A Unified Framework for Layout Pattern Analysis with Deep Causal Estimation

Ran Chen[1†], Shoubo Hu[2], Zhitang Chen[2], Shengyu Zhu[2], Bei Yu[1]
Pengyun Li[3] Cheng Chen[3] Yu Huang[3] Jianye Hao[2]
[1]The Chinese University of Hong Kong [2]Huawei Noah's Ark Lab [3]HiSilicon

***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 paper, 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 $\times$8.4 speedup on average.**

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 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, 2, 3] 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 [4], which characterizes the conditional distribution of systematic defect given potential root causes. It

[†]This work was done when Ran Chen was intern at Huawei Noah's Ark Lab.

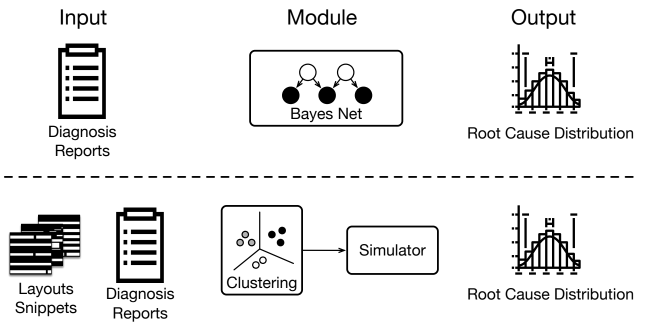


Fig. 1 Overview of prior methods. Upper: [4, 6]; lower: [8, 9].

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 [5, 6] 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.* [7] proposed an advanced solution based on [4]. 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 [7].

Another class of seminal works [8, 9] 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 [9] 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 and improve the resolution of root cause identification, a unified framework for layout pattern analysis 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. Lastly, 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 average causal effect estimation to identify the root cause. The main contributions of this work are threefold:

- 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. Our framework can identify the critical root causes and provide high-resolution clustered snippets for further analysis.
- 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 average causal effect for root cause identification.
- 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 in different scenarios and we get $\times 8.4$ speedup on average at inference.

The remainder of this paper 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.

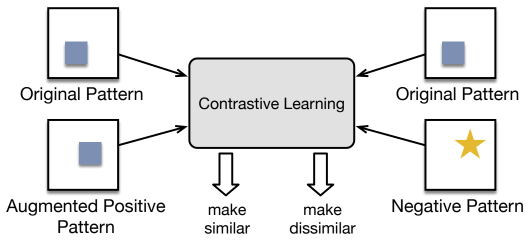


Fig. 2 Idea of contrastive learning: maximize the similarity between latent features of an image and its augmented version and simultaneously minimize the similarity between latent features of inputs correspond to different original images.

A. Layout Pattern Analysis

Layout Pattern Analysis (LPA) takes a step towards identifying the layout patterns which cause systematic defects. Cheng *et al.* [7] proposed an LPA solution which is an enhanced flow based on [4]. 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 [4], root cause analysis results including root cause distribution and layout patterns are returned for further study.

B. Contrastive Learning

Conventional deep networks 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 [10, 11]. 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 keep the discriminative power against different instances through a contrastive loss in the latent space, as illustrated in Fig. 2. Recent contrastive learning methods [12, 13, 14, 15] have achieved competitive results in visual representation learning compared with prominent supervised learning methods for computer vision tasks.

C. Structural Causal Models

Structural Causal Models (SCMs) [16] are developed towards a comprehensive theory of causation and serve as a key ingredient of our framework.

Definition 1 (Structural Causal Model [16])**.** *A structural causal model M is a 4-tuple $(E, X, F, P(E))$, where*

- *E is a set of exogenous (unobserved) variables;*

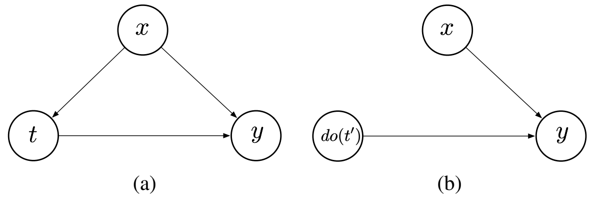


Fig. 3 An SCM (a) without and (b) under intervention. Nodes represent random variables and directed edges $x \rightarrow y$ indicates that x is a direct cause of y. Under intervention $do(t')$, the intervened variable t is fixed to the intervened value t' and all its incoming edges are removed.

- *X is a set of endogenous (observed) variables;*
- *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;*
- *The uncertainty is encoded through a probability distribution over the exogenous variables, $P(E)$.*

SCMs provide a compact way of characterizing average causal effect $ACE^y_{do(x_i)}$, 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* [16], 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 $x_i = \alpha$.

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 [17, 18]. Recently, [19] approached neural network attribution problems from a causal perspective. They view a multilayer perceptron (MLP) $\{l_{in}, \ldots, l_{out}\}$ as an SCM $M'(E, \{l_{in}, l_{out}\}, f', P(E))$, where l_{in} is the input layer, l_{out} is the output layer, E refers to a set of exogenous random variables which act as causal factors for the input neurons l_{in}, f' refers to the mapping from the input to output by marginalizing out all hidden neurons.

Based on SCM reformulation, [19] approximated the interventional expectation of the output neurons $f'(l_{in})$ under the intervention $do(x_i = \alpha)$ as:

$$\begin{aligned}&\mathbb{E}[f'(l_{in})|do(x_i = \alpha)] \approx f'(\mu) + \\ &\frac{1}{2}\text{tr}(\nabla^2 f'(\mu)\mathbb{E}[(l_{in} - \mu)(l_{in} - \mu)^T|do(x_i = \alpha)]),\end{aligned} \tag{1}$$

where $\text{tr}(\cdot)$ is the trace operator, $\mu = [\mu_1, \ldots, \mu_k]^T$ and each entry $\mu_{i'} = \mathbb{E}[x_{i'}|do(x_i = \alpha)]$, $\forall x_{i'} \in l_{in}$, 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 true root cause(s) of 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 causal relationship between potential root causes and systematic defect (Section III-C).

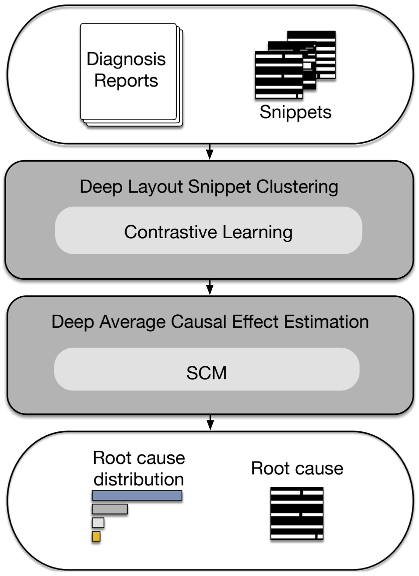


Fig. 4 Overview of the framework.

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 feed-forward 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 Average Causal Effects (ACEs) of layout patterns to systematic defect are evaluated to identify the true root cause(s).

B. Deep Layout Snippet Clustering (DLSC)

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 prototypes of layout snippets. The network structure of our model is shown in Fig. 5. "SeparableConv" indicates depthwise separable convolution layer which is a variant convolution layer widely used in [20] 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 two-dimensional features to embeddings.

Contrastive Loss. 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 non-equivalent as dissimilar as possible. To achieve this, ℓ_p-norm is used as a metric to measure the dissimilarity between embeddings $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ of a pair of layout snippets:

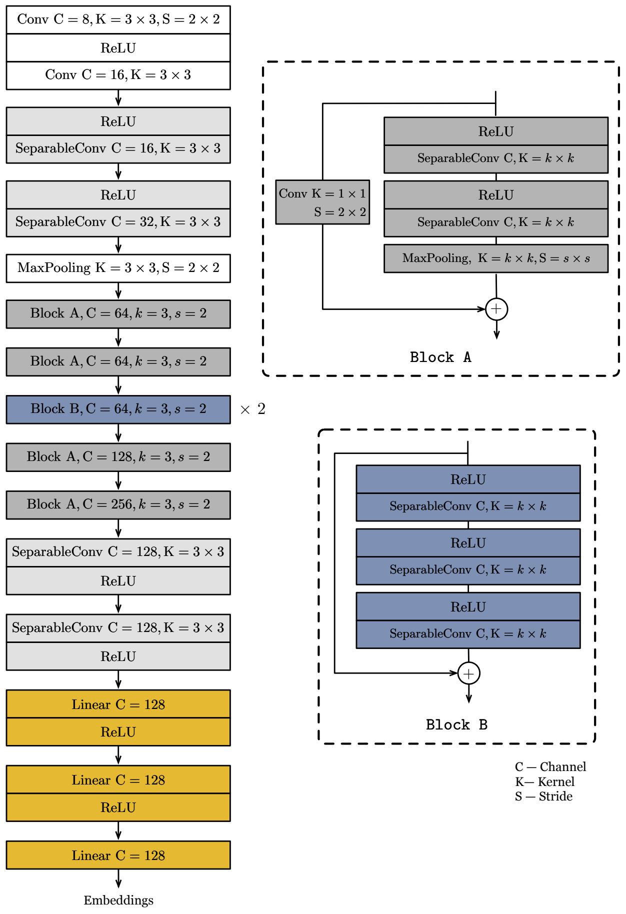
$$d(\boldsymbol{z}_i, \boldsymbol{z}_j) = \left\| \boldsymbol{z}_i - \boldsymbol{z}_j \right\|_p , \qquad (2)$$

where p is a real number greater than 1 and is set to $p = 2$ in this work. Contrastive loss [21] based on the metric above is given by

$$L(\boldsymbol{z}, \boldsymbol{p}, \boldsymbol{n}) = \max(d(\boldsymbol{z}, \boldsymbol{p}) - d(\boldsymbol{z}, \boldsymbol{n}) + marg, 0), \qquad (3)$$

where $marg$ is a non-negative value indicating an appropriately set margin, $\boldsymbol{z}$, $\boldsymbol{p}$, and $\boldsymbol{n}$ are the embedding of one layout snippet, the embedding of a positive sample of $\boldsymbol{z}$, and the embedding of a negative sample of $\boldsymbol{z}$, 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 $\boldsymbol{z}$ and negative samples $\boldsymbol{n}$ are penalized to be far from anchor embedding. An illustration on contrastive learning with an encoder is shown in Fig. 6. 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 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



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.

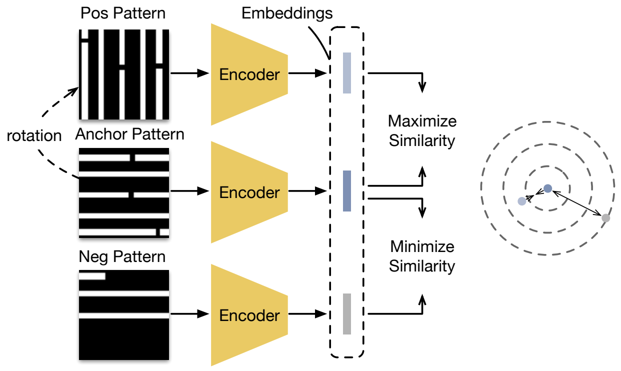


Fig. 6 Illustration of encoder network training using contrastive learning. Parameters of encoders are shared.

pattern. The silhouette method [22] 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.

An example of the DLSC is illustrated in Fig. 7, layout

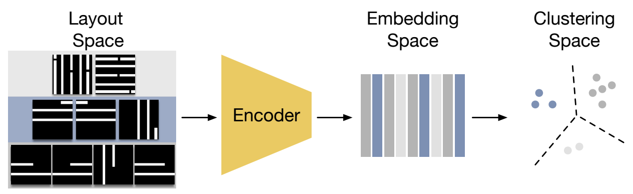


Fig. 7 An example of Deep Layout Snippet Clustering.

snippets with a large number of pixels are transformed to low-dimensional embeddings which reduces the clustering computation remarkably while improves 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 (DACE)

In this section, we introduce how we use average causal effect estimation to identify 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 j-th embedding to the center of cluster i. Then the distance matrix are 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

$$[\mathbf{P}]_{j,i} = \frac{\exp\left(-\mathbf{D}_{j,i}/\tau\right)}{\sum_{i'} \exp\left(-\mathbf{D}_{j,i'}/\tau\right)}, \tag{4}$$

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. The input layer l_{in} of $\mathcal{M}$ has k input neurons x_i, $i \in \{1,\dots,k\}$, each of which corresponds to one layout pattern. Its output layer l_{out} has one output neuron indicating the probability of systematic defect. The objective function for training the $\mathcal{M}$ is

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{e=1}^{m} \log\left[\sum_i p(r^e|y_i)p(y_i|\boldsymbol{\mu}_i,\boldsymbol{\theta})\right], \tag{5}$$

where m is the number of diagnosis reports, $\boldsymbol{\theta}$ denotes the parameters of $\mathcal{M}$, $\boldsymbol{\mu}_i = \frac{1}{n_i}\sum_{j\in C_i}^{n} \mathbf{P}_{j,:}$ 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 Equation (5). The detail on estimating $p(r^e|y_i)$ is elaborated in Section IV.

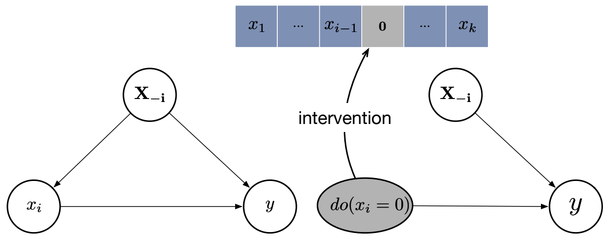


Fig. 8 Left: The defect SCM for Layout Pattern Analysis without intervention. Right: Apply intervention on cluster i.

LPA by ACE estimation. After the objective in Equation (5) 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 average causal effect on the systematic defect. Therefore, average causal effects 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 Equation (1) is adopted to compute the ACE of each layout patterns on the systematic defect.

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 j-th 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

$$ACE^y_{do(x_i)} = \left|\mathbb{E}[y|do(x_i=0)] - \mathbb{E}[y|do(x_i=1)]\right|. \tag{6}$$

The interventional expectation when x_i is intervened to 0 in Equation (6) can be estimated using Equation (1) as

$$\begin{aligned}&\mathbb{E}[y|do(x_i=0)] \approx f'(\boldsymbol{\mu}_{i0}) + \\ &\frac{1}{2}\mathrm{tr}(\nabla^2 f'(\boldsymbol{\mu}_{i0})\mathbb{E}[(l_{in}-\boldsymbol{\mu}_{i0})(l_{in}-\boldsymbol{\mu}_{i0})^T|do(x_i=0)]),\end{aligned} \tag{7}$$

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 i-th 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

$$p(x_i) = \frac{ACE^y_{do(x_i)}}{\sum_{i'} ACE^y_{do(x'_i)}}. \tag{8}$$

D. Inference flow

An overview of this unified framework and modules of DLSC and DACE are introduced in Section III-B and Section III-C. Here we give a detailed explanation on the inference flow of the framework.

Algorithm 1 The 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($\mathbf{s}_j$) $\to \boldsymbol{z}_j$, $\forall j \in d$;        ▷ Equation (3)
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 $\mathbf{D}_d$, $\forall d \in \{1, \ldots, L\}$;
9: Convert $\mathbf{D}$ to $\mathbf{P}$;        ▷ Equation (4)
10: Train defect SCM $\mathcal{M}(\boldsymbol{\theta})$;        ▷ Equation (5)
11: for $i = 1 \to k$ do
12:     Calculate ACE of clusters;        ▷ Equation (6)
13: end for
14: return Root cause distribution;        ▷ Equation (8)
```

TABLE I 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 II 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

Pseudocode of the inference flow is presented in Algorithm 1, lines 1-7 correspond to the inference steps of DLSC and lines 8-12 represent the procedure of DACE. Firstly, 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. Secondly, the membership matrix of layout snippets in each layer are computed and different matrices from all layout layers are concatenated along axis zero to form $\mathbf{P}$. An SCM $\mathcal{M}$ is then learned using the information in diagnosis reports. Lastly, the average causal effect of each cluster on the systematic defect is estimated according to Equation (6). The root causes are identified based on the estimated ACE of all candidate layout patterns.

TABLE III Defect Injection Statistics.

	`#TotalInjections`	`#Open`	`#Bridge`
Case 1	68	50	18
Case 2	107	72	35
Case 3	93	69	24
Case 2 noise	856	576	280

IV. Experimental results

We evaluate the effectiveness of our proposed framework by testing its root cause identification accuracy on three noise-free datasets from three different layout designs, and eight noisy datasets from one layout design. 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. 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 [4] to generate diagnosis reports. Defects of type *Open* and *Bridge* are considered in the injection steps. The detailed information within diagnosis reports are listed in TABLE I. Three classes of datasets are considered in our evaluation: (1) Noise-free dataset. Three different layout designs are used to construct noise-free datasets and their basic information is shown in TABLE II. Case 2 and case 3 are two real silicon datasets that result from real in-production IC. Diagnosis reports in one noise-free dataset share one single true root cause of 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 III). (2) Noisy dataset. A certain percentage of diagnosis reports in the dataset share 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 share one root cause independently. Three portions of root causes are true and the rest portion is different from the true one. E.g., proportion '50-20-20-10' in TABLE VI 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

TABLE IV Accuracy(%) on Noise-free Datasets.

Dataset	Baseline	Commercial Tool	Ours
Case 1	25.00	98.53	**100.00**
Case 2	55.88	92.52	**98.04**
Case 3	58.06	98.92	98.92
Average	46.31	96.66	**98.99**

TABLE V Accuracy(%) on Noisy Case 2 Datasets.

Noise(%)	Baseline	Commercial Tool	Ours
80	19.57	84.11	**97.83**
70	37.62	92.52	**95.05**
60	44.55	94.39	**98.02**
50	49.02	94.39	**96.08**
40	50.98	93.45	**95.10**
30	51.96	**93.45**	93.14
20	58.82	92.52	**95.10**
10	55.88	93.46	**98.04**
Average	46.05	92.29	**96.05**

of injection and types of defect are the same across different noise levels. We merged them as ‘Case 2 noise’ in TABLE III.

B. Implementation Details

The proposed framework is implemented in Python with PyTorch library [23]. The encoder network is trained using four Nvidia Tesla V100 GPUs. SGD optimizer is adopted with initial learning rate $1\mathrm{e}{-1}$, weight decay $5\mathrm{e}{-4}$, and momentum 0.9. The batch size, number of epochs, margin $marg$ in Equation (3) are set to 64, 16, and 1.5 in the experiments, respectively. Following [14], we add a single linear layer before the output of encoder during training to avoid the feature collapsion problem.

When conducting LPA, the defect SCM $\mathcal{M}$ is trained using SGD optimizer with initial learning rate $1\mathrm{e}{-2}$, weight decay $1\mathrm{e}{-3}$, and momentum 0.9. The maximum number of epoch 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

$$p(r^e|y_i) = \frac{h^e_{j*} s^e_{j*}}{v_{j*}} \mathbb{1}_{\{s|s\geq 90\}}(s^e_{j*}), \tag{9}$$

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 with 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 volumn 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 mean ratio within each cluster. (3) The cluster with the top rank of ratio is regarded as the root cause predicted by the diagnosis reports.

TABLE VI Accuracy(%) on Mixture Datasets.

Proportion (r1%-r2%-r3%-noise%)	Commercial Tool		Ours	
	Case 2	Case 3	Case 2	Case 3
30-30-30-10	85	70	**87**	**81**
40-20-20-20	66	24	**73**	**74**
40-30-20-10	**81**	70	79	**75**
40-30-30-00	**88**	**82**	83	78
50-20-20-10	**77**	**58**	76	58
50-30-20-00	84	**82**	84	79
60-20-20-00	75	**71**	**79**	50
20-20-20-40	63	8	**81**	**49**
30-20-20-30	63	18	**83**	**58**
30-30-20-20	78	36	**84**	**75**
Average	76	52	**81**	**68**

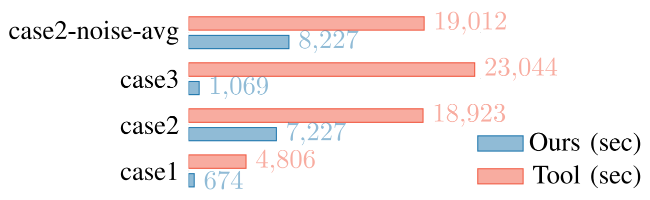


Fig. 9 Inference speed comparison between our framework and the commercial tool.

On noise-free datasets. As shown in TABLE IV, our method outperforms the commercial tool 2.33% on average under the setting of noise-free defect injection. The average accuracy of the baseline is 46.31% which is much lower than our method and the commerical tool. This indicates that by using simple statistics according to the diagnosis report we can never locate the root cause accurately.

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 V. The average accuracy of the framework is 3.76% higher than the commercial tool. Especially when the ratio of noise greater than 70%, it is difficult for the commercial tool to identify the root cause precisely. While our framework locates the root cause with 13.72% higher accuracy then the commercial tool. The proposed method is robust to the injection noise.

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 VI show that the proposed framework has competitive results compared to the commercial tool especially in tasks of identifying 1 or 2 true root causes from mixture dataset, while the commercial tool may provide a misleading results which can not be used for further study. And we got 10.5% better accuracy on average than the

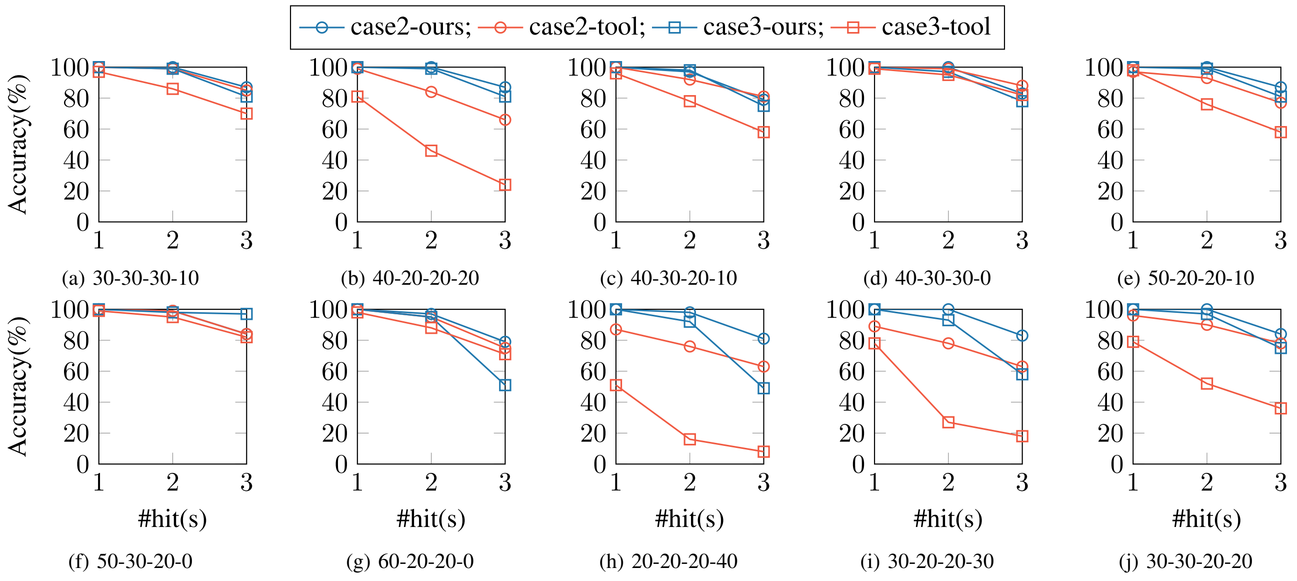


Fig. 10 Accuracy of identifying 1, 2, and 3 true root causes in top-3 layout patterns on mixture datasets.

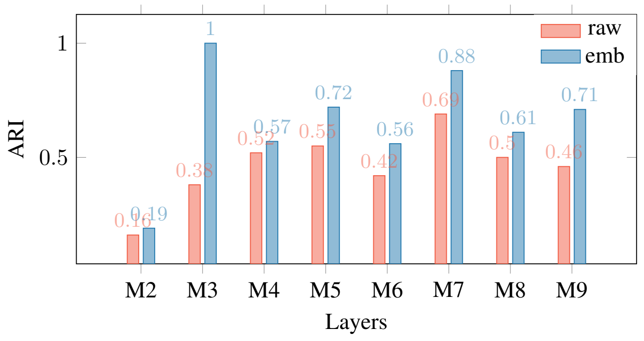


Fig. 11 ARI of conducting layout pattern matching using raw layout snippets (raw) and embeddings (emb).

commercial tool.

Top-3 accuracy on mixture datasets. We analyze the performance of identifying 1, 2, and 3 true root causes from the mixture datasets. Since there are 3 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, 2, and 3 true root causes are shown in Fig. 10, our method can identify at least 1 root cause in all cases while the commercial tool fails to achieve 100% accuracy. Also, the average accuracy of identifying 2 root causes of our method is greater than 95%.

Inference speed. The inference time of our framework and the commercial tool on single root cause datasets are shown in Fig. 9. The speed of the proposed deep learning framework method surpasses the conventional commercial tool by a large margin. We got around $\times 10.4$ and $\times 2.3$ speedup on noise-free datasets and noisy datasets. The robustness of accuracy and inference speed indicate our method is valuable for industry.

Clustering quality. We compare the clustering quality of DLSC with directly applying k-means algorithm on raw layout snippets using the adjusted rand index (ARI $\in [-1, 1]$, [24]). 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 good match between the clustering results and the ground truth. The experiments are conducted on case 2, 256 layout snippets are sampled on each layer for evaluation. The ARIs of clustering using raw layout snippets and embeddings in DLSC presented in Fig. 11 are the mean of ten independent evaluations. The DLSC got higher ARI scores across all metal layers, which indicates our contrastive learning method can improve the quality of layout pattern matching.

V. Discussion and Conclusion

In this paper, a unified framework on layout pattern analysis is proposed to identify the root cause(s). Based on the concept of contrastive learning, a deep layout snippet clustering module is designed to solve the 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 structural causal model, a deep average causal effect 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 layout pattern analysis for yield improvement.

Acknowledgment

This work is partially supported by The Research Grants Council of Hong Kong SAR CUHK14209420, CUHK14208021.

References

[1] R. Chen, W. Zhong, H. Yang, H. Geng, F. Yang, X. Zeng, and B. Yu, "Faster region-based hotspot detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020.

[2] 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 Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 38, no. 6, pp. 1175–1187, 2019.

[3] W. Ye, Y. Lin, M. Li, Q. Liu, and D. Z. Pan, "LithoROC: lithography hotspot detection with explicit ROC optimization," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2019, pp. 292–298.

[4] 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 of Computers*, vol. 29, no. 1, pp. 8–18, 2012.

[5] H. Tang, S. Manish, J. Rajski, M. Keim, and B. Benware, "Analyzing volume diagnosis results with statistical learning for yield improvement," in *12th IEEE European Test Symposium (ETS'07)*. IEEE, 2007, pp. 145–150.

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

[7] W.-T. Cheng, R. Klingenberg, B. Benware, W. Yang, M. Sharma, G. Eide, Y. Tian, S. M. Reddy, Y. Pan, S. Fernandes *et al.*, "Automatic identification of yield limiting layout patterns using root cause deconvolution on volume scan diagnosis data," in *2017 IEEE 26th Asian Test Symposium (ATS)*. IEEE, 2017, pp. 219–224.

[8] W. C. J. Tam and R. D. S. Blanton, "Lasic: Layout analysis for systematic ic-defect identification using clustering," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 8, pp. 1278–1290, 2015.

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

[10] I. Misra and L. v. d. Maaten, "Self-supervised learning of pretext-invariant representations," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6707–6717.

[11] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1422–1430.

[12] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[13] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning (ICML)*. PMLR, 2020, pp. 1597–1607.

[14] X. Chen and K. He, "Exploring simple siamese representation learning," 2020.

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

[16] J. Pearl, *Causality*. Cambridge university press, 2009.

[17] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3319–3328.

[18] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3145–3153.

[19] A. Chattopadhyay, P. Manupriya, A. Sarkar, and V. N. Balasubramanian, "Neural network attributions: A causal perspective," in *International Conference on Machine Learning*. PMLR, 2019, pp. 981–990.

[20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[21] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks." in *BMVC*, vol. 1, no. 2, 2016, p. 3.

[22] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[23] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Annual Conference on Neural Information Processing Systems (NIPS)*, 2019, pp. 8024–8035.

[24] D. Steinley, "Properties of the hubert-arable adjusted rand index." *Psychological methods*, vol. 9, no. 3, p. 386, 2004.