A Unified Framework for Layout Pattern Analysis With Deep Causal Estimation

Ran Chen, Shoubo Hu, Zhitang Chen, Shengyu Zhu, Bei Yu, *Member, IEEE*, Pengyun Li, Cheng Chen, Yu Huang, *Senior Member, IEEE*, and Jianye Hao

***Abstract*—The decrease of feature size and the growing complexity of the fabrication process lead to more failures in manufacturing semiconductor devices. Therefore, identifying the root cause layout patterns of failures becomes increasingly crucial for yield improvement. In this article, a novel layout-aware diagnosis-based layout pattern analysis framework is proposed to identify the root cause efficiently. At the first stage of the framework, an encoder network trained using contrastive learning is used to extract representations of layout snippets that are invariant to trivial transformations, including shift, rotation, and mirroring, which are then clustered to form layout patterns. At the second stage, we model the causal relationship between any potential root cause layout patterns and the systematic defects by a structural causal model, which is then used to estimate the average causal effect (ACE) of candidate layout patterns on the systematic defect to identify the true root cause. Experimental results on real industrial cases demonstrate that our framework outperforms a commercial tool with higher accuracies and around ×8.4 speedup on average.**

***Index Terms*—Causality, deep learning, design for manufacturability, failure analysis, fault diagnosis.**

I. Introduction

The YIELD of manufactured integrated circuits (ICs) is defined as the percentage of good dies among all dies manufactured. A high and stable yield could ensure the profitability and reliability of products. However, as the feature size decreases, specific layout patterns that are hard to fabricate tend to cause more systematic defects, such as open or bridge defects in neighboring wires. These layout patterns are an important source of yield loss. Since layout configurations of new designs may differ from existing ones, identifying layout patterns that lead to yield loss through test chips, SRAMs, etc., is becoming less effective. Performing hotspot detection [1]–[4] on entire layouts may result in overcorrection which can adversely affect chip area and performance. Physical failure analysis (PFA) is a straightforward method to determine whether a layout pattern is the root cause of systematic defects. However, it requires both experience and a proper understanding of the fabrication process and is usually time consuming and expensive.

To efficiently identify the root cause of systematic defects, statistical methods have been adopted to automatically identify common physical defect features by analyzing volume diagnosis reports. One of the most prominent work is a Bayesian method proposed in [5], which characterizes the conditional distribution of systematic defect given potential root causes. It learns the optimal root cause distribution by maximizing the likelihood of observed diagnosis report using an expectation–maximization (EM) learning algorithm. There are also works [6], [7] focusing on improving the quality of diagnosis results by evaluating the impact of diagnosis features to improve the root cause identification accuracy. These methods fall short of considering root cause layout patterns which largely restricts their applicability to real tasks. Cheng *et al.* [8] proposed an advanced solution based on [5]. They take root cause layout patterns into consideration when identifying the correct layout patterns inducing systematic yield loss. In practice, there usually exists complex interactions between different root causes, as well as root cause and systematic defect, but the causal relationship between candidate layout patterns and the systematic defect was not considered in [8].

Another class of seminal works [9], [10] focus more on the geometric structure of layout patterns by adopting clustering algorithms to improve the systematic IC-defect identification. Both connectivity-based and centroid-based clustering algorithms are used to group rotation-, mirror-equivalent layout snippets together. These works conduct clustering on raw layout snippets in a two-stage manner and manually check all possible geometric equivalence between different clusters. Simulation experiments in [10] indicate that the resolution of the identification results of clustering-based methods is limited, since they may require a failure analysis expert's judgment to pick a single layout snippet for each cluster. Furthermore, layout snippets that are shift equivalent are regarded as different candidates, which is commonly considered unreasonable since these snippets share identical or similar geometric structures.

To address the above issues presented in Fig. 1 and improve the resolution of root cause identification, a unified framework

Manuscript received 7 December 2021; revised 25 April 2022; accepted 27 June 2022. Date of publication 19 July 2022; date of current version 21 March 2023. This work was supported in part by Huawei and in part by the Research Grants Council of Hong Kong, SAR, under Grant CUHK14209420 and Grant CUHK14208021. The preliminary version has been presented at the IEEE/ACM International Conference on Computer-Aided Design (ICCAD) in 2021 [DOI: 10.1109/ICCAD51958.2021.9643458]. This article was recommended by Associate Editor L.-C. Wang. *(Corresponding author: Ran Chen.)*

Ran Chen and Bei Yu are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, SAR (e-mail: rchen@cse.cuhk.edu.hk).

Shoubo Hu, Zhitang Chen, Shengyu Zhu, and Jianye Hao are with the Noah's Ark Lab, Huawei, Hong Kong.

Pengyun Li, Cheng Chen, and Yu Huang are with the HiSilicon Research and Development Department, HiSilicon, Shenzhen 518129, China.

Digital Object Identifier 10.1109/TCAD.2022.3192363

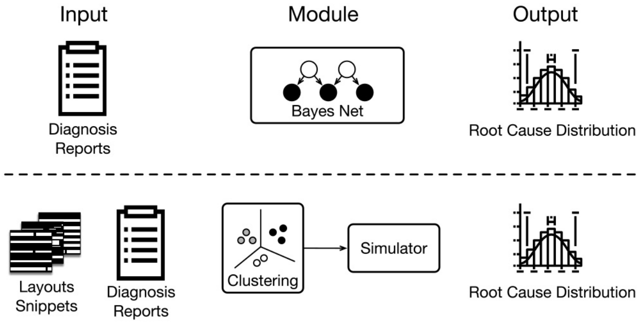


Fig. 1. Overview of prior methods. Upper: [5], [7]; lower: [9], [10].

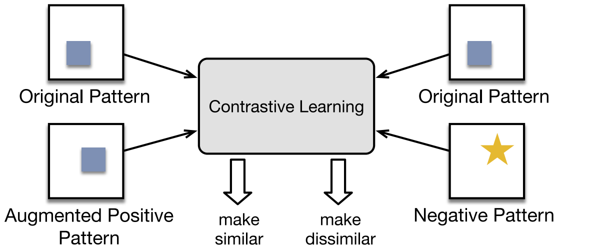


Fig. 2. Idea of contrastive learning: maximize the similarity between latent features of an image and its augmented version and simultaneously minimize the similarity between latent features of inputs correspond to different original images.

for layout pattern analysis (LPA) with deep causal effect estimation is proposed in this work. Compared to existing statistical learning methods, our framework characterizes the causal relationship between potential root causes and the systematic defect. Compared to methods using clustering algorithms, our framework regards rotation-, mirror-, and shift-invariant layout snippets as equivalent without requiring any manual equivalence check. At the first stage of the framework, a novel contrastive learning method is used to train an encoder network to extract from layout snippets their rotation-, mirror-, and shift-invariant latent features. The latent features are then clustered to form layout patterns. Based on the learned layout patterns, we use a structural causal model (SCM) to model the causal relationship between candidate layout patterns and the systematic defect, i.e., the model describes the relationship between the occurrence/presence of a certain layout pattern and the systematic defect. Finally, the average causal effect (ACE) of candidate layout patterns on the systematic defect is estimated as the metric for the identification of the root cause of systematic defects. Experimental results on large-scale designs show that our framework achieves state-of-the-art results which significantly outperforms a commercial tool in terms of accuracy as well as inference time.

To the best of our knowledge, this work is the first to apply contrastive learning-based deep learning techniques and ACE estimation to identify the root cause. The main contributions of this work are threefold.

1) We propose a unified solution to volume diagnosis-based root causes layout pattern identification task. Both pattern clustering and root cause identification are taken into consideration. A novel clustering loss is proposed to solve the limitation of the conventional contrastive learning method. Our framework can identify the critical root causes and provide high-resolution clustered snippets for further analysis.
2) The causal relationship between different candidate layout patterns and the systematic defects is characterized using a neural network and a neural network attribution method is adopted to estimate the ACE for root cause identification.
3) Experimental results on several industrial designs show the effectiveness and robustness against the noise of our framework. The accuracy of our framework outperforms a commercial tool and state-of-the-art framework in different scenarios and we get ×8.4 speedup on average at inference.

The remainder of this article is organized as follows. Section II introduces terminologies and problem formulation related to this work. Section III describes the problem formulation and the algorithmic details of our framework. Section IV lists the experimental results, followed by the discussion and conclusion in Section V.

II. Preliminaries

In this section, preliminary knowledge related to the proposed framework is briefly reviewed.

A. Layout Pattern Analysis

LPA takes a step toward identifying the layout patterns which cause systematic defects. Cheng *et al.* [8] proposed an LPA solution which is an enhanced flow based on [5]. Attribute to the external steps on layout pattern processing, this work makes root cause identification on layout patterns becomes feasible. The challenge of how to handle a large number of potential layout patterns to be considered for analysis is solved. And the risk of overfitting caused by Bayesian modeling is also addressed. Layout pattern extraction is designed to extract all unique layout patterns around locations that could be physical defects. In layout pattern matching, layout patterns are transformed to canonical forms which make shifted, rotated, or mirrored patterns identical. Combining previous steps with the root cause identification method proposed in [5], root cause analysis results, including root cause distribution and layout patterns, are returned for further study.

B. Contrastive Learning

Conventional deep network training often relies on large amounts of annotated data to learn representations in a latent space. Since the annotated data can be costly or even impossible to collect, self-supervised learning leverages unlabeled data to perform pretext tasks for representation learning [11], [12]. Contrastive learning is a class of self-supervised learning that uses contrastive objectives. The general idea of contrastive learning is to maximize the similarity between an instance and its augmentation, while keeping the discriminative power against different instances through a contrastive loss in the latent space, as illustrated in Fig. 2. Recent contrastive learning

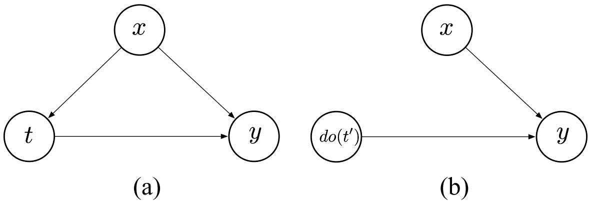


Fig. 3. SCM (a) without and (b) under intervention. Nodes represent random variables and directed edges $x \rightarrow y$ indicate that x is a direct cause of y. Under intervention $do(t')$, the intervened variable t is fixed to the intervened value t' and all its incoming edges are removed.

methods [13]–[16] have achieved competitive results in visual representation learning compared with prominent supervised learning methods for computer vision tasks.

C. Structural Causal Models

SCMs [17] are developed toward a comprehensive theory of causation and serve as a key ingredient of our framework.

Definition 1 (Structural Causal Model [17]): An SCM M is a 4-tuple $[E, X, F, P(E)]$, where

1) E is a set of exogenous (unobserved) variables;
2) X is a set of endogenous (observed) variables;
3) F represents a collection of functions $F = \{f_i\}$ such that each endogenous variable $x_i \in X$ is determined by a function $f_i \in F$, where f_i is a mapping from the respective domain of $\epsilon_i \cup Pa_i$ to x_i, with $\epsilon_i \subseteq E$, $Pa_i \subseteq X \backslash \{X_i\}$ is the set of direct parents of x_i;
4) the uncertainty is encoded through a probability distribution over the exogenous variables, $P(E)$.

SCMs provide a compact way of characterizing ACE $\mathrm{ACE}^{y}_{do(x_i)}$, which is defined as $\mathbb{E}[y|do(x_i = 1)] - \mathbb{E}[y|do(x_i = 0)]$ for binary x_i. $\mathbb{E}[y|do(x_i = \alpha)]$, known as *interventional expectation* [17], denotes the expectation of y when intervening the value of x_i to be α. For an SCM, such intervened model can be represented by replacing the structural equation $x_i = f_i(Pa_i, \epsilon_i)$ by a constant $x_i = \alpha$. A directed acyclic graph representation of SCMs is presented in Fig. 3.

D. Neural Network Attributions

Attribution methods aim to provide interpretability of deep networks by identifying the effect of an input neuron on a specific output neuron [18], [19]. Recently, [20] approached neural network attribution problems from a causal perspective. They view a multilayer perceptron (MLP) $\{l_1, \ldots, l_n\}$ as an SCM $M'(E, \{l_1, l_n\}, f', P(E))$, where l_1 is the input layer, l_n is the output layer, E refers to a set of exogenous random variables which act as causal factors for the input neurons l_1, and f' refers to the mapping from the input to output by marginalizing out all hidden neurons.

Based on SCM reformulation, [20] approximated the interventional expectation of the output neurons $f'(l_1)$ under the intervention $do(x_i = \alpha)$ as

$$\mathbb{E}\big[f'(l_1)|do(x_i = \alpha)\big] \approx f'(\mu) + \frac{1}{2}\mathrm{tr}(\nabla^2 f'(\mu)\mathbb{E}[(l_1 - \mu)(l_1 - \mu)^\top|do(x_i = \alpha)]) \quad (1)$$

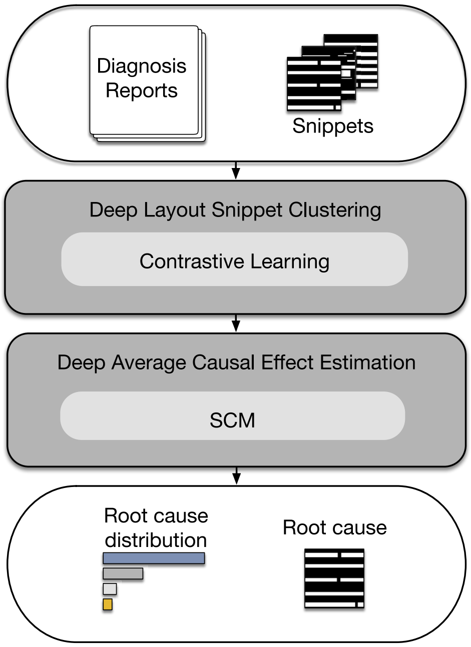


Fig. 4. Overview of the framework.

where tr$(\cdot)$ is the trace operator, $\mu = [\mu_1, \ldots, \mu_k]^\top$ and each entry $\mu_{i'} = \mathbb{E}[x_{i'}|do(x_i = \alpha)]$ $\forall x_{i'} \in l_1$, is the interventional expectation of $x_{i'}$ when x_i is intervened to the value α.

III. Methodologies

A. Overview

The objective of LPA in this work is to identify the true root cause(s) of the systematic defect by analyzing a dataset consisting of m diagnosis reports $R = \{r^e\}_{e=1}^m$ and layout snippets of potential root causes in these reports. Each report r^e consists of several independent symptoms (i.e., defects), whose possible causes are also given along with several important properties (e.g., ID, score, etc.). Our framework identifies the true root cause(s) inducing systematic defects in R by exploiting both the geometric structure of layout snippets (Section III-B) and the causal relationship between potential root causes and systematic defect (Section III-C).

An illustration of our LPA framework is given in Fig. 4. It uses diagnosis reports and layout snippets of potential root causes in these reports as the inputs. First, a contrastive learning-based method is adopted to extract rotation-, mirror-, and shift-invariant latent features from input layout snippets. Then, the latent features are clustered using k-means clustering to identify layout patterns from a large amount of layout snippets. Each cluster corresponds to one layout pattern. Third, a feedforward neural network, which acts as the SCM involves all candidate layout patterns and systematic defect, is trained to maximize the likelihood of the input diagnosis reports given the representation of candidate layout patterns as inputs. After training, the ACEs of layout patterns to systematic defect are evaluated to identify the true root cause(s).

B. Deep Layout Snippet Clustering

The identification of layout patterns from layout snippets using clustering algorithm is elaborated in this section.

Since there are a considerable number of duplicate and equivalent layout snippets in the diagnosis reports, layout pattern matching is usually conducted to recognize layout snippets that are rotated, mirrored, or shifted version of each other as geometrically equivalent. Clustering algorithms are a widely used class of techniques in layout pattern matching. Although applying connectivity-based or centroid-based clustering algorithms on raw layout snippets achieved certain improvements on identifying root causes, they heavily rely on manual design and may have difficulty when generalized to new manufacturing processes.

To circumvent the need of manually designed clustering rules, we introduce deep neural networks in layout pattern matching. Specifically, an encoder network is trained using contrastive learning to extract latent features that are invariant to trivial transformations, such as rotation, mirror, and shift. Besides, the self-supervised nature of contrastive learning allows us to construct a huge amount of training data set by cropping unlabeled layout snippets from the entire layout designs.

Encoder Network: The principle of the encoder network is to transform raw layout snippets into a low-dimensional latent space, in which equivalent layout snippets are mapped to an identical embedding (vector). The low-dimensional embeddings represent the prototypes of layout snippets. The network structure of our model is shown in Fig. 5. “SeparableConv” indicates the depthwise separable convolution layer which is a variant convolution layer widely used in [21] for computation efficiency. “Block A” and “Block B” are two modules with residual connections. Three “Linear” layers are attached to the feature extractor as a bottleneck structure maps 2-D features to embeddings.

Contrastive Learning: Given a batch of embeddings transformed from raw layout snippets by the encoder network, our goal is to make the embeddings of equivalent layout snippets identical, while keeping those of nonequivalent as dissimilar as possible. To achieve this, ℓ_p-norm is used as a metric to measure the dissimilarity between embeddings z_i and z_j of a pair of layout snippets

$$d(z_i, z_j) = \left\| z_i - z_j \right\|_p \tag{2}$$

where p is a real number greater than 1 and is set to $p = 2$ in this work. Conventional contrastive loss [22], [23] based on the metric above is given by

$$L_{\text{con}}(z, \boldsymbol{p}, \boldsymbol{n}) = \max(d(z, \boldsymbol{p}) - d(z, \boldsymbol{n}) + \text{marg}, 0) \tag{3}$$

where marg is a non-negative value indicating an appropriately set margin, and z, $\boldsymbol{p}$, and $\boldsymbol{n}$ are the embeddings of one layout snippet, the embedding of a positive sample of z, and the embedding of a negative sample of z, respectively. marg represents the minimum difference between positive and negative distances that is required for the loss to be zero. During training, positive samples $\boldsymbol{p}$ are getting closer to the anchor embedding z and negative samples $\boldsymbol{n}$ are penalized to be far from anchor embedding. An illustration of contrastive learning with an encoder is shown in Fig. 6.

Clustering Loss: According to (3), the embedding clustering is improved by pair-wise comparison directly. While this method has drawbacks, the property of clusters is overlooked. An example of a small batch size with two clusters is presented in Fig. 7, the pair-wise operation may pull the positive samples away from the center of clusters. The double-headed arrow with a dotted line indicates the penalization term in (3), this term causes two samples away from the center of corresponding clusters. This may cause negative effects on the efficiency of convergence and clustering quality. It will be more difficult for training when the batch size is larger, since more clusters

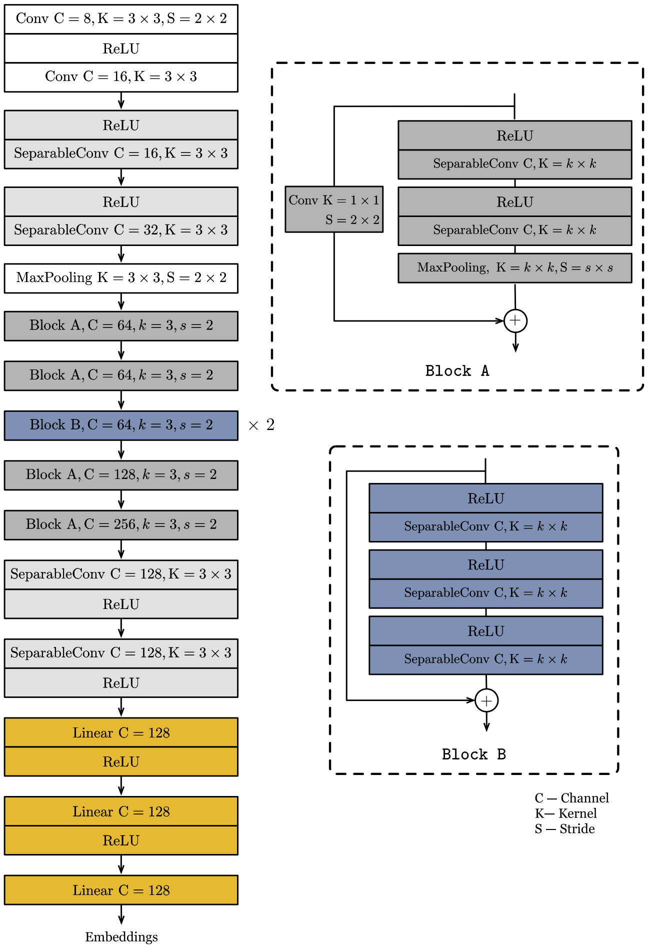

Conv C = 8, K = 3 × 3, S = 2 × 2
ReLU
Conv C = 16, K = 3 × 3
ReLU
SeparableConv C = 16, K = 3 × 3
ReLU
SeparableConv C = 32, K = 3 × 3
MaxPooling K = 3 × 3, S = 2 × 2
Block A, C = 64, k = 3, s = 2
Block A, C = 64, k = 3, s = 2
Block B, C = 64, k = 3, s = 2
× 2
Block A, C = 128, k = 3, s = 2
Block A, C = 256, k = 3, s = 2
SeparableConv C = 128, K = 3 × 3
ReLU
SeparableConv C = 128, K = 3 × 3
ReLU
Linear C = 128
ReLU
Linear C = 128
ReLU
Linear C = 128
Embeddings
Conv K = 1 × 1 S = 2 × 2
ReLU
SeparableConv C, K = k × k
ReLU
SeparableConv C, K = k × k
MaxPooling, K = k × k, S = s × s
Block A
ReLU
SeparableConv C, K = k × k
ReLU
SeparableConv C, K = k × k
ReLU
SeparableConv C, K = k × k
Block B
C — Channel
K— Kernel
S — Stride

Fig. 5. Encoder network structure for contrastive learning. Note that all convolution, separable convolution, and linear layers are followed by batch normalization.

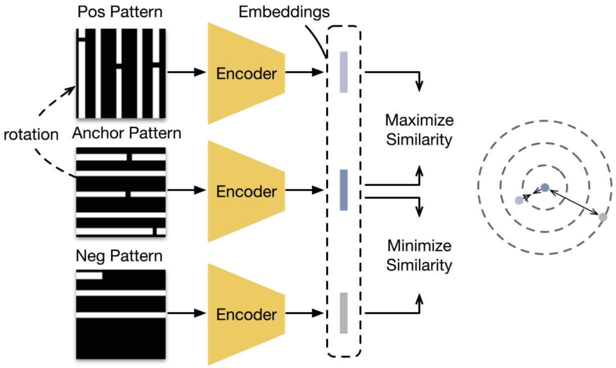


Fig. 6. Illustration of encoder network training using contrastive learning. Parameters of encoders are shared.

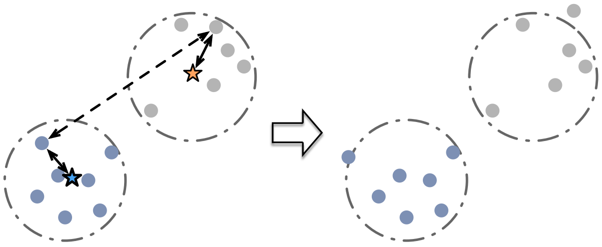


Fig. 7. Drawback of contrastive loss. Pushing away two samples from different clusters may lead to negative effects on resolution.

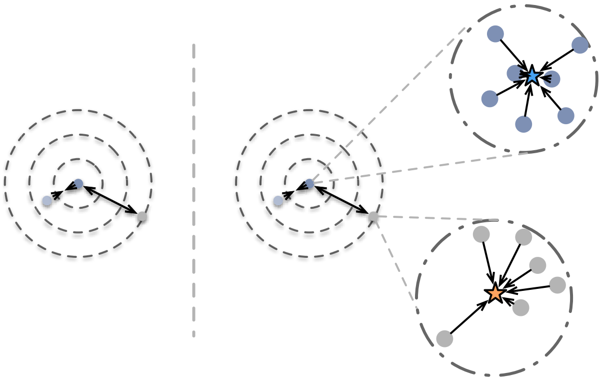


Fig. 8. Contrastive loss in [23] versus clustering loss. The bold stars are the arithmetic mean centers of clusters.

might be disrupted. To tackle this issue, we propose a clustering loss by adding a regularization term to improve the quality of clusters. This term is expressed as

$$R(\boldsymbol{m}_z, z) = \|\boldsymbol{m}_z - z\|_p \tag{4}$$

where m_z is the cluster center of sample z. The whole optimization objective is expressed as

$$\begin{aligned} L(z, \boldsymbol{p}, \boldsymbol{n}, \boldsymbol{m}_p, \boldsymbol{m}_n) &= L_{\text{con}}(z, \boldsymbol{p}, \boldsymbol{n}) \\ &\quad + \alpha R(\boldsymbol{m}_p, \boldsymbol{p}) + \beta R(\boldsymbol{m}_n, \boldsymbol{n}) \end{aligned} \tag{5}$$

where α and β are the weights of regularization terms. With the regularization terms for positive samples and negative samples, the drawback of contrastive loss in [23] is avoided and the risk of low resolution is reduced. Each iteration of the training decreases the quantization error of the clusters since for the same cluster, we have the following inequality:

$$\sum_i R\big(\boldsymbol{m}', z_i\big) - R(\boldsymbol{m}, z_i) = \sum_i \big\|\boldsymbol{m}' - z_i\big\|_p - \|\boldsymbol{m} - z_i\|_p \leq 0$$

where $\boldsymbol{m}'$ is the updated cluster center and $\boldsymbol{s}_i$ is the sample i belongs to the cluster. Repeatedly replacing $\boldsymbol{m}$ by $\boldsymbol{m}'$ speeds up the convergence of the training schedule. Experimental results verified that the proposed clustering loss requires less training data and computation resources. The advantage of clustering loss is summarized in Table I. A visualized illustration of (5) is presented in Fig. 8. The detail of how to construct the positive and negative samples and training scheme is clarified in Section IV.

After training the encoder network, it is used to extract embeddings of layout snippets. Then, the k-means clustering algorithm is applied to these embeddings to partition them into k clusters C_i, $i \in \{1, \ldots, k\}$. Each cluster C_i consists of n_i equivalent layout snippets that correspond to one layout pattern. The silhouette method [24] which is a measurement of how similar an object is to its own cluster compared to other clusters is adopted to determine the optimal value of the number of clusters k. By embedding clustering, equivalent layout snippets can be grouped into the same cluster without artificial modulation.

TABLE I
ADVANTAGE OF CLUSTERING LOSS COMPARED TO CONTRASTIVE LOSS USED IN [23]

	Cluster Awareness	Training Effciency	Low Data Requirement
Ours	✓	✓	✓
[23]	×	×	×

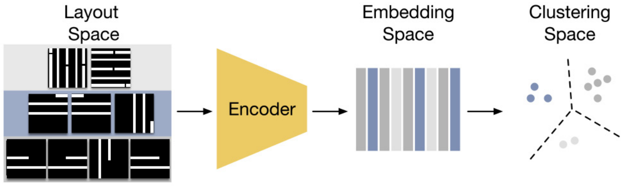


Fig. 9. Example of DLSC.

An example of the deep layout snippet clustering (DLSC) is illustrated in Fig. 9, layout snippets with a large number of pixels are transformed to low-dimensional embeddings which reduces the clustering computation remarkably while improving the layout pattern matching accuracy. Experimental results in Section IV empirically show that encoder network trained using layout snippets of one layout design can also generalize to new layout designs.

C. Deep Average Causal Effect Estimation

In this section, we introduce how we use ACE estimation to identify the true root cause(s) from a large amount of potential root causes using diagnosis reports and the results of layout pattern matching.

Defect SCM Training: Based on the clustering results, we transform the embeddings of layout snippets to the cluster space and then build the SCM between candidate layout patterns and systematic defect. First, distance matrix $\mathbf{D} \in \mathbb{R}^{n \times k}$ is computed, whose (j, i)th entry $[\mathbf{D}]_{j,i}$ denotes the distance of the jth embedding to the center of cluster i. Then, the distance matrix is converted to a cluster membership matrix $\mathbf{P} \in \mathbb{R}^{n \times k}$ whose entries indicate the probability of each embedding belonging to each cluster as follows:

$$[\mathbf{P}]_{j,i} = \frac{\exp\left(-\mathbf{D}_{j,i}/\tau\right)}{\sum_{i'} \exp\left(-\mathbf{D}_{j,i'}/\tau\right)} \tag{6}$$

where τ is a temperature parameter, set as 0.1 in this work. The layout snippets closer to the cluster center have higher probabilities.

With all layout snippets represented in the form of membership vectors in $\mathbf{P}$, we model the SCM between candidate layout patterns and systematic defect with an MLP $\mathcal{M}$ to characterize their causal relationship. MLP as a neural network can be regarded as directed graphs with directed edges from a lower layer to the layer above. The final output is based on the hierarchy of interactions between lower level nodes.

Proposition 1: Given an l-layer feedforward neural network $N(l_1, l_2, \ldots, l_n)$, where l_i is the set of neurons in layer i has a corresponding SCM $M(X, [l_1, l_2, \ldots, l_n], [f_1, f_2, \ldots, f_n], P(E))$, where l_1 is the input layer and l_n is the output layer. Corresponding to every l_i, f_i refers to the set of causal functions for neurons in layer i.

Proof: In a feedforward neural network, each layer neuron can be regarded as functions of neurons in its previous layer, i.e., $\forall i \in l\colon \forall l_{i_j} \in l_i\colon l_{i_j} = f_{i_j}(l_{i-1})$. The input layer l_1 can be assumed to be functions of exogenous variables E such that $l_{1_i} = f_{1_i}(e_i) \forall l_{1_i} \in l_1$ and $e_i \in E$. This structure in the random variables, neurons in the network, can be equivalently expressed by an SCM $M(X, [l_1, l_2, \ldots, l_n], [f_1, f_2, \ldots, f_n], P(E))$. ■

The causal structure can be reduced to SCM $M^{'}(X, [l_1, l_n], f^{'}, P(E))$ by marginalizing out hidden neurons, since only the neurons in layer l_1 and layer l_n are observables.

Corollary 1: Every l-layer feedfoward neural network $N(l_1, l_2, \ldots, l_n)$ with l_i denoting the set of neurons in layer i, has a corresponding SCM $M(X, [l_1, l_2, \ldots, l_n], [f_1, f_2, \ldots, f_n], P(E))$ which can be reduced to an SCM $M^{'}(X, [l_1, l_n], f^{'}, P(E))$.

Proof: Starting with each neuron l_{n_i} in the output layer l_n, the corresponding causal function $f_{n_i}(l_{n-1})$ can be substituted as $f_{n_i}(f_{n-1_1}(l_{n-2}), f_{n-1_2}(l_{n-2}), f_{n-1_3}(l_{n-2}), \ldots, f_{n-1_{|l_{n-1}|}}(l_{n-2}))$. This can be written as $l_{n_i} = f^{'}_{n_i}(l_{n-2})$. f_{i_j} refers to the causal function of neuron j in layer i and l_{i_j} refers to neuron j in layer i. Proceeding recursively layer by layer, we have modified functions such that, $\forall l_{n_i} \in l_n\colon l_{n_i} = f^{'}_{n_i}(l_1)$. The causal mechanisms set $f^{'}$ of the reduced SCM $M^{'}$ would be $\{f^{'}_{n_i} | l_{n_i} \in l_n\} \cup \{l_{1_i} = f_{l_i}(e_i) | l_{1_i} \in l_1$ and $e_i \in E\}$. ■

With Proposition 1 and Corollary 1, we can simplify the SCM as $\mathcal{M}$ and concentrate on the input and output layers of $\mathcal{M}$ in the following procedures. The input layer l_1 of $\mathcal{M}$ has k input neurons x_i, $i \in \{1, \ldots, k\}$, each of which corresponds to one layout pattern. Its output layer l_n has one output neuron indicating the probability of systematic defect. The objective function for training the $\mathcal{M}$ is

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{e=1}^{m} \log\left[\sum_{i} p(r^e|y_i) p(y_i|\boldsymbol{\mu}_i, \boldsymbol{\theta})\right] \tag{7}$$

where m is the number of diagnosis reports, $\boldsymbol{\theta}$ denotes the parameters of $\mathcal{M}$, $\boldsymbol{\mu}_i = (1/n_i) \sum_{j \in C_i}^{n} \mathbf{P}_{j,:}$ is the mean representation of cluster i with n_i layout snippets, $p(y_i|\boldsymbol{\mu}_i, \boldsymbol{\theta})$ is the output of $\mathcal{M}$ corresponding to layout pattern x_i, which indicates the probability of layout pattern x_i inducing the systematic defect, $p(r^e|y_i)$ is the conditional probability of diagnosis report r^e if layout pattern x_i occurs. We train the neural network $\mathcal{M}$ by minimizing the negative log likelihood in (7). The detail on estimating $p(r^e|y_i)$ is elaborated in Section IV.

LPA by ACE Estimation: After the objective in (7) converges, $\mathcal{M}$ is viewed as an SCM containing candidate layout patterns and systematic defect. We assume that the true root cause has the most significant ACE on the systematic defect. Therefore, ACEs of input neurons (corresponding to layout patterns) on the output neuron (corresponding to the systematic defect) are estimated as the metric to identify root causes. Causal neural attribution in (1) is adopted to compute the ACE of each layout pattern on the systematic defect.

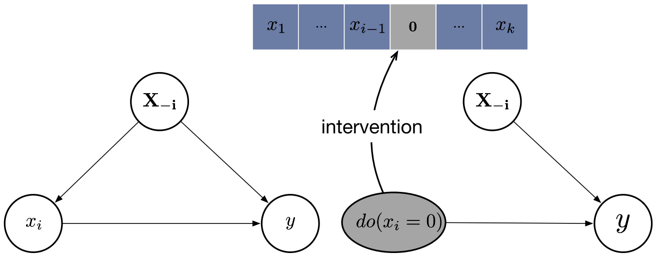


Fig. 10. Left: The defect SCM for LPA without intervention. Right: Apply intervention on cluster i.

When training the $\mathcal{M}$, the inputs are representations obtained from the membership matrix $\mathbf{P}$ whose entries are continuous values between [0, 1]. Since entries in $\mathbf{P}$ indicate the probability of a layout snippet belonging to a certain cluster, they have monotonic property, i.e., the closer (j, i)th entry is to 1 (resp., 0), the higher (resp., lower) the probability of the jth layout snippet belonging to cluster i is. As a result, when regarding x_i as a binary variable, the ACE of x_i on y characterizes the causal effect of the presence of layout pattern x_i on the systematic defect. This ACE can be estimated as

$$\text{ACE}^{y}_{do(x_i)} = |\mathbb{E}\,[y|do(x_i = 0)] - \mathbb{E}\,[y|do(x_i = 1)]|. \tag{8}$$

The interventional expectation when x_i is intervened to 0 in (8) can be estimated using (1) as

$$\begin{aligned}\mathbb{E}[y|do(x_i = 0)] &\approx f'(\boldsymbol{\mu}_{i0})\\ &+ \frac{1}{2}\text{tr}\Big(\nabla^2 f'(\boldsymbol{\mu}_{i0}) \mathbb{E}\Big[\big(l_{in} - \boldsymbol{\mu}_{i0}\big)\big(l_{in} - \boldsymbol{\mu}_{i0}\big)^{\top} | do(x_i = 0)\Big]\Big)\end{aligned} \tag{9}$$

where f' refers to the mapping from the input of $\mathcal{M}$ to its output y, the vector of interventional expectation $\boldsymbol{\mu}_{i0}$ is obtained by intervening the value of the ith entry of $\boldsymbol{\mu}_i$ to 0. Similar steps apply for the computation of the interventional expectation when x_i is intervened to 1. After obtaining the ACE of all layout patterns on systematic defect, we normalize them to form the root cause distribution of all candidate layout patterns as

$$p(x_i) = \frac{\text{ACE}^{y}_{do(x_i)}}{\sum_{i'} \text{ACE}^{y}_{do(x_i')}}. \tag{10}$$

D. Inference Flow

An overview of this unified framework and modules of DLSC and deep average causal effect estimation (DACE) are introduced in Sections III-B and III-C. Here, we give a detailed explanation on the inference flow of the framework.

The pseudocode of the inference flow is presented in Algorithm 1, lines 1–7 correspond to the inference steps of

Algorithm 1 Inference Flow of the Framework

```
Input: $R = \{r^e\}_{e=1}^m$ - a set of diagnosis reports, $S = \{s_j\}_{j=1}^n$ -
    a set of layout snippets of all potential root causes;
Output: Root cause distribution
 1: for $d = 1 \to L$ do
 2:     for $j = 1 \to |S|$ do
 3:         Encoder($s_j$) $\to z_j, \forall j \in d$;            ▷ Equation 5
 4:     end for
 5:     Get optimal $k_d$ with highest silhouette score;
 6:     Compute distance matrix $\mathbf{D}_d$ using optimal $k_d$;
 7: end for
 8: Construct $\mathbf{D}$ by concatenating $\mathbf{D}_d, \forall d \in \{1, \dots, L\}$;
 9: Convert $\mathbf{D}$ to $\mathbf{P}$;                              ▷ Equation 6
10: Train defect SCM $\mathcal{M}(\boldsymbol{\theta})$;                  ▷ Equation 7
11: for $i = 1 \to k$ do
12:     Calculate ACE of clusters;                      ▷ Equation 8
13: end for
14: return Root cause distribution;                     ▷ Equation 10
```

TABLE II
NOTATION ON DIAGNOSIS REPORT FEATURES

Feature	Description
rule_id	ID of the potential root cause
s_j^e	The score of potential root cause j in r^e
h_j^e	DFM hits of potential root cause j in r^e
v_j	DFM violations of potential root cause j
$\langle x_j, y_j \rangle$	Coordinate of potential root cause j
layer	Layer name of current potential root cause
type	Defect category of current potential root cause

TABLE III
LAYOUT DESIGN INFORMATION

	Size ($\mu m \times \mu m$)	#Layers	#Gates
Case 1	8881 × 9328	5	9337
Case 2	429 × 384	9	1560k
Case 3	8033 × 7822	6	9278k
Case 4	1091 × 1304	8	4176k
Case 5	2300 × 2410	9	5598k
Case 6	1483 × 1736	7	455k

TABLE IV
DEFECT INJECTION STATISTICS

	#TotalInjections	#Open	#Bridge
Case 1	68	50	18
Case 2	107	72	35
Case 3	93	69	24
Case 4	44	28	16
Case 5	25	18	7
Case 6	39	28	11
Case 2 noise*	963	648	315
Case 3 noise	736	544	192
Case 4 noise	221	145	76
Case 5 noise	176	125	51
Case 6 noise	429	356	73

* We increase the number of injection experiments which is greater than the statistics presented in [23].

DLSC, and lines 8–12 represent the procedure of DACE. First, layout snippets are transformed to embeddings in a latent space and clustered within each layer with the dedicated optimal cluster number k_d, respectively. Layer-wise clustering is performed due to the consideration of process variance of different layers and efficiency. Second, the membership matrix of layout snippets in each layer is computed and different matrices from all layout layers are concatenated along axis 0 to form $\mathbf{P}$. An SCM $\mathcal{M}$ is then learned using the information in diagnosis reports. Finally, the ACE of each cluster on the systematic defect is estimated according to (8). The root causes are identified based on the estimated ACE of all candidate layout patterns.

IV. EXPERIMENTAL RESULTS

We evaluate the effectiveness of our proposed framework by testing its root cause identification accuracy on six noise-free datasets from six different layout designs, 40 noisy datasets from five layout designs, and 50 mixture datasets from five layout designs. The advantage of our framework over compared methods in runtime is also validated by all related experiments. The accuracy is defined as the percentage of datasets that the real root cause is identified.

A. Datasets

Encoder Training: We crop layout snippets according to the point of interests (POI) from the layout design in case 2. Their size is determined by the pitch size in the corresponding layer. Layout snippets in other designs are not used during the training. We find out that the trained encoder can be applied to new designs directly without sacrificing clustering quality. Each cropped layout snippet is rotated, mirrored, and shifted to generate positive samples for itself. Samples corresponding to different original layout snippets are deemed negative samples in contrastive learning.

LPA: We follow the same steps of defect injection performed in [5] to generate diagnosis reports. It requires around 1–2 h to generate reports for one injection experiment. Defects of type *Open* and *Bridge* are considered in the injection steps. The detailed information within diagnosis reports is listed in Table II. Three classes of datasets are considered in our evaluation.

1) *Noise-Free Dataset:* Besides three different layout designs presented in [23], three more layout designs are used to construct noise-free datasets. Basic information of six designs is shown in Table III. Cases 2–6 are five real silicon datasets that result from real in-production IC. Diagnosis reports in one noise-free dataset share one single true root cause of the systematic defect. The data of each layout design consists of #TotalInjections noise-free datasets and both open and bridge types are considered in our experiments (Table IV).
2) *Noisy Dataset:* A certain percentage of diagnosis reports in the dataset shares a single true root cause and the remaining diagnosis reports have root causes different from the true one (i.e., noise). Different percentages of

noise are considered during the process of injections and the sources of noise are randomly sampled from the entire layout designs among all metal layers.

3) *Mixture Dataset:* Diagnosis reports in this dataset are divided into four portions. Each portion shares one root cause independently. Three portions of root causes are true and the rest portion is different from the true one. For example, proportion "50-20-20-10" in Table VII means 50%, 20%, and 20% of diagnosis reports have three true root causes correspondingly and 10% of diagnosis reports have random noise.

Besides the diagnosis reports, layout snippets of potential root causes in these reports are another required inputs of our framework. Note that in the noisy dataset, both the number of injections and types of defects are the same across different noise levels. We merged them as "Case # noise" in Table IV.

B. Implementation Details

The proposed framework is implemented in Python with PyTorch library [25]. The encoder network is trained using four Nvidia Tesla V100 GPUs. The SGD optimizer is adopted with initial learning rate 1e−1, weight decay 5e−4, and momentum 0.9. The batch size, number of epochs, and margin *marg* in (3) are set to 64, 16, and 1.5 in the experiments, respectively. Regularization weights α and β in (5) are set as 1 in all experiments. Following [15], we add a single linear layer before the output of the encoder during training to avoid the feature collapse problem.

When conducting LPA, the defect SCM $\mathcal{M}$ is trained using the SGD optimizer with initial learning rate 1e−2, weight decay 1e−3, and momentum 0.9. The maximum number of epochs of model training is set as 100 and the training will be early stopped if there is no improvement of the training loss in consecutive 5 epochs. The conditional probability of diagnosis report r^e if layout pattern x_i is true is calculated as

$$p\left(r^e|y_i\right) = \frac{h^e_{j*}s^e_{j*}}{v_{j*}}\mathbb{1}_{\{s|s\geq 90\}}\left(s^e_{j*}\right) \tag{11}$$

where $j* = \arg\max_{j\in C_i}\sum_e s^e_j\mathbb{1}_{\{s|s\geq 90\}}(s^e_j)$, $\mathbb{1}_{\{s|s\geq 90\}}(s^e_j)$ is an indicator function which evaluates to 1 if $s^e_j \geq 90$ and 0 otherwise.

C. Results and Analysis

We compare the proposed framework `eLPA-DCE` with `LPA-DCE` [23] and an industry-leading commercial tool. One Nvidia Tesla V100 GPU is used for inference. The DFM hits are the number of a potential root cause appearing in the diagnosis report, more DFM hits indicate the layout snippet is more likely to be the root cause to a certain extent. To justify the necessity of our causality-based approach, a diagnosis statistical approach is presented as the baseline. The baseline approach finds the root cause in the following steps: 1) given a volume of diagnosis reports, collect the DFM hits and DFM violations of potential root causes; 2) calculate the ratio between DFM hits and DFM violations of layout snippets and get the mean ratio within each cluster; and 3) the cluster with the top rank of ratio is regarded as the root cause predicted by the diagnosis reports.

TABLE V
ACCURACY(%) ON NOISE-FREE DATASETS

Dataset	`Baseline`	`Commercial Tool`	`LPA-DCE`[23]	`eLPA-DCE`
Case 1	25.00	98.53	100.00	100.00
Case 2	55.88	92.52	98.04	98.04
Case 3	58.06	98.92	98.92	100.00
Case 4	43.71	72.09	97.67	100.00
Case 5	39.25	82.05	91.30	100.00
Case 6	56.29	84.62	94.87	89.74
Average	46.37	88.12	96.80	**97.96**

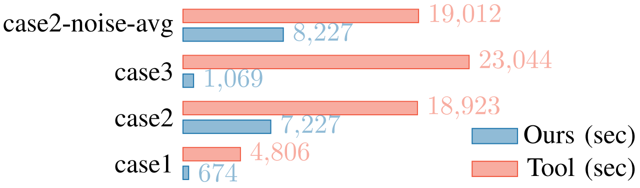


Fig. 11. Inference speed comparison between our framework and the commercial tool.

On Noise-Free Datasets: As shown in Table V, our method outperforms the commercial tool 9.84% on average under the setting of noise-free defect injection. The average accuracy of the baseline is 46.37% which is much lower than our method and the commercial tool. This indicates that by using simple statistics according to the diagnosis report it is hard to locate the root cause accurately. With the new clustering loss, we got 1.16% improvement compared to `LPA-DCE` [23]. This means we get a better quality on the membership matrix with the proposed regularization.

On Noisy Datasets: When given more challenging tasks of root cause identification, we observe that the performance of baseline becomes worse when the noise level is higher. Our proposed method can estimate the root cause with better performance under different noise levels (see Table VI). The average accuracy of the framework is 40.01% higher than the commercial tool. In case 5, the commercial tool cannot identify the root cause under the noise perturbation. Our method can identify the root cause with robust performance across different noise levels. For other cases, especially when the ratio of noise is greater than 70%, it is difficult for the commercial tool to identify the root cause precisely. While our framework locates the root cause with higher accuracy than the commercial tool. For example, the performance of case 3 and case 4 of the commercial tool at 80% noise level is 26.88% and 14.29% and ours are 100% and 76%. The proposed method is robust to the injection noise and it also outperforms `LPA-DCE` [23] in most cases.

On Mixture Datasets: We conduct the mixture root causes identification experiments to test whether our framework can be extended to multiple root causes scenario. The experimental results in Table VII show that the proposed framework has competitive results compared to the commercial tool especially in tasks of identifying three true root causes from the mixture dataset, while the commercial tool may provide misleading results which cannot be used for further study. Especially

TABLE VI
ACCURACY(%) ON NOISY DATASETS

Noise(%)	Commercial Tool					LPA-DCE[23]					eLPA-DCE				
	Case 2	Case 3	Case 4	Case 5	Case 6	Case 2	Case 3	Case 4	Case 5	Case 6	Case 2	Case 3	Case 4	Case 5	Case 6
80	70.09	26.88	14.29	-	17.95	94.32	95.70	64.00	100	86.67	96.59	100.00	76.00	100.00	75.86
70	88.79	44.09	35.71	-	28.21	97.17	92.47	94.59	92.00	92.31	98.11	96.77	94.59	96.00	89.74
60	93.46	52.69	50.00	-	46.15	92.52	97.85	95.12	92.00	94.87	93.46	97.85	95.12	92.00	89.74
50	93.46	69.89	52.38	-	64.10	87.85	98.92	88.10	96.00	89.74	91.59	97.85	95.24	92.00	87.18
40	93.46	89.25	57.14	-	82.05	87.85	95.70	95.24	100.00	89.74	89.72	97.85	95.24	95.24	79.49
30	93.46	90.22	61.90	-	74.19	91.59	94.57	100.00	89.74	96.77	86.92	94.57	97.62	84.21	90.32
20	92.52	90.11	64.29	-	78.57	94.39	98.90	97.62	100.00	92.86	99.07	98.90	100.00	94.44	92.86
10	93.46	97.73	66.67	-	84.62	99.07	100.00	100.00	94.44	100.00	99.07	100.00	100.00	100.00	92.31
Average	89.84	70.11	50.30	-	59.48	93.10	96.76	91.83	**95.49**	**92.87**	**94.32**	**97.97**	**94.23**	94.24	87.19

TABLE VII
ACCURACY(%) ON MIXTURE DATASETS

Proportion (r1%-r2%-r3%-noise%)	Commercial Tool					LPA-DCE[23]					eLPA-DCE				
	Case 2	Case 3	Case 4	Case5	Case6	Case 2	Case 3	Case 4	Case5	Case6	Case 2	Case 3	Case 4	Case5	Case6
20-20-20-40	62.07	8.05	29.21	-	13.83	77.19	78.02	64.00	79.37	86.32	85.96	93.41	84.00	74.60	87.37
30-20-20-30	65.91	19.28	29.03	-	19.57	82.35	89.01	82.61	72.60	83.15	80.88	92.31	78.26	79.45	85.39
30-30-20-20	71.74	38.82	32.22	-	41.94	80.23	86.81	78.26	84.34	91.11	93.02	92.31	69.57	84.34	88.89
30-30-30-10	79.17	74.42	40.00	-	45.83	89.47	93.41	92.31	87.88	87.00	86.32	96.70	88.46	87.88	92.00
40-20-20-20	70.79	27.06	30.00	-	38.71	84.51	85.71	69.23	81.43	91.11	84.51	91.21	69.23	74.29	84.44
40-30-20-10	69.15	62.79	35.23	-	45.45	82.76	83.52	84.09	92.05	91.30	88.51	98.90	84.09	95.45	92.39
40-30-30-0	83.51	81.32	42.42	-	55.00	91.84	96.70	89.19	89.00	87.00	92.85	96.70	86.49	89.00	95.00
50-20-20-10	70.65	61.63	31.33	-	32.63	72.46	90.11	78.79	70.27	89.16	81.16	91.21	75.76	70.27	91.57
50-30-20-0	73.12	84.27	40.22	-	46.00	83.13	91.21	90.24	83.70	95.45	85.54	96.70	80.49	92.39	98.86
60-20-20-0	72.22	74.71	37.08	-	37.11	81.54	82.42	83.78	68.57	86.90	78.46	89.01	81.08	81.43	94.05
Average	71.83	53.24	34.67	-	37.61	82.55	87.69	**81.25**	80.92	88.85	**85.72**	**93.85**	79.74	**82.91**	**91.00**

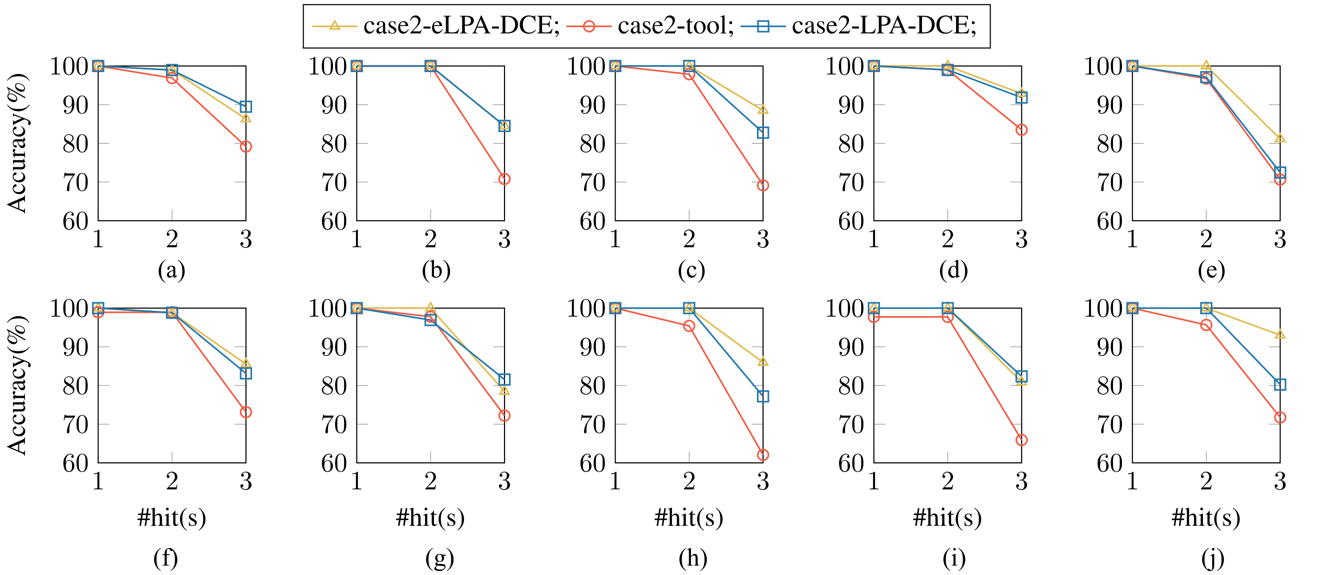


Fig. 12. Accuracy of identifying 1–3 true root causes in top-3 layout patterns on mixture dataset case 2. (a) 30-30-30-10. (b) 40-20-20-20. (c) 40-30-20-10. (d) 40-30-30-0. (e) 50-20-20-10. (f) 50-30-20-0. (g) 60-20-20-0. (h) 20-20-20-40. (i) 30-20-20-30. (j) 30-30-20-20.

when the percentage of root causes is low, e.g., proportion "20-20-20-40," the causal-based methods are much more reliable, since it is more frequent we have a low percentage of root causes. Similar to the results of the commercial tool presented in Table VI, the root causes of case 5 cannot be identified. We infer that the pitch sizes of patterns used in the commercial tool are different and this might be the reason why the identification of root causes of this case failed. Our framework achieves 82.91% high accuracy and got 47.17% better on average than the commercial tool. eLPA-DCE also got better results on four out of five designs than the LPA-DCE [23]. The major advantage of eLPA-DCE is owed to the advantage of clustering loss, the causation among snippets becomes more identifiable based on the clusters with better quality.

Top-3 Accuracy on Mixture Datasets: We analyze the performance of identifying 1–3 true root causes from the mixture datasets. Since there are three true root causes in each dataset, a true root cause belonging to one of top-3 layout pattern in the root cause distribution is a successful identification. The accuracies of identifying 1–3 true root causes are shown in Figs. 12–16, our method can identify at least one root cause in all cases while the commercial tool fails to achieve 100%

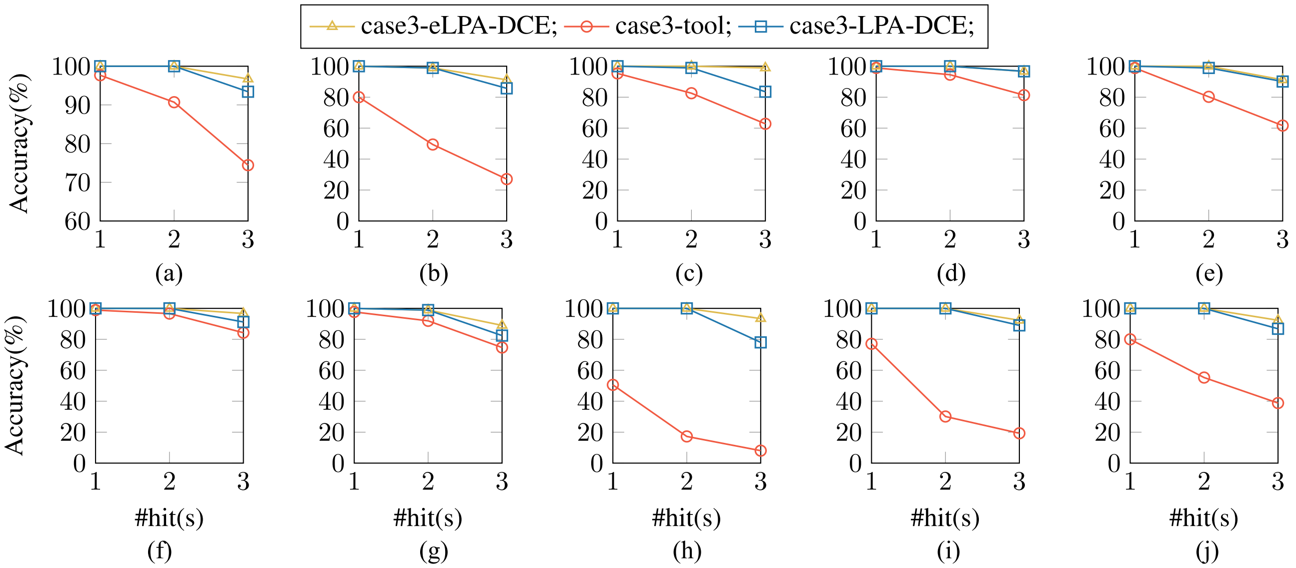


Fig. 13. Accuracy of identifying 1–3 true root causes in top-3 layout patterns on mixture dataset case 3. (a) 30-30-30-10. (b) 40-20-20-20. (c) 40-30-20-10. (d) 40-30-30-0. (e) 50-20-20-10. (f) 50-30-20-0. (g) 60-20-20-0. (h) 20-20-20-40. (i) 30-20-20-30. (j) 30-30-20-20.

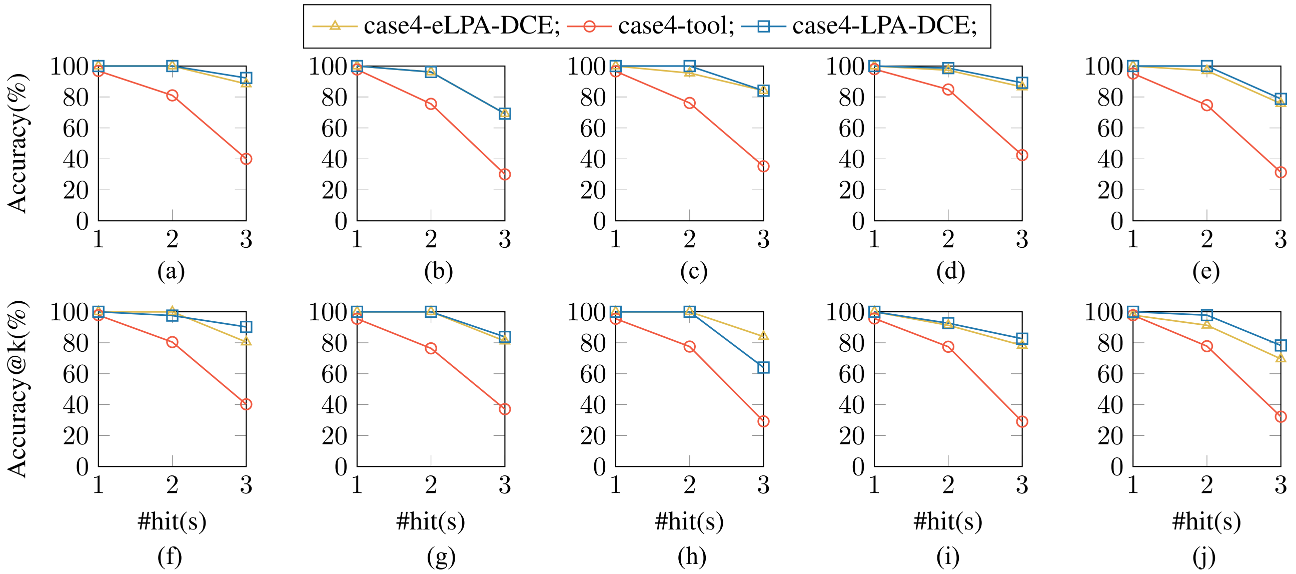


Fig. 14. Accuracy of identifying 1–3 true root causes in top-3 layout patterns on mixture dataset case 4. (a) 30-30-30-10. (b) 40-20-20-20. (c) 40-30-20-10. (d) 40-30-30-0. (e) 50-20-20-10. (f) 50-30-20-0. (g) 60-20-20-0. (h) 20-20-20-40. (i) 30-20-20-30. (j) 30-30-20-20.

accuracy. Also, the average accuracy of identifying two root causes of our method is greater than 95%, which is around 30% better than the commercial tool.

Comparing the performance of `eLPA-DCE` and `LPA-DCE` [23], the root cause distributions generated with the encoder trained by clustering loss are more concentrated than the ones trained with conventional contrastive loss. Dense and recognizable clusters are beneficial to identify the root causes especially if the number of clusters is large.

Inference Speed: The inference time of our framework and the commercial tool on single root cause datasets are shown in Fig. 11. The speed of the proposed deep learning framework method surpasses the conventional commercial tool by a large margin. We got around ×10.4 and ×2.3 speedup on noise-free datasets and noisy datasets. The robustness of accuracy and inference speed indicate our method is valuable for the industry.

Clustering Quality: We compare the clustering quality of DLSC with directly applying the k-means algorithm and DBSCAN algorithm on raw layout snippets using the adjusted rand index (ARI $\in [-1, 1]$, [26]). ARI computes a similarity measure between two partitions by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings. High ARI indicates a good match between the clustering results and the ground truth. In LPA, high ARI scores indicate high resolution. The experiments are conducted on cases 2–6, 256 layout snippets are sampled layer-wise of each case for evaluation. Note that these cases are not available during the training of the encoder network. The ARIs of clustering using raw layout

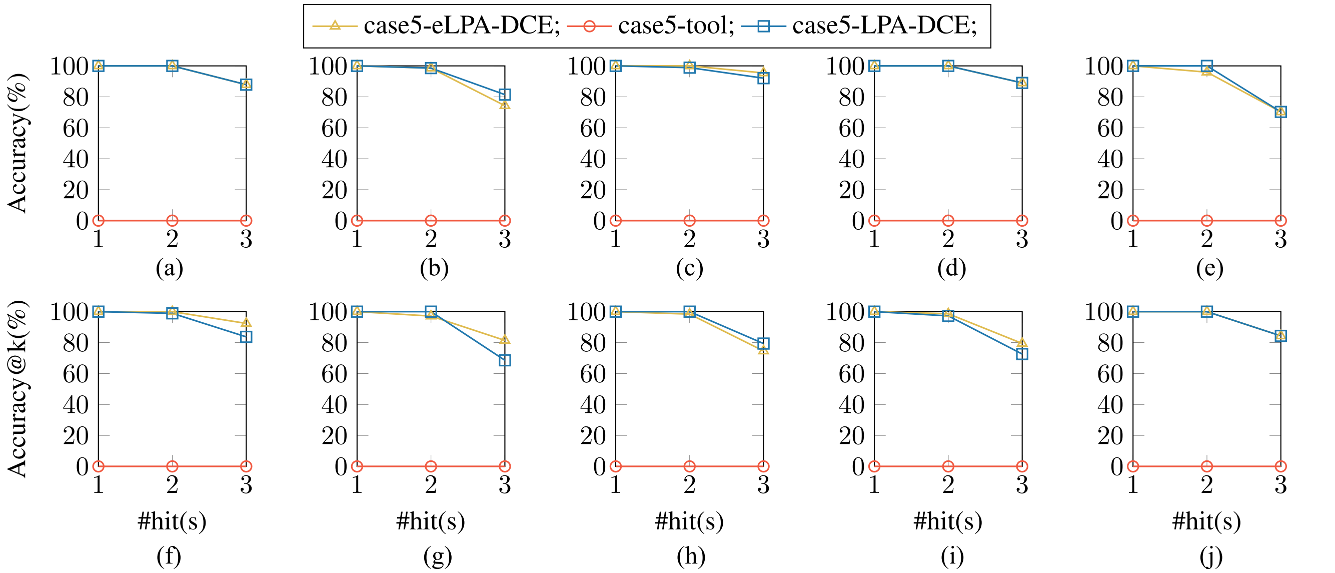


Fig. 15. Accuracy of identifying 1–3 true root causes in top-3 layout patterns on mixture dataset case 5. (a) 30-30-30-10. (b) 40-20-20-20. (c) 40-30-20-10. (d) 40-30-30-0. (e) 50-20-20-10. (f) 50-30-20-0. (g) 60-20-20-0. (h) 20-20-20-40. (i) 30-20-20-30. (j) 30-30-20-20.

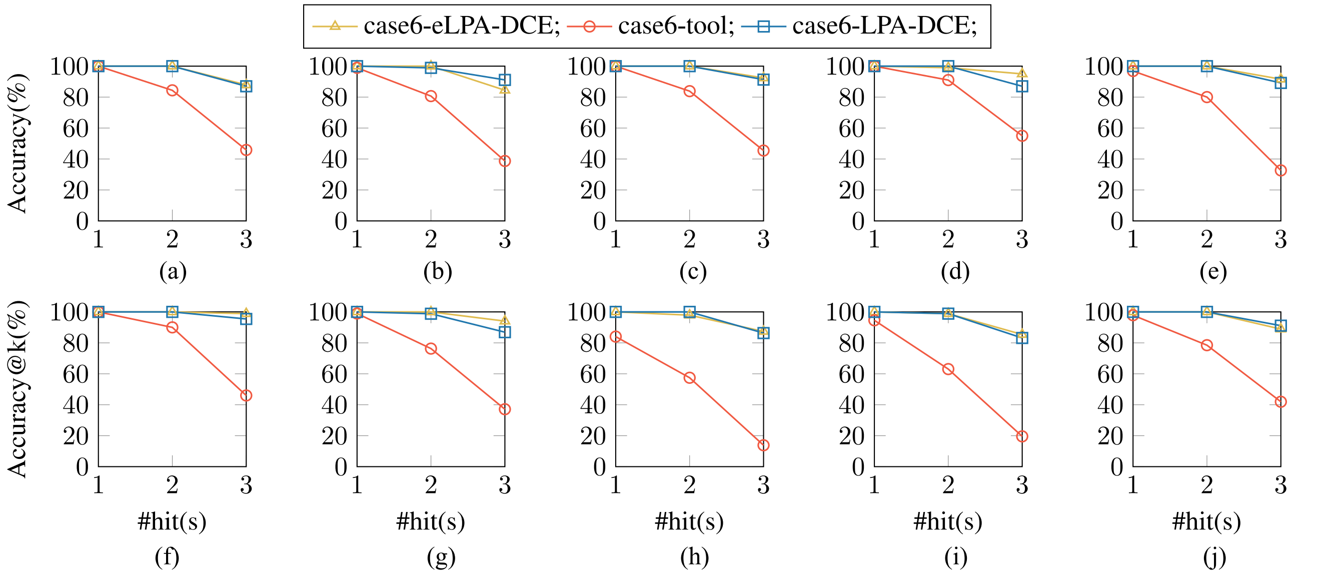


Fig. 16. Accuracy of identifying 1–3 true root causes in top-3 layout patterns on mixture dataset case 6. (a) 30-30-30-10. (b) 40-20-20-20. (c) 40-30-20-10. (d) 40-30-30-0. (e) 50-20-20-10. (f) 50-30-20-0. (g) 60-20-20-0. (h) 20-20-20-40. (i) 30-20-20-30. (j) 30-30-20-20.

snippets, embeddings in `LPA-DCE` [23], and embeddings in this work presented in Fig. 17 are the mean of ten independent evaluations. The ARI scores of embeddings with the clustering loss outperform the conventional contrastive learning method presented in [23] in all metal layers. Also, both the embedding-based clustering get higher ARI scores than the raw-based clustering. This indicates that the contrastive learning can improve the quality of layout pattern matching. The clustering loss can make it perform better since it avoids the drawback of the primitive method.

Advantage of Clustering Loss on Encoder Training: The number of layout snippets used to train encoder with contrastive learning proposed in `LPA-DCE` and the clustering loss presented in this work is shown in Fig. 18. Only 4×10^5 samples are required to train the encoder in this work, which is 10× lower than the number of samples used in `LPA-DCE` [23]. The budget for training data and computation resources is much less. Regularization with clustering loss lower the requirements of training set scales and speed up the procedure of model training, which reduces the budgets of retraining on new designs with different technological process.

V. Discussion and Conclusion

In this article, a unified framework on LPA is proposed to identify the root cause(s). Based on the concept of contrastive learning, a DLSC module is designed to solve the

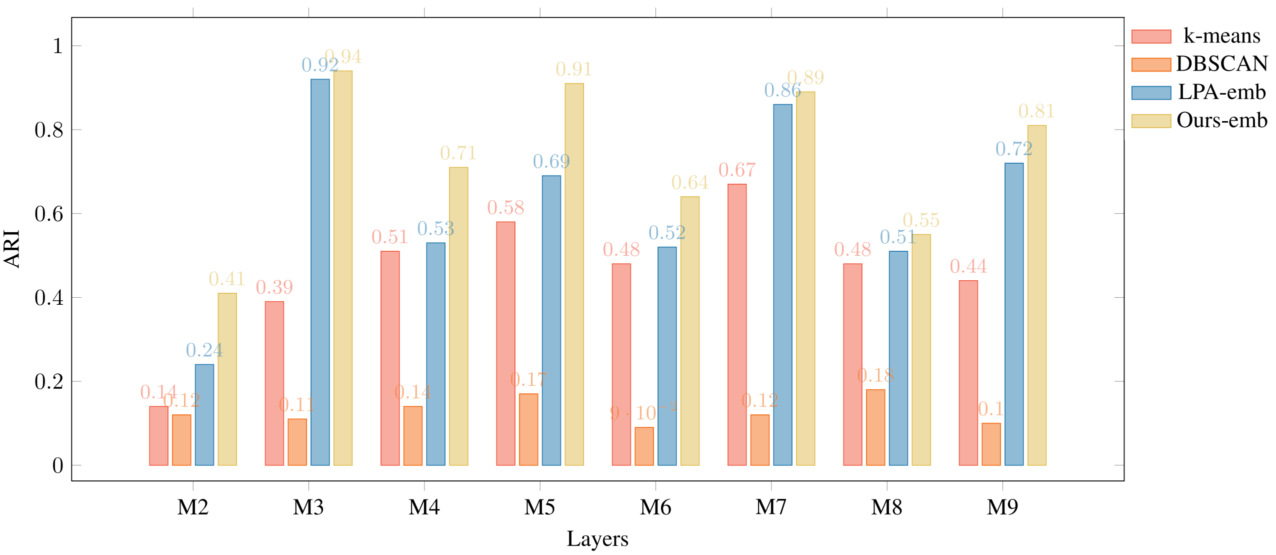


Fig. 17. ARI of conducting layout pattern matching using raw layout snippets (*k*-means and DBSCAN), embeddings presented in `LPA-DCE`(LPA-emb) [23], and embeddings presented in this work(Ours-emb).

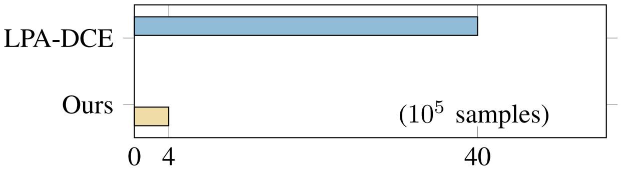


Fig. 18. Training sample requirements for training encoder with contrastive learning proposed in `LPA-DCE` versus the clustering loss used in this work.

ambiguity issue and improve the resolution of root cause identification. Embedded canonical forms are presented as latent codes of layout patterns, which decreases the size of features and the inference times. According to the principle of SCM, a deep ACE estimation method is proposed to model the causal relationship between candidate layout patterns and the systematic defect. The experimental results on several industrial designs show that our framework outperforms a commercial tool in different scenarios. We hope our research provides a new perspective on LPA for yield improvement.

References

[1] H. Yang, J. Su, Y. Zou, Y. Ma, B. Yu, and E. F. Y. Young, "Layout hotspot detection with feature tensor generation and deep biased learning," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38, no. 6, pp. 1175–1187, Jun. 2019.

[2] R. Chen *et al.*, "Faster region-based hotspot detection," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 3, pp. 669–680, Mar. 2022.

[3] W. Ye, Y. Lin, M. Li, Q. Liu, and D. Z. Pan, "LithoROC: Lithography hotspot detection with explicit ROC optimization," in *Proc. IEEE/ACM Asia South Pac. Des. Autom. Conf. (ASPDAC)*, 2019, pp. 292–298.

[4] H. Geng *et al.*, "Hotspot detection via attention-based deep layout metric learning," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des. (ICCAD)*, 2020, pp. 1–8.

[5] B. Benware, C. Schuermyer, M. Sharma, and T. Herrmann, "Determining a failure root cause distribution from a population of layout-aware scan diagnosis results," *IEEE Design Test Comput.*, vol. 29, no. 1, pp. 8–18, Feb. 2012.

[6] H. Tang, S. Manish, J. Rajski, M. Keim, and B. Benware, "Analyzing volume diagnosis results with statistical learning for yield improvement," in *Proc. 12th IEEE Eur. Test Symp. (ETS)*, 2007, pp. 145–150.

[7] W.-T. Cheng, Y. Tian, and S. M. Reddy, "Volume diagnosis data mining," in *Proc. 22nd IEEE Eur. Test Symp. (ETS)*, 2017, pp. 1–10.

[8] W.-T. Cheng *et al.*, "Automatic identification of yield limiting layout patterns using root cause deconvolution on volume scan diagnosis data," in *Proc. IEEE 26th Asian Test Symp. (ATS)*, 2017, pp. 219–224.

[9] W. C. J. Tam and R. D. S. Blanton, "LASIC: Layout analysis for systematic IC-defect identification using clustering," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 8, pp. 1278–1290, Aug. 2015.

[10] W. C. Tam, O. Poku, and R. D. Blanton, "Systematic defect identification through layout snippet clustering," in *Proc. IEEE Int. Test Conf.*, 2010, pp. 1–10.

[11] I. Misra and L. V. D. Maaten, "Self-supervised learning of pretext-invariant representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 6707–6717.

[12] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2015, pp. 1422–1430.

[13] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 9726–9735.

[14] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2020, pp. 1597–1607.

[15] X. Chen and K. He, "Exploring simple siamese representation learning," 2020, *arXiv:2011.10566*.

[16] C. Feichtenhofer, H. Fan, B. Xiong, R. Girshick, and K. He, "A large-scale study on unsupervised spatiotemporal representation learning," 2021, *arXiv:2104.14558*.

[17] J. Pearl, *Causality*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[18] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3319–3328.

[19] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3145–3153.

[20] A. Chattopadhyay, P. Manupriya, A. Sarkar, and V. N. Balasubramanian, "Neural network attributions: A causal perspective," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 981–990.

[21] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

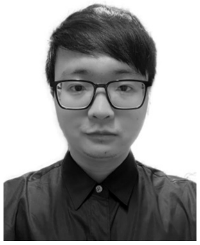
[22] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks," in *Proc. BMVC*, vol. 1, 2016, p. 3.

[23] R. Chen, "A unified framework for layout pattern analysis with deep causal estimation," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, 2021, pp. 1–9.

[24] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, Nov. 1987.

[25] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, 2019, pp. 8024–8035.

[26] D. Steinley, "Properties of the Hubert-arable adjusted rand index," *Psychol. Methods*, vol. 9, no. 3, p. 386, 2004.



Ran Chen received the B.E. degree from the Department of Information Engineering, Chinese University of Hong Kong, Hong Kong, in 2018, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering.

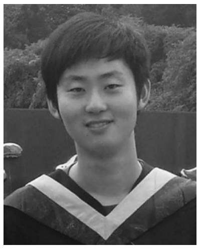
His research interests include machine learning application in VLSI design for manufacturing, yield learning, and pattern recognition.

Shoubo Hu received the Ph.D. degree in computer science and engineering from The Chinese University of Hong Kong, Hong Kong, in 2020.

He is currently a Researcher with Noah's Ark Lab, Huawei Technologies, Hong Kong. His current research interests include machine learning, causality, and their application in electronic design automation.

Zhitang Chen received the bachelor's degree from the Department of Automation, Sun Yat-sen University, Guangzhou, China, in 2010, and the Ph.D. degree from the Department of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong, in 2014.

He is a Researcher with the Noah's Ark Lab, Huawei Technologies, Hong Kong. His current research interests include kernel methods, deep learning, reinforcement learning, and multiagent systems and their applications to computer networks.



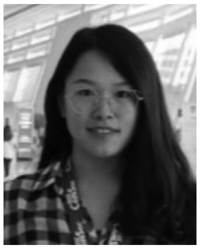
Shengyu Zhu received the B.E. degree in electrical engineering from the Beijing Institute of Technology, Beijing, China, in 2010, and the M.S. degree in mathematics and the Ph.D. degree in electrical and computer engineering from Syracuse University, Syracuse, NY, USA, in 2016 and 2017, respectively.

He is currently a Principal Researcher with Noah's Ark Lab, Huawei, Hong Kong, China. His research interests include causality, machine learning, information theory, and statistical signal processing.

Bei Yu (Member, IEEE) received the Ph.D. degree from The University of Texas at Austin, Austin, TX, USA, in 2014.

He is currently an Associate Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

Dr. Yu received nine Best Paper Awards from DATE 2022, ICCAD 2021 and 2013, ASPDAC 2021 and 2012, ICTAI 2019, *Integration, the VLSI Journal* in 2018, ISPD 2017, and SPIE Advanced Lithography Conference 2016, and six ICCAD/ISPD contest awards. He has served as the TPC Chair of ACM/IEEE Workshop on Machine Learning for CAD, and in many journal editorial boards and conference committees. He is an Editor of IEEE TCCPS Newsletter.



Pengyun Li received the M.S. degree from Xidian University, Xi'an, China, in 2020.

She is currently working as a Senior Algorithm Engineer with Huawei, Hong Kong, reaching topics about chip yield learning.

Cheng Chen received the M.S. degree from Xidian University, Xi'an, China, in 2013.

He is a Senior Software Engineer with Huawei, Hong Kong. His research interests include physical design, yield analysis, and machine learning.

Yu Huang (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Iowa, Iowa City, IA, USA.

He is a Semiconductor Scientist of Huawei, Hong Kong, and an EDA Chief Architect and an EDA Lab Director of HiSilicon, Shenzhen, China. Before joining HiSilicon, he was a Senior Key Expert of Mentor Graphics, Wilsonville, OR, USA. He is also an Adjunct Professor with the School of Microelectronics, Fudan University, Shanghai, China. He has more than 70 patents and published more than 140 papers on leading IEEE journals, conferences, and workshops. His research interests include VLSI SoC testing, ATPG, compression, diagnosis, yield analysis, machine learning, and AI chips.

Dr. Huang has served as a Technical Program Committee Member for DAC, ITC, VTS, ATS, ETS, ASPDAC, NATW, and many other conferences and workshops in the testing area.

Jianye Hao received the Ph.D. degree in computer science and engineering from The Chinese University of Hong Kong, Hong Kong, in 2013.

He is currently an Associate Professor with the School of Computing and Intelligence, Tianjin University, Tianjin, China. His research interests focus on fundamental problems of multiagent reinforcement learning and its key applications, such as game AI, network optimization, E-commerce optimization, and autonomous driving. He has published over 80 research papers in prestigious conferences and journals and two books.

Prof. Hao's team also has received a number of awards, including the Distinguished Paper Award of ASE 2019, the Best System Paper Award of CoRL 2020 and Champion of BBO and MineRL challenge hosted at NeurIPS 2020, and Champion Team of BBO for LP/MIP Solvers Competition at NeurIPS 2021. He was a (Senior) PC Member of a number of prestigious conferences, such as IJCAI, AAAI, ICML, NeurIPS, and ICLR and a Reviewer for journals, such as IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, and *Journal of Autonomous Agents and Multi-Agent Systems*.