PRICING: Privacy-Preserving Circuit Data Sharing Framework for Lithographic Hotspot Detection

Chen-Chia Chang[1], Wan-Hsuan Lin[2], Jingyu Pan[1], Guanglei Zhou[1], Zhiyao Xie[3], Jiang Hu[4], and Yiran Chen[1]

Duke University[1] University of California, Los Angeles[2]
Hong Kong University of Science and Technology[3] Texas A&M University[4]
{chenchia.chang, jingyu.pan, yiran.chen}@duke.edu, wanhsuanlin@ucla.edu, eezhiyao@ust.hk, jianghu@tamu.edu

ABSTRACT

To apply machine learning (ML) techniques for electronic design automation (EDA), training models on diverse datasets is essential for model reliability and generalizability, especially when applied to modern circuits. However, data availability remains a severe issue as circuit data is typically kept confidential within each data provider due to the difficulty of secure data sharing. This problem has impeded the development of ML for EDA in both industry and academia and has never been well addressed. To facilitate model development, enabling secure data sharing among various data providers is needed. To this end, we propose PRICING, a privacy-preserving circuit data sharing framework. This is the first exploration to (1) investigate the secure data sharing problem in EDA and (2) generate protected circuit features that hide important circuit information while preserving sufficient information for a well-known EDA application, lithographic hotspot detection. Our results demonstrate that our approach successfully protects raw circuit features, providing 55% superior protection over existing state-of-the-art techniques in computer vision. Moreover, models trained with our protected data achieve up to 48% higher accuracy than models trained with limited raw data. This shows the effectiveness of PRICING in enhancing model development for EDA.

ACM Reference Format:
Chen-Chia Chang[1], Wan-Hsuan Lin[2], Jingyu Pan[1], Guanglei Zhou[1],, Zhiyao Xie[3], Jiang Hu[4], and Yiran Chen[1]. 2025. PRICING: Privacy-Preserving Circuit Data Sharing Framework for Lithographic Hotspot Detection . In *30th Asia and South Pacific Design Automation Conference (ASPDAC '25), January 20–23, 2025, Tokyo, Japan.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3658617.3697773

1 INTRODUCTION

ML-aided EDA approaches have shown its potential in enhancing cross-stage interactions by performing quality prediction and optimization [6], e.g., lithographic hotspot detection [22, 24] and routability prediction [2, 9, 20]. The success of ML-based strategies relies heavily on access to diverse and representative circuit data for both model development and evaluation. However, data availability [1, 12] is a long-lasting issue in EDA due to the difficulty of secure data sharing not only between academia and industry but also between different design teams within a single company [14]. Sensitive design information needs to be protected to maintain competitive advantages and safeguard intellectual property. This data scarcity leaves developers with limited and often outdated circuit designs, hindering them from developing models that are generalizable and reliable to address cutting-edge design challenges. In addition, it results in the absence of widely-recognized and high-quality datasets, making fair comparisons among different solutions challenging.

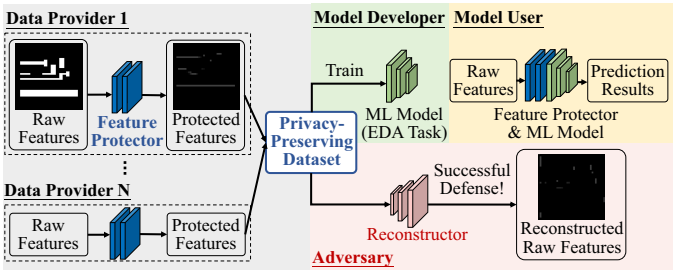

Figure 1: Our framework PRICING. Given a feature protector, data providers use it to locally transform their raw features into protected ones and share them to a privacy-preserving dataset. Model developers use the dataset to train ML models, and model users apply both the feature protector and the model to obtain prediction results. Our feature protector hides the sensitive information and prevents attacks of recovering raw features from adversaries.

Despite widespread acknowledgment of these challenges within the EDA community, efforts [1, 12] to mitigate the data availability issue have been inadequate. A work [1] publishes an open-sourced dataset but without industrial circuit data. The other work [12] builds a federated learning framework to collaboratively train ML models with data from multiple clients. However, it only supports the training of shallow neural networks and does not enable secure data sharing to facilitate model evaluation. Thus, enabling privacy-preserving data sharing is an urgent need to enhance both ML model development and evaluation for EDA.

Our framework PRICING, as shown in Figure 1, offers a circuit data sharing mechanism from various data providers. It ensures privacy protection against third-party malicious attackers and prevents data users from accessing actual design details. Thus, data providers can share their data without privacy leakage concern. The key of this framework is to have a feature protector that transforms raw features into protected ones to ensure circuit data privacy. An ideal feature protector should accomplish three goals. First, protected features should not reveal any sensitive information of the original circuit. Second, protected features should be able to defend against *reconstruction attacks* [8, 17], whose goal is to reconstruct the raw features from the protected ones. Third, protected features need to contain sufficient information to train ML models for the target prediction task. With the feature protector, design owners can generate protected features and share them to construct a *privacy-preserving dataset*. Then, developers can train models based on a wider range of circuits to improve model generalization and reliability. This privacy-preserving dataset can also faciliate fair comparisons in EDA research. The proposed framework offers a promising solution to data availability to accelerate the advancement of ML for EDA.

ASPDAC '25, January 20–23, 2025, Tokyo, Japan

ACM ISBN 979-8-4007-0635-6/25/01.
https://doi.org/10.1145/3658617.3697773

To the best of our knowledge, we are the first to explore the privacy-preserving circuit data sharing problem in EDA. To demonstrate the concept, we focus on a well-established research topic, lithographic hotspot detection. In this context, protecting the detailed polygon patterns in layout clips (raw features) is crucial for privacy, as these could reveal sensitive technology and design information. The challenge of protection lies in the fact that layout clips are single-channel binary images, making them more difficult to obfuscate compared to three-channel RGB images commonly used in computer vision. Thus, we develop a novel feature protector to make sensitive information unrecognizable in the protected and reconstructed features. Recognizing that obfuscating raw features may lose essential information for the prediction tasks, we incorporate an adversarial training method for the feature protector to intelligently strike a balance between protecting privacy and preserving utility for model training. Additionally, we train our feature protector using limited data and evaluate it on more extensive datasets, demonstrating the broad applicability of our framework PRICING.

The contributions of this paper are summarized as follows:

- We propose PRICING, the pioneering privacy-preserving circuit data sharing framework for EDA.
- We design a novel feature protector consisting of an obfuscating network and a mask-based pruning network to effectively obfuscate sensitive information and intelligently incorporate with adversarial training to retain abundant information for model training in the protected features.
- Compared to state-of-the-art obfuscating methods [8, 17] in computer vision, our method achieves up to 55% better feature protection on peak signal-to-noise ratio (PSNR). Through qualitative evaluation, PRICING stands as the only method that provides satisfying feature protection.
- The privacy-preserving dataset constructed by PRICING can largely improve the model performance and generalizability. Specifically, our models outperform those trained on limited raw features with up to 48% improvement on the area under the receiver operating characteristic curve (ROC-AUC). Our models also outperform those trained on all raw features, showing that PRICING further provides an effective regularization benefit.

The similarities between our target application and other essential ML for EDA problems, e.g., routability prediction [2, 9, 20], IR drop estimation [21], and clock tree prediction [11], suggest that our approach is not task-specific and could benefit the development of diverse EDA problems. In addition, PRICING will be open-sourced to encourage collaboration and advancement in both academia and industry.

2 PRELIMINARIES

2.1 Privacy-Preserving Data Sharing Scenario

In this scenario (Figure 1), there are several data providers, model developers, model users, and an adversary. Let $f(\cdot|\theta)$ denote a model f with weights θ. In the following, we detail our feature protector setup and formally define the privacy, the adversary, the utility for model training, and our problem formulation.

Feature protector. The goal of the feature protector $f_c(\cdot|\theta_c)$ is to transform raw features X into protected ones $Z = f_c(X|\theta_c)$. Each data provider applies f_c locally to generate protected features and contributes them to a collaborative privacy-preserving dataset $\mathbb{D}$. Then, model developers can use $\mathbb{D}$ to train their own models $f_e(\cdot|\theta_e)$. Model users use f_c and f_e to perform inference on their own data.

Privacy. We perform quantitative and qualitative evaluation for privacy protection. In general, obfuscating the raw features can be achieved by minimizing the similarity between raw features and protected and reconstructed features. Thus, for quantitative evaluation, the quality of the feature protection is evaluated by PSNR [5], which is widely used for image quality assessment [8, 17]. PSNR gives the ratio between the power of a signal and the mean square error (ℓ_2) between each pixel, evaluating the extent of distortion noise (similarity) between two images. A lower PSNR indicates better feature protection. For qualitative evaluation, we provide the visualization for human perceptual study to present whether these features reveal any layout patterns under the adversary's attack.

Adversary. The adversary's goal is to build a *reconstructor* $f_a(\cdot|\theta_a)$ to recover Z to $X' = f_a(Z|\theta_a)$ such that PSNR of X and X' is minimized. To rigorously test the effectiveness of our feature protection, we simulate a white-box attack by granting the attacker the access to the dataset consisting of raw features that we used to learn the feature protector. Therefore, the adversary can learn $f_a(\cdot|\theta_a)$ to infer the other raw features from D.

Utility for model training. In this context, *utility* refers to the effectiveness of protected features in training ML models to accurately perform lithographic hotspot detection. We quantify the utility by measuring the ROC-AUC of models trained with our privacy-preserving dataset $\mathbb{D}$. A higher ROC-AUC indicates better model performance, showing that protected features maintain essential predictive qualities.

Our problem formulation can be defined as follows:

Problem 1 (**Privacy-preserving data-sharing scenario**). *Given circuit features X from data providers, the feature protector $f_c(\cdot|\theta_c)$ aims to make the protected features $Z = f_c(X|\theta_c)$ obfuscated and let the corresponding reconstructed features $X' = f_a(Z|\theta_a)$ produced by any adversary reconstructor $f_a(\cdot|\theta_a)$ do not contain layout information of X, i.e., PSNR of X and X' is maximized. In addition, when using $\mathbb{D}$ to train the model $f_e(\cdot|\theta_e)$, the ROC-AUC of $f_e(\cdot|\theta_e)$ is maximized.*

2.2 Related Works for Data Privacy Protection

Given that our targeting applications depend on image-based features, we focus on the efforts for image privacy protection in computer vision. There are three types of approach for image privacy protection in computer vision. First, encryption-based methods [3, 23] are a direction to protect data privacy. However, they are only applicable for one- or two-layer models and are not suitable for deep neural networks in EDA due to heavy computational overhead. Most importantly, their encryption cannot be applied on model training stages, thereby not able to construct a privacy-preserving dataset.

Differential privacy (DP) [13, 18] is an another widely-adopted method with theoretical protection guarantee by injecting Laplace noise in raw features. However, DP-based protected features cannot defense against reconstruction attacks. Our experimental results in Section 4.2 show that DP fails to meet the privacy requirements for the EDA application.

In recent years, adversarial representation learning (ARL)-based methods [7, 8, 17] have emerged as a leading approach for balancing image privacy and the utility for model inference. Two state-of-the-art methods [8, 17] develop adversarial training and channel pruning methods to combat reconstruction attacks. However, these methods cannot provide satisfying feature protection for lithographic hotspot detection as the feature protection results shown in Section 4.2. The binary and the single-channel raw features make the protection harder. Therefore, this work introduces a novel mask-based pruning network that integrates with ARL to offer enhanced flexibility. This approach allows for more precise retention of information crucial for training within protected features, while ensuring that sensitive information is discarded.

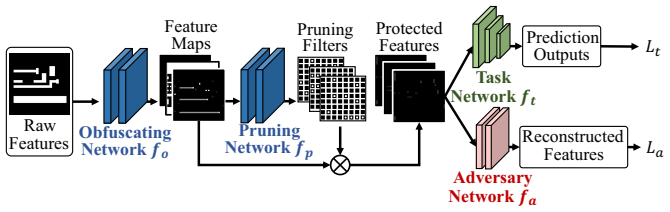

Figure 2: Overview of our feature protector and the training method. We include a task network to assist retaining information for prediction by minimizing the prediction loss. In addition, an adversary network is introduced to strengthen the reconstruction protection by maximizing the reconstruction loss.

3 FEATURE PROTECTOR CONSTRUCTION

The overview of our feature protector construction is sketched in Fig 2. To provide sufficient feature obfuscation, we propose an obfuscating network and a mask-based pruning network in our feature protector. To assist the training of the feature protector, we utilize an adversarial training algorithm along with a task network and an adversary network.

3.1 Feature Protector

Our feature protector consists of an *obfuscating network* and a *mask-based pruning network*.

Obfuscating network. The obfuscating network, represented as $f_o(\cdot|\theta_o)$, serves as an encoder designed to transform raw features X into obfuscated intermediate outputs $\hat{Z} = f_o(X|\theta_o)$. This transformation involves non-linear operations designed to distort the features for enhanced privacy. Table 1 shows the network architecture. Note that $\hat{Z}$ contains 16 features with the same width and height of the raw feature X. This design facilitates seamless integration into subsequent processing stages while maintaining the original data structure.

Mask-based pruning network. Since the distortion from non-linear operations can be reconstructed easily, we develop a novel mask-based pruning network $f_p(\cdot|\theta_p)$ to generate input-dependent element-wise binary pruning filters $F = f_p(\hat{Z}|\theta_p)$ to remove sensitive feature information in $\hat{Z}$. F is element-wised multiplied to $\hat{Z}$ to produce our protected features $Z = \hat{Z} \circ F$. Note that the dimensions of F is the same as $\hat{Z}$. Our pruning network architecture is shown in Table 1. To perform binarization while preserving continuity for gradient update, we design a sigmoid activation function:

$$F_{i,j,k} = \frac{1}{1 + e^{(F'_{i,j,k} - threshold_k)/t}},$$

where F' is the output of conv2, $F'_{i,j,k}$ is the element of F' at the position (i, j, k), and t is the sigmoid temperature to enhance the sharpness of the binarization.. To control the extent of pruning, $threshold_k$ is set to the γ-portion largest elements in channel k, where γ is a predefined parameter to control the pruning ratio in $\hat{Z}$. Thus, the sigmoid function can make elements smaller than $threshold_k$ close to zero and the rest close to one. We set $\gamma = 0.95$ to activate only 0.5% pixels in the protected features to ensure that sensitive information is robustly obfuscated, reducing the likelihood of revealing critical layout details. For simplicity in later discussions, we refer to the output of the pruning as $Z = f_p(\hat{Z}|\theta_p)$ for convenience. Our mask-based pruning network enables selective retention or removal of feature elements on the per-pixel basis for layout data, providing high flexibility for our adversarial training algorithm.

Table 1: Architecture configurations of the obfuscating network f_o, the pruning network f_p, and the adversary network f_a.

Network	Layer	Kernel	#Filters	Stride	Activation
f_o	conv1	3×3	16	1	ReLU
	conv2	3×3	16	1	ReLU
f_p	conv1	3×3	16	1	ReLU
	conv2	3×3	16	1	Sigmoid
f_a	trans_conv1	3×3	16	1	ReLU
	trans_conv2	3×3	1	1	None

3.2 Task Network

The task network $f_t(\cdot|\theta_t)$ is designed to predict our targets $y = f_t(Z|\theta_t)$ with the protected features Z to simulate future model development. This setup not only simulates future model development but also crucially informs the training of the feature protector to ensure that essential information for model utility is retained in the protected features. Among the existing models designed for lithographic hotspot detection [22, 24], we select relatively fundamental models in [22] as our task networks. The model is chosen for their simplicity, which facilitates the transferability of learned protected features to more complex models in future developments. Our experimental results (Section 4.3) demonstrate that even these basic models effectively serve as indicators for preserving necessary information for accurate predictions, thereby showcasing the practicality and adaptability of PRICING.

3.3 Adversary Network

To effectively protect potential reconstruction attacks, our framework includes an adversary network, denoted $f_a(\cdot|\theta_a)$, designed to mimic the real-world attacker's behavior. The adversary network $f_a(\cdot|\theta_a)$, as shown in Table 1, takes the protected features Z as inputs to reconstruct the features $X' = f_a(Z|\theta_a)$. Recognizing that a real-world attacker might employ various architectures to decode protected data, we specifically design f_a to mirror the inverse architecture of our feature protector. This is achieved by setting the adversarial model such that each convolutional layer in the feature protector corresponds to a transposed convolution layer in f_a. In this way, we can let f_a be a powerful reconstructor by assuming the attacker knows the feature protector architecture. Additionally, this setup also ensures that our feature protector can withstand attempts to reverse-engineer protected features using advanced machine learning techniques.

3.4 Our Training Algorithm

After introducing all networks, we present our training method, designed to effectively balance information preservation in protected features with robustness against reconstruction attacks. This balance is achieved through a min-max optimization framework, which is core of our adversarial training algorithm.

To maintain the information for model training utility in our protected features, we consider the interaction between our feature protector and the task network. We use the prediction loss of the task network as a measurement of information for model training utility. Intuitively, if protected features contain sufficient information, the task network should yield high-quality prediction. The prediction loss L_t is the cross-entropy loss formulated as follows: $L_t = -(y \log(y') + (1 - y) \log(1 - y'))$, where y is the truth labels, and $y' = f_t(f_p(f_o(X|\theta_o)|\theta_p)|\theta_t)$ is the prediction results based on protected features. Then, f_o, f_p, and f_t will be jointly trained to minimize the

Algorithm 1

Output: feature protector $f_o(\cdot|\theta_o), f_p(\cdot|\theta_p)$
train $f_o(\cdot|\theta_o)$ and $f_t(\cdot|\theta_t)$ by L_t
train $f_a(\cdot|\theta_a)$ with $f_o(\cdot|\theta_o)$ by L_a
initialize weight θ_p
for every epoch
  for every batch of data
    update θ_t by L_t ▷ Eq. 1
    update θ_a by L_a ▷ Eq. 2
    update θ_o and θ_p by $-L_a + L_t$ ▷ Eq. 3
return $f_o(\cdot|\theta_o), f_p(\cdot|\theta_p)$

prediction loss L_t:

$$\min_{\theta_o,\theta_p,\theta_t} L_t. \tag{1}$$

To defend against reconstruction attacks, we consider the reconstruction loss L_a to measure the extent of feature protection in training, where $L_a = \ell_2(X, X')$ is the mean-square error between raw features X and the reconstructed ones $X' = f_a(f_p(f_o(X|\theta_o)|\theta_p)|\theta_a)$. The training goal of f_a is to minimize the reconstruction loss L_a. On the contrary, the goal of our feature protector is to maximize L_a. Therefore, the objective of the reconstruction protection can be defined as a min-max function between the feature protector and the adversary network:

$$\max_{\theta_o,\theta_p} \min_{\theta_a} L_a. \tag{2}$$

In addition, defending against reconstruction attacks can further strengthen feature obfuscation because recognizable layout information in protected features could enable adversaries to easily reconstruct raw features.

Finally, our overall training objective incorporates a balance between maintaining utility and enhancing privacy, expressed as

$$\min_{\theta_o,\theta_p} (\max_{\theta_a} -L_a + \lambda \min_{\theta_t} L_t) \tag{3}$$

by applying a negation on $\max_{\theta_o,\theta_p} \min_{\theta_a} L_a$ to opposite its min-max order, where λ is a trade-off parameter ($\lambda = 1$). In theory, the min-max formulation in Eq. 3 seeks a saddle point for all $\theta_o, \theta_p, \theta_a$, and θ_t. This saddle point represents a state where the losses associated with the adversary and the feature protector are balanced optimally.

To solve Eq. 3, we adopt a well-established adversarial training method [10, 16], which simulates training as an adversary game between the attacker who aims to minimize the reconstruction error and the protector who needs to prevent reconstructions. Our training algorithm, sketched in Algorithm 1, is designed to find an equilibrium between these opposing objectives. First, we combine the obfuscating network $f_o(\cdot|\theta_o)$ and the task network $f_t(\cdot|\theta_t)$ and pretrain them together using the prediction loss L_t. Then, we use the pretrained $f_o(\cdot|\theta_o)$ to further pretrain the adversary network $f_a(\cdot|\theta_a)$ using the reconstruction loss L_a. The pretraining step helps establish initial robustness for these networks, providing a stable starting point for the later adversarial training process. After the pretraining phase, we initialize the weights θ_p of the pruning network and proceed to our adversarial training process that trains four networks jointly. For each batch, we calculate the prediction loss L_t and the reconstruction loss L_a and apply gradient descent to update θ_t and θ_a, respectively. For the feature protector $f_o(\cdot|\theta_o)$ and $f_p(\cdot|\theta_p)$, the loss is calculated according to Eq. 3 to find a balance between feature protection and accurate prediction. After training, our optimized feature protector is returned.

Table 2: Benchmark statistics for lithographic hotspot detection.

Benchmarks	Training		Testing	
	#Hotspot	#Non-hotspot	#Hotspot	#Non-hotspot
ICCAD-1	99	340	226	3869
ICCAD-2	174	5285	498	41298
ICCAD-3	909	4643	1808	46333
ICCAD-4	95	4452	177	31890
ICCAD-5	26	2716	41	19327

Table 3: Feature protection quantitative evaluation of our method, DP, DObfus [8], and DISCO [17] for lithographic hotspot detection and routability prediction . We use our adversary network architecture (Table 1) and U-net [15] to train reconstructors (RCs).

Method	RC	Combination 1				Combination 2			
		Data providers		Model users		Data providers		Model users	
		PSNR ↓	ℓ_2 ↑	PSNR ↓	ℓ_2 ↑	PSNR ↓	ℓ_2 ↑	PSNR ↓	ℓ_2 ↑
DP	Table 1	20.748	0.046	20.665	0.055	19.350	0.062	20.925	0.050
DObfus		17.068	0.103	20.824	0.080	17.410	0.119	22.072	0.085
DISCO		14.416	0.177	15.834	0.164	18.556	0.103	20.701	0.090
Ours		**14.131**	**0.200**	**15.123**	**0.189**	**16.507**	**0.142**	**17.648**	**0.127**
DP	U-net	28.199	0.012	27.140	0.016	20.925	0.050	28.440	0.008
DObfus		31.340	0.014	34.496	0.011	24.167	0.031	31.942	0.015
DISCO		23.022	0.042	23.951	0.041	19.834	0.055	20.538	0.053
Ours		**14.927**	**0.143**	**15.549**	**0.139**	**15.758**	**0.143**	**18.460**	**0.090**

4 EXPERIMENTAL RESULTS

4.1 Experiment Setup

Benchmarks. The benchmark statistics are in Table 2. For lithographic hotspot detection, following [22, 24], we evaluate our method on an open-source dataset in ICCAD 2012 contest [19], which is composed of five different benchmarks ICCAD-1 to ICCAD-5.

Scenarios. To simulate a real-world scenario for our applications, we divide each dataset into three benchmark sets:

(1) *Protector dataset*: used to construct the feature protector.
(2) *Data provider dataset*: used to generate protected features.
(3) *Model user dataset*: consisting of unseen circuits, used to evaluate the generalizability of the model

Our *privacy-preserving dataset* is constructed by all available protected features from the protector and the data provider dataset.

We evaluate two data combinations by exchanging the protector and provider datasets. This approach mitigates potential bias from different design settings and demonstrates PRICING's wide applicability. The model user dataset remains fixed to ensure consistent assessment of performance generalization across different training data distributions.

- *Combination 1*: ICCAD-3 and ICCAD-5 (protector), ICCAD-1 and ICCAD-2 (provider)
- *Combination 2*: ICCAD-1 and ICCAD-2 (protector), ICCAD-3 and ICCAD-5 (provider)
- *Both combinations*: ICCAD-4 (model user)

Baselines. As there are no prior works on this topic in EDA, we adopt three state-of-the-art obfuscation techniques from the computer vision as our baselines, DObfus [8], DISCO [17], and a differential privacy (DP) method by adding the Laplace noise with scale 2 to the protected features. Note that these methods were not designed for improving model development, so we only compare their feature protection performance with PRICING. We implement all methods based on the network structures in Table 1 to ensure a fair comparison.

Experimental platform and hyperparameters. Our experiment are run on a machine with one NVIDIA TITAN RTX GPU with Intel® Xeon® E5-2687W CPUs. Hyperparameters of the feature protector training are detailed as follows: We perform training for 140 epochs in 0.5 hours using Adam optimizer with a learning rate of 3×10^{-4}, a batch size of 128, and a L2 regularization strength of 10^{-5}.

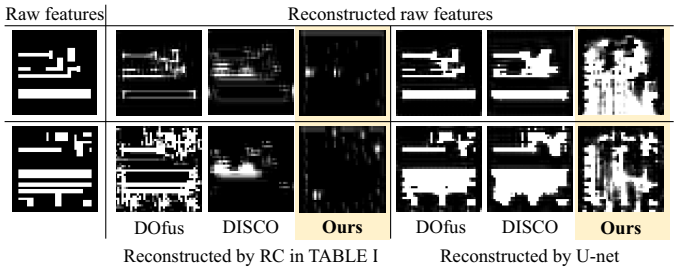

Figure 3: Qualitative evaluation of the reconstructed raw features protected by our method, DP, DObfus [8], and DISCO [17].

Table 4: Performance comparison with a CNN proposed in [22] and ResNet-18 [4] by training with three datasets: (1) all raw features of the protector and data provider dataset (golden), (2) limited raw features from the protector dataset (baseline), and (3) our privacy-preserving dataset.

Training set	Model	Combination 1		Combination 2	
		ROC-AUC ↑		ROC-AUC ↑	
		Data providers	Model users	Data providers	Model users
All raw features	CNN	0.996	0.665	0.987	0.605
Limited raw features		0.908	0.583	0.900	0.583
Ours		**0.942**	**0.760**	**0.988**	**0.678**
All raw features	ResNet	0.996	0.579	0.992	0.648
Limited raw features		0.944	0.582	0.872	0.626
Ours		**0.978**	**0.861**	**0.987**	**0.684**

Adversary scenario. We adopt the *white box attack*, which the strongest possible adversary scenario, to evaluate the feature protection based on the performance of an adversary's reconstructor (RC) model. Specifically, we grant the adversary access to the protector dataset to train the RC model from scratch on the protector dataset to learn the mapping from protected features to their corresponding raw ones. While such data access may be limited in practical scenarios, this setup allows us to evaluate the robustness of our approach against sophisticated adversaries with maximal information.

To comprehensively analyze the threat model, we consider two RC model architectures: the one in Table 1 and a representative encoder-decoder RC architecture, U-net [15]. U-net has 19 convolutional layers, which is deeper and more complex than the one in Table 1.

4.2 Feature Protection Results

We perform a comprehensive performance analysis with quantitative (Table 3) and qualitative evaluation (Fig. 3 to 4). The feature protectors are evaluated on the model user dataset and the testing set of the data provider dataset because data providers are primarily concerned with the model's generalizability and performance on their own circuits, respectively.

Quantitative evaluation. We evaluate feature protection performance by measuring the difference between raw features and reconstructed features using PSNR and ℓ_2. Lower PSNR and higher ℓ_2 mean larger feature difference and thus better protection performance. Table 3 presents the protection evaluation results. Under different combinations and RCs, our method consistently demonstrates stronger feature protection than three baselines even when attacked by U-net, which demonstrates a strong reconstruction ability. Specifically, in two combinations, the feature protector trained by our method has up to 54.9% and 42.2% PSNR improvement, respectively. Note that these remarkable results are obtained from the unseen circuits using the unseen RC model, exhibiting strong generalization capability of our protector.

Qualitative evaluation. We conduct a comprehensive qualitative evaluation through visualization for human perceptual study.

To validate the protection against reconstruction attacks, we plot and compare the reconstructed raw features protected by our method and three baselines under two RC architectures: the model in Table 1 and U-net [15]. The results are illustrated in Fig. 3. When using baselines for feature protection, both RCs can recover patterns that show a good correlation with the raw features, leaking enough information for human to recover clearer layouts. In comparison, when faced with our protected features, RCs struggle to produce meaningful output, yielding blurred and unrecognizable patterns that lack any exploitable information.

Then, we visualize the protected features produced by all methods. In Figure 4, the raw layout clips exhibit intricate polygon patterns that could potentially reveal critical design information. However, our protected features appear as sparse and seemingly random points, making it impossible to reverse-engineer the original layout patterns. In contrast, the baseline methods struggle to provide adequate protection, with their outputs often retaining noticeable similarities to the raw features, potentially enabling visual identification by human observers. The single-channel and binary nature of the layout data poses a significant challenge for existing techniques.

In summary, both quantitative and qualitative evaluations indicate that PRICING offers robust and satisfiable feature obfuscation and protection against reconstruction attacks. In addition, our method stands as the only applicable solution for this EDA application.

4.3 Privacy-Preserving Dataset Training Results

This section assesses the effectiveness of PRICING in enhancing model development under privacy constraints. We simulate two distinct development environments to compare model performances: (1) models trained with limited raw features and (2) models trained with a broad, privacy-preserving dataset obfuscated by our feature protector. These models are evaluated on both the testing set from the data provider and the model user dataset, as previously outlined in Section 4.2. Additionally, we further apply all raw features from the protector and the data provider dataset as the golden one to observe the performance effect after the feature obfuscation.

We demonstrate the adaptability of PRICING to real-world scenarios by employing two model architectures: a basic CNN [22] (our task network) and the more advanced ResNet18 [4], a representative image classification model. The performance comparison is shown in Table 4. Both CNN [22] and ResNet-18 [4] trained with our privacy-preserving dataset can clearly outperform those trained with limited raw features by up to 13.2% and 47.9% accuracy improvement on the data provider dataset and the model user dataset, respectively. Compared to models trained with all raw data, our model only has slight performance degradation evaluated on the data provider dataset. It is interesting that our model can outperform training with all raw data evaluated on the model user dataset, implying that our model can have better generalizability to unseen circuits. We posit that this is due to obfuscation of protected features, which serves as an effective regularization method to reduce overfitting.

The results validate PRICING's ability to enhance model performance and generalizability, even across different architectures beyond the task network initially used for training. These findings reflect the effectiveness of our training algorithm in preserving essential information for accurate prediction while adhering to privacy constraints. Most importantly, it confirms that enabling secured data sharing is essential for improving model development.

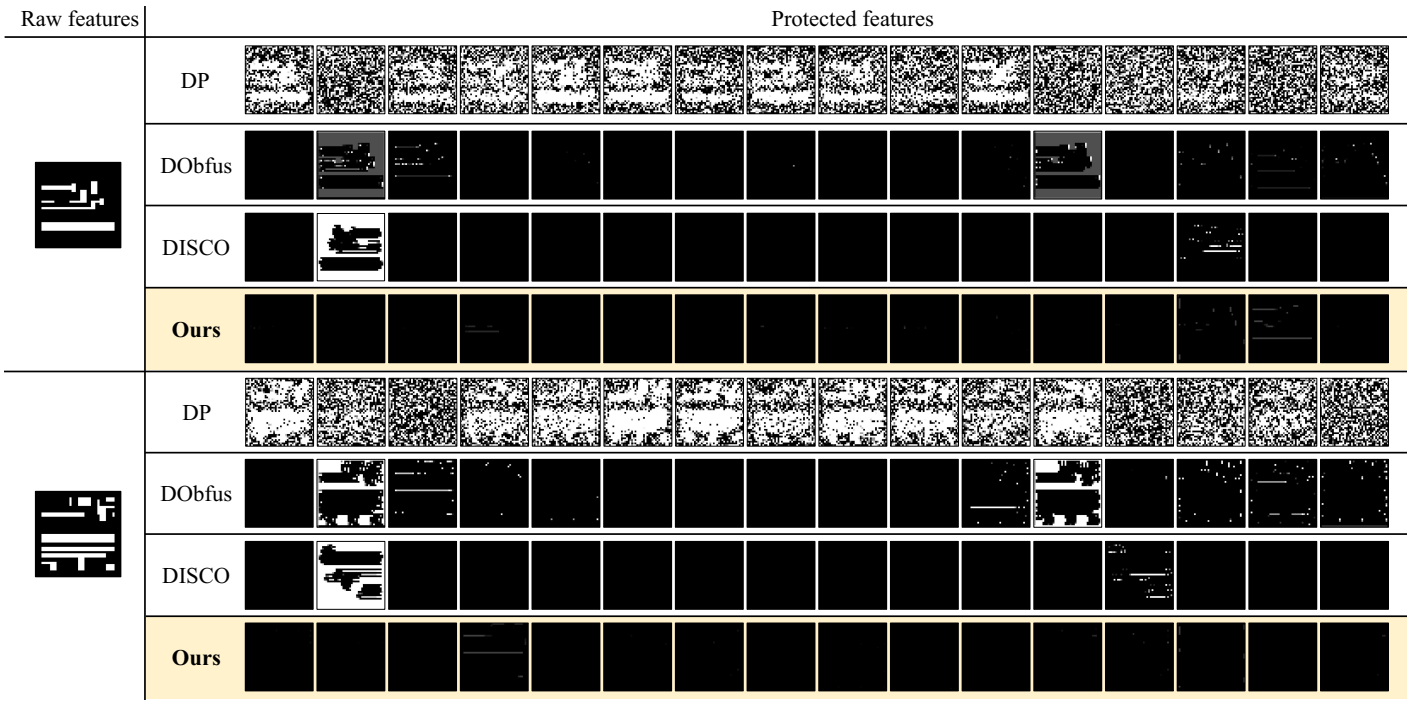

Figure 4: Qualitative evaluation of protected features using our method, DP, DObfus [8], and DISCO [17] for lithographic hotspot detection.

5 CONCLUSION

In this paper, we investigate and propose the first privacy-preserving circuit data sharing framework PRICING. Under PRICING, we develop a novel feature protector to effectively achieve privacy protection while retaining sufficient information on the protected features for future model development in lithographic hotspot detection. Based on quantitative and qualitative evaluation, PRICING has demonstrated its capability for robust feature protection. It successfully eliminates recognizable patterns and prevent reconstruction attacks with 55% better than existing methods. In addition, PRICING boosts model accuracy by up to 48% compared to training on limited data.

The promising results highlight the pivotal role of secure data sharing in advancing ML for EDA techniques. By facilitating the construction of privacy-preserving datasets, PRICING enables collaboration and training on more diverse data, thereby improving model generalizability. This addresses a long-standing bottleneck in applying data-driven techniques to modern EDA challenges. Looking ahead, we plan to extend PRICING to graph-based data as GNN-based models are also widely utilized in other EDA problems. We hope this work will stimulate increased collaboration between industry and academia for secure data sharing.

ACKNOWLEDGMENTS

This work is supported by SRC 3104.001.

REFERENCES

[1] Zhuomin Chai et al. 2022. CircuitNet: an open-source dataset for machine learning applications in electronic design automation (EDA). *arXiv preprint* (2022).
[2] Chen-Chia Chang et al. 2021. Automatic routability predictor development using neural architecture search. In *ICCAD*. IEEE, 1–9.
[3] Ran Gilad-Bachrach et al. 2016. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *ICML*. PMLR, 201–210.
[4] Kaiming He et al. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.
[5] Alain Hore and Djemel Ziou. 2010. Image quality metrics: PSNR vs. SSIM. In *ICPR*. 2366–2369.
[6] Guyue Huang, , et al. 2021. Machine learning for electronic design automation: A survey. *TODAES* 26, 5 (2021), 1–46.
[7] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. 2019. Learning not to learn: Training deep neural networks with biased data. In *CVPR*. 9012–9020.
[8] Ang Li et al. 2021. Deepobfuscator: Obfuscating intermediate representations with privacy-preserving adversarial learning on smartphones. In *IoTDI*. 28–39.
[9] R. Liang et al. 2020. DRC Hotspot Prediction at Sub-10nm Process Nodes Using Customized Convolutional Network. In *ISPD*.
[10] Sicong Liu et al. 2019. Privacy adversarial network: representation learning for mobile data privacy. *IMWUT* 3, 4 (2019), 1–18.
[11] Yi-Chen Lu et al. 2019. GAN-CTS: A generative adversarial framework for clock tree prediction and optimization. In *ICCAD*. IEEE, 1–8.
[12] Jingyu Pan et al. 2022. Towards collaborative intelligence: Routability estimation based on decentralized private data. In *DAC*. 961–966.
[13] Zhan Qin, Yin Yang, Ting Yu, Issa Khalil, Xiaokui Xiao, and Kui Ren. 2016. Heavy hitter estimation over set-valued data with local differential privacy. In *CCS*. 192–203.
[14] Wally Rhines. 2023. Taking AI to the next level. Keynote Presentation at the Design Automation Conference (DAC 2023). https://www.youtube.com/watch?v=uFWD2Tl_s3s&list=PLKqCo4MpJlW8wftD6a9IFLOikUam0zYbC&index=4 Available on YouTube.
[15] Olaf Ronneberger et al. 2015. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*. Springer, 234–241.
[16] Proteek Chandan Roy and Vishnu Naresh Boddeti. 2019. Mitigating information leakage in image representations: A maximum entropy approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2586–2594.
[17] Abhishek Singh et al. 2021. Disco: Dynamic and invariant sensitive channel obfuscation for deep neural networks. In *CVPR*. 12125–12135.
[18] Adam Smith et al. 2017. Is interaction necessary for distributed private learning?. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 58–77.
[19] J Andres Torres. 2012. ICCAD-2012 CAD contest in fuzzy pattern matching for physical verification and benchmark suite. In *ICCAD*. 349–350.
[20] Zhiyao Xie et al. 2018. RouteNet: Routability prediction for mixed-size designs using convolutional neural network. In *ICCAD*. IEEE, 1–8.
[21] Zhiyao Xie et al. 2020. Fast IR drop estimation with machine learning. In *ICCAD*. 1–8.
[22] Haoyu Yang et al. 2017. Layout hotspot detection with feature tensor generation and deep biased learning. In *DAC*. 1–6.
[23] Ryo Yonetani et al. 2017. Privacy-preserving visual learning using doubly permuted homomorphic encryption. In *ICCV*. 2040–2050.
[24] Qing Zhang et al. 2022. Litho-NeuralODE 2.0: Improving hotspot detection accuracy with advanced data augmentation, DCT-based features, and neural ordinary differential equations. *Integration* 85 (2022), 10–19.