, 1998, vol. 9, no.8, pp. 1735-1780.

[16] J. Li, K. Chenget, S. Wang, et al, "Feature Selection: A Data Perspective," in *ACM Computing Surveys*,2018, vol. 50, no. 6, pp. 1–45.

[17] Srivastava N, Hinton G, Krizhevsky A, et al, "Dropout: a simple way to prevent neural networks from overfitting", *The journal of machine learning research*, 2014, vol. 15, no. 1, pp. 1929-1958.

[18] F. Brglez, D. Bryan, and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," in *International Symposium on Circuits and Systems (ISCAS)*, 1989, pp.1929-1934.

[19] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. in *Journal of machine learning research*, 2008,vol. 9, no. 11, pp. 2579-2605.