Lithography Hotspot Detection based on Heterogeneous Federated Learning with Local Adaptation and Feature Selection

Jingyu Pan*, Xuezhong Lin*, Jinming Xu, Yiran Chen *Fellow, IEEE*, Cheng Zhuo *Senior Member, IEEE*

***Abstract*—Since the scaling of advanced technology nodes is pushing to its physical limit, lithography hotspot detection has become more significant than ever in design for manufacturability. Recently, machine learning techniques have been deployed to greatly reduce simulation time for hotspot detection, but high-quality data are required to build a model. Many design companies do not have enough high-quality data and are hesitant to share it for fear of intellectual property theft or model ineffectiveness. Furthermore, using locally trained models with limited and similar data can lead to overfitting and lack of generalization and robustness when applied to new designs. In this paper, we propose a heterogeneous federated learning framework for lithography hotspot detection that can address the aforementioned issues. Our framework can overcome the challenges of non-independent and identically distributed data and heterogeneous communication, ensuring high performance and good convergence in various scenarios. The proposed framework creates a more robust centralized global sub-model through heterogeneous knowledge sharing while keeping local data private. Then, it combines the global sub-model with a local sub-model for better adaptation to local data heterogeneity. Our experimental results show that the proposed framework outperforms other state-of-the-art methods.**

I. Introduction

As technology scaling is reaching its physical limits, the lithography process has become crucial for maintaining the Moore's law [1]. Recently, the advances in transistor technology has pushed the transistor feature size to be smaller than the light wavelength, posing challenges to lithography processing. However, recent advances in lithography processing, $e.g.$, multi-patterning, optical proximity correction, $etc.$, have made it possible to overcome the sub-wavelength lithography gap [2]. Despite such advances in lithography processing, because of the complexity of sub-14nm design rules and the process control, circuit designers have to consider the design for lithography-friendliness as part of design for manufacturability (DFM) [3].

Nowadays, lithography hotspot detection (LHD) has become no longer optional in DFM of modern sub-14nm VLSI designs. A lithography hotspot is a mask layout location that is susceptible to having fatal pinching or bridging owing to the poor printability of certain layout patterns. To avoid manufacturing failures due to poor print quality, designers usually conduct full mask lithography simulations to identify such lithography hotspots at the design stage. Despite the fact that lithography simulation is the most precise way to identify lithography hotspots, it can be very computationally costly to get a complete understanding of the chip's characteristics. To save simulation time, pattern matching and machine learning techniques have been used as more efficient alternatives [4]–[9]. For example, a hotspot library can be built to match and identify hotspot candidates [5]. In [6], low-dimensional feature vectors were extracted from layout clips and machine learning or deep learning techniques were used to predict hotspots. It is clear that *the effectiveness of the aforementioned methods is heavily reliant on both the quantity and quality of the underlying hotspot data which is used to build the library or train the model*. Without sufficient data, these methods may lack generalization ability, particularly for topologies in advanced technology nodes or unique circuit patterns.

In reality, each design company can have its own dataset on hotspots, which can be homogeneous* and does not suffice to have the model/library reach a balance point of robustness and generalability via local learning. At the same time, due to data privacy concerns, design companies are usually hesitant to share their data directly with either other companies or tool developers for *centralized learning*. To address this issue, advances in federated learning in the deep learning community offer a promising solution.

Here, we justify the need for applying federated learning in the scenario of lithography hotspot detection. After optical proximity correction, design houses are able to pinpoint layout hotspots through lithography simulation, circumventing the need to proceed to the fabrication stage. Besides, after a new technology node is developed, a foundry can only obtain limited layout patterns from some test chips. As a consequence, a design house must employ lithography simulation to identify their unique hotspot patterns and dedicate efforts to DFM development to rectify these hotspots. Nevertheless, the design

*Jingyu Pan and Xuezhong Lin contributed equally to this work.

Manuscript was received May 31, 2023; revised September 15, 2023; accepted October 20, 2023. This work is supported in parts by the National Key R&D Program of China under Grant No. 2022YFB3102100, and in parts by National Natural Science Foundation of China (NSFC) under Grants 62034007, 62141404, 62373323 and 62003302. *(Corresponding author: Cheng Zhuo.)*

Jingyu Pan and Cheng Zhuo are with the College of Information Science and Electronic Engineering at Zhejiang University (email: joeypan@zju.edu.cn, czhuo@zju.edu.cn).

Xuezhong Lin and Jinming Xu are with the College of Control Science and Engineering at Zhejiang University (email: 22032133lxz@gmail.com, jimmyxu@zju.edu.cn).

Yiran Chen is with the Department of Electrical and Computer Engineering at Duke University, Durham, NC 27708, USA (email: yiran.chen@duke.edu).

Jingyu Pan finished this work as a research assistant supervised by Cheng Zhuo. He is a Ph.D. student of Yiran Chen at the time of this submission.

*Homogeneous hotspot data refers to the hotspot candidates that share the same feature space due to similar design patterns or layout topologies.

houses are unwilling to share proprietary information about their specific hotspot patterns. This unwillingness to share data necessitates federated learning as a valuable supplement to traditional simulation methods. Unlike centralized learning which requires data to be collected at a central server or local learning which merely uses a company's own data, federated learning allows each company to train the model locally and then upload only the updated model to a central server. And the central server will aggregate and distribute the updated global model back to each company.

Though federated learning ensures no leakage of private layout information throughout model development, *its performance (or even convergence) can suffer when the data is heterogeneous (or non-Independent and Identically Distributed, i.e., non-IID)*. This is actually very common for lithography hotspot data as each design company has its unique circuit topologies or patterns, which lead to heterogeneity in lithography hotspot patterns. To address this challenge, various federated learning techniques have been proposed by the deep learning community [10]–[19], such as federated transfer learning that incorporates knowledge from the source domain [10] and federated multi-task learning that allows the model to learn shared and unique features of different tasks [11]. And to provide more local model adaptability, [12] used meta-learning to fine-tune the global model to generate different local models for different tasks. [13] defined the output layer of each client's neural network model as the personalization layer for local personalized update, which didn't explain clearly why the output layer is used as a personalization layer. [14] divided the model into global and local representations, which can result in sub-optimal results if the global representation is significantly larger compared to the local representation during the alternating model update process. [15] adds a regularization term between the local model and the global model to seek an explicit trade-off between the global model and the local model. But this trade-off is hard to learn from the small amount of private data per customer. [16] updated part of the neural network model blocks of each client individually, and proposed two model update methods. But it didn't take into account the case of model heterogeneity. [17] proposed a technology called FedMD, which uses distillation technology to uniformly aggregate the model output of each client, but it requires a feature-rich and sufficient public dataset for knowledge distillation, which is often difficult to obtain. [18] proposed Federated Mutual Learning which uses a knowledge distillation approach for personalization that applies regularization to predictions between local and global models. However, it uses a unified global model as the basis for personalization and cannot provide the optimal personalization model for clients with heterogeneous data.

In [19], a framework called FedProx was introduced that addressed statistical heterogeneity by adding a proximal regularization term to the objective function. However, this approach may not be suitable for lithography hotspot detection, which has unique characteristics compared to typical deep learning applications. Lithography hotspot detection is performed by a small number (typically between several and tens) of design companies, each of which has a relatively small amount of data (thousands to tens of thousands of layout clips). Previous federated learning methods [10]–[19] are not designed to handle these specific requirements. For instance, meta-learning may be insufficient in ensuring model consistency among local nodes when the number of nodes is small, whereas FedProx strictly enforces consistency, resulting in limited local adaptivity to support local data heterogeneity. Therefore, a balanced framework that can properly handle both local heterogeneity and global robustness is essential for effective lithography hotspot detection.

To address the aforementioned issues in centralized learning, local learning, and federated learning, in this work, we propose an **accurate and efficient lithography hotspot detection framework using heterogeneous federated learning with local adaptation**. The major contributions are summarized as follows:

- The proposed framework takes into consideration the domain knowledge of lithography hotspot detection to create a federated learning-based framework that can handle data heterogeneity. A local adaptation mechanism is implemented to balance the model's robustness against local data heterogeneity and its global accuracy.
- Instead of empirically deciding the layout feature representation, we present an efficient approach to decide the low-dimensional representation of layout clips by automatically eliminating redundant information via a regularization-based training procedure, resulting in a compact and precise feature representation.
- An heterogeneous federated learning with local adaptation (HFL-LA) algorithm is introduced to manage data heterogeneity with a combination of a global sub-model for shared knowledge and local sub-models for adapting to specific data features. A synchronization machanism is also introduced to address the communication heterogeneity issue during training.
- We present a thorough theoretical analysis to ensure the convergence of the proposed HFL-LA algorithm and to reveal the relationship between the model's hyperparameters and its convergence performance.

The experiment results demonstrate the superiority of our framework compared with other local, centralized, or federated learning methods [4], [19], [20] on both open-source and industrial layout hotspot datasets. Our framework surpasses [19], [20] with 7-11% accuracy improvement and a much lower false positive rate. Furthermore, our framework maintains its performance even when the number of clients or the size of the dataset increases, while the performance of local learning [4] deteriorates in such situations.

II. Background

A. Layout Hotspot Detection

For lithography hotspot detection, the raw dataset is comprised of hotspot and non-hotspot layout clips, each of which contains several polygonal patterns. Fig. 1(a) gives an example of a lithography layout clip. If layout clips are directly used as features without proper preprocessing for machine learning (ML)-based models, the computation cost for both model

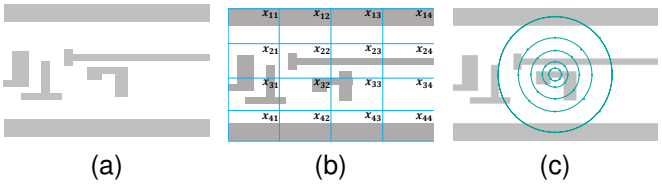


Fig. 1: (a) An example of a layout clip; (b) Local density extraction; (c) Concentric circle sampling.

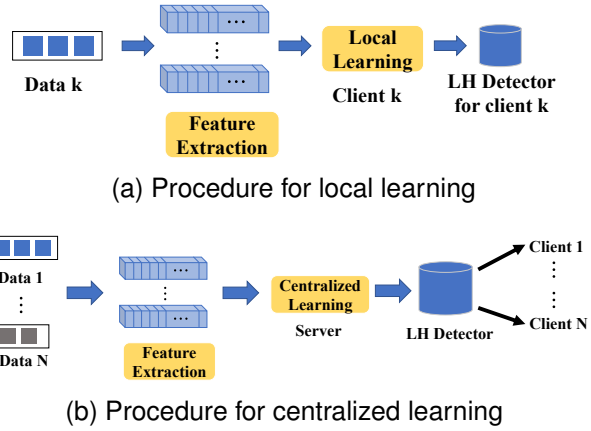


Fig. 2: Two commonly used procedures for LHD.

training and inference will be high due to the complexity of high dimensional data. To address this issue, many approaches of feature tensor extraction were proposed to reduce the data dimensionality. In earlier lithography hotspot detection and optical proximity correction works [2], [4], local density extraction and concentric circle sampling have been studied. Fig. 1(b) displays an example of local density extraction, where it converts a layout clip to a vector by calculating the density of patterns in each rectangular region. Fig. 1(c) gives an example of concentric circle sampling, where the density is sampled from the layout clip in a concentric circling way. These approaches extracts vector-based features by exploiting prior knowledge of lithography layout patterns. Indeed, they help reduce the feature complexity in ML-based lithography hotspot detection. However, since these methods ignore the spatial information surrounding the polygonal patterns within the layout clips, they *inevitably fail to utilize the spatial information* which is useful for lithography hotspot detection and usually cause low detection accuracy [4].

A promising feature extraction approach [4] is to encode the spectral domain information, which inherently reflects spatial information. For example, [4] applies discrete cosine transform (DCT) to convert a layout clip pattern into coefficients of frequency components in the spectral domain and uses the frequency coefficients as the feature representation of the layout clip. Since such a feature representation still has a high data dimension that leads to non-trivial computational overhead, [4] proposes to neglect the coefficients of high-frequency components, which are usually very sparse and thus have limited exploitable information for lithography hotspot detection. A similar approach [21] implicitly inclines on the same assumption but uses FFT for feature extraction and claims that FFT has an advantage over DCT in that it utilizes both cosine and sine functions and thus provides a stronger ability to represent the shapes, whereas DCT only uses cosine functions and is thus weaker. However, such an assumption that reducing data dimensionality by narrowing the focus of features to lower frequency components does not always hold for advanced technology nodes since they can have very subtle and abrupt variations in their pattern shapes. Consequently, this method might inadvertently fail to encode such patterns in the extracted features and thus suffer from accuracy loss. In conclusion, *current feature extraction methods either overlook potential critical features and thus compromise performance or fail to achieve optimal computation efficiency*.

There are other advances in heterogeneity-aware lithography hotspot detection. [9] brings attention to the use of the area under the ROC curve (ROC-AUC) as a more holistic metric for the highly imbalanced lithography hotspot problem and proposes a novel loss function for direct ROC-AUC optimization. [22] aims to address the reliability of common ML methods for lithography hotspot detection by introducing Gaussian process assurance that suggests the confidence of each hotspot prediction. However, few works have touched on the problem of developing ML-based lithography hotspot detectors in a privacy-preserved decentralized setting.

TABLE I: Symbols used in the proposed framework.

Symbol	Definition
w	The set of weights of a CNN model
w_g	Global weights of the model
$w_{l,i}$	Local weights of the i_{th} client model
n	Total number of clients
a_i	The data size of client i

B. Federated Learning

Federated learning allows several computation nodes to collaboratively construct a shared ML-based model without exposing a computation node's training data to any other node or any third party [20]. Consider a set of N local computation nodes, called clients, connected to a central server. Each client only has access to its own local training data and has an optimization objective $F_i : \mathbb{R}^d \to \mathbb{R}, i = 1, ..., N$.

$$\min_w f(w) = \frac{1}{N} \sum_{i=1}^{N} F_i(w), \tag{1}$$

where w denotes the model parameter, and f is the global optimization objective. FedAvg [20] is a popular federated learning algorithm that solves the above problem. In FedAvg, each client sends parameter updates of its locally trained model to the central server at the end of each training round. The server then computes the average of the collected parameter updates and deploys the average update back to all the clients. FedAvg works well with independent and identically distributed (IID) datasets but may suffer from significant performance degradation when applied to non-IID datasets.

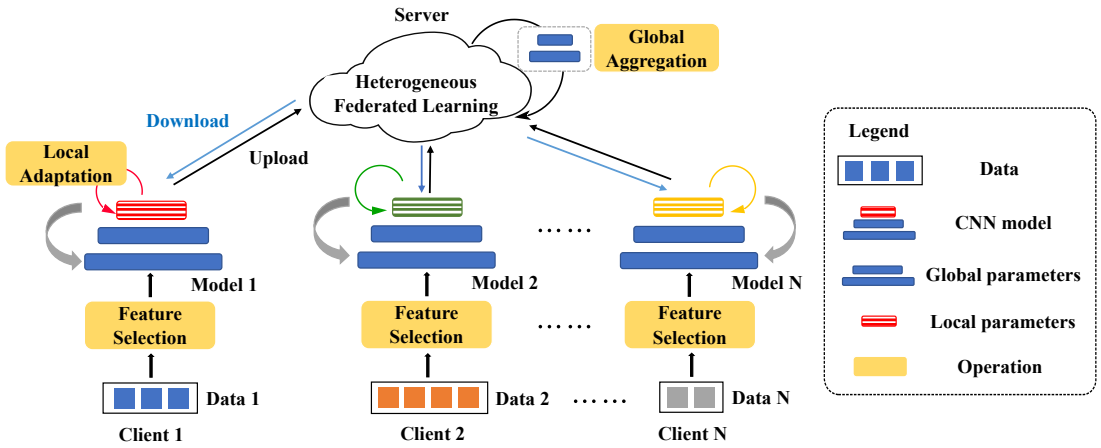


Fig. 3: Overview of the proposed framework for LHD using heterogeneous federated learning with local adaptation.

III. Proposed Framework

A. Overview

Figure 2 demonstrates procedures that are commonly used for lithography hotspot detection, *i.e.*, local learning in Figure 2(a) and centralized learning in Figure 2(b). In both procedures, feature tensor extraction and learning are two essential steps. We select these two procedures as the baseline models of our method for lithography hotspot detection. In Table I, we define the symbols that will be used in the rest of the paper.

Here we introduce the performance metrics of the lithography hotspot detection models. The accuracy of lithography hotspot detection can be evaluated by the true positive rate (TPR), the false positive rate (FPR), and the overall accuracy. These metrics are defined as follows:

Definition 1 (True Positive Rate). *The proportion of correctly classified hotspots out of the total number of classified layout hotspots.*

Definition 2 (False Positive Rate). *The proportion of incorrectly classified layout hotspots (i.e., false alarms) out of the total number of classified layout hotspots.*

Definition 3 (Accuracy). *The proportion of correctly classified hotspots and non-hotspots out of the total number of layout clips.*

With the above definitions, we summarize the formulation of the heterogeneous federated learning based lithography hotspot detection problem as follows:

Problem Formulation 1 (Heterogeneous Federated Learning Based Lithography Hotspot Detection). *Given n clients (or design companies) owning unique lithography layouts, the proposed lithography hotspot detection method aims at gathering the information from all the clients and hence construct a **local sub-model** for each client and a **global sub-model** shared by all the clients. In this way, for each client, the pair of a local sub-model and the global sub-model forms a unique hotspot detector that is dedicated to that client.*

The proposed heterogeneous federated learning (HFL)-based lithography hotspot detection method aims to adapt to the heterogeneity at different perspectives, *i.e.*, data, model, and algorithm:

- **Data**: The distribution of hotspot/non-hotspot lithography layout patterns can be non-IID.
- **Model**: The lithography hotspot detector model includes a shared global sub-model and a unique local sub-model. The local sub-model can be different from client to client during the procedure of local adaptation.
- **Algorithm**: Unlike the former federated learning method [20], our proposed framework can achieve a good convergence and accuracy when allowing asynchronous updates from the clients.

Fig. 3 presents an overview of the proposed framework which includes three key operations:

- **Feature Selection:** We propose an efficient feature selection method which automatically discovers the feature components that has critical contribution to the lithography hotspot detection model, thus reducing the redundancy in feature space and lessen computation overhead.
- **Global Aggregation:** We propose that global aggregation is only performed on the global sub-model that is shared across the clients. In this way, it not only decreases the training computation cost but also make heterogeneous communication more efficient.
- **Local Adaptation:** We propose to allow each client to optimize its local sub-model with customized parameters depending on the heterogeneity or uniqueness of local lithography layout features. This optimization process is called local adaptation.

The above three key operations construct a lithography hotspot detection framework that preserves the data privacy of each client. They allow sharing knowledge during training via federated learning and is able to maintain the balance between model generality and customization for heterogeneous local lithography features. In the remaining part of this section, we will give detailed illustration of each of the three operations.

B. Feature Selection

As discussed in Section II-A, while DCT-based methods are able to employ more spatial information than other sampling methods, they also show risks of introducing redundancy of extracted feature vectors and thereby cause unnecessary computational overhead. And to reduce the computational cost, the vectors are often truncated based on domain knowledge of lithography or other heuristics [4]. In this paper, we propose a novel feature selection technique that utilizes structured regularization to penalize unimportant feature components during model training. Note that by selecting important features, we are able to further remove the redundancy in the CNN model design, which helps improve the training convergence in the federated learning scenario.

Fig. 4 shows the proposed feature selection procedure. First, the lithography layout clips are viewed as single-channel images and are transformed into spectral domain using two-dimensional DCT. Second, we employ group LASSO-based regularization in the model training procedure to penalize feature components with less contribution [23]. We formulate the optimization penalized by group LASSO regularization as

$$L(w) = L_D(w) + R(w) + \sum_{c=1}^{C} |R_{\ell_2}(w_c)|, \tag{2}$$

where w denotes weights of the CNN-based hotspot detection model, $L_D(w)$ denotes cross entropy loss, $R(w)$ is a general regularization term, and $R_{\ell_2}(w_c)$ is structured ℓ_2 regularization on the c_{th} weight group w_c. In particular, in the first convolution layer of a deep CNN model, the parameters of each convolutional filter can be grouped by channels, each of which exactly correspond to a channel in the feature tensor. If we make the parameters from the c-th channels of all the filters a group, we have c parameter groups in total. And by applying group LASSO on these groups, the optimization would tend to prune less important parameter groups, and thus less important channels of feature tensors, $i.e.$ frequency components in spectral domain in our case. The optimization objective with the channel-wise group LASSO regularization can be expressed as:

$$L(w) = L_D(w) + \lambda_R \left|\left|w\right|\right|_2 + \lambda_{\text{GL}} \sum_{c=1}^{C^{(0)}} \left|\left| w^{(0)}_{:,c,::} \right|\right|_2, \tag{3}$$

where w is the model's weight, $w^{(0)}$ is the weight of the first convolutional layer, $w^{(0)}_{:,c,:,:}$ is the group of the c_{th} channel of layer $w^{(0)}$, λ_R is the strength of ℓ_2 regularization, and λ_{GL} is the group LASSO regularization strength. This regularization reduces the c^{th} feature channel's impact on $L_D(w)$ and encourages the ℓ_2-norm of $w^{(0)}_{:,c,::}$ to be zero if it has less significance. The remaining channels become the most important components, reducing redundancy in the layout clip feature representation and computational overhead. It's worth noting that the selected $w^{(0)}$ for feature selection is assigned as the global parameters, as shown in Fig. 3, and thus the selection result is shared among all clients.

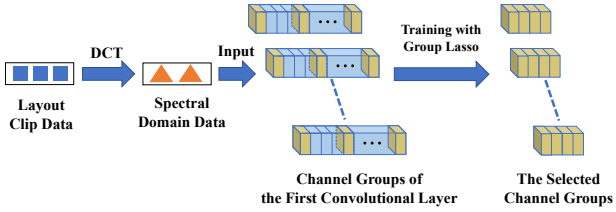


Fig. 4: The proposed feature selection procedure. Each selected channel group corresponds to a spectral domain feature channel.

C. Global Aggregation and Local Adaptation

Global aggregation and local adaptation are two essential operations in our proposed Heterogeneous Federated Learning with Local Adaptation algorithm (HFL-LA). Our proposed HFL-LA is designed for ML-based lithography hotspot detection with exploitation of lithography domain knowledge, which is summarized as follows: (1) Though different clients represent different design companies, they contain hotspot patterns that may share a non-trivial portion of similarity, which indicates the need for the global sub-model that enables knowledge sharing; (2) The total client count may hardly be larger than tens; (3) The lithography layout data at each client may not be sufficient to successfully train a model with a large local sub-model.

Fig. 3 shows the flow of our proposed HFL-LA which is similar to conventional federated learning methods, where a central server aggregates the parameters fetched from the clients. However, we highlight that, unlike conventional federated learning methods, in the proposed HFL-LA framework, the model that each client trains and uses can be split into global and local sub-models. The global sub-model is obtained from the server and shared among all clients to consolidate common knowledge for lithography hotspot detection, while the local sub-model is kept within the client to adjust to the non-IID local data, which differs from client to client.

To derive such a model, we define the following objective function for optimization:

$$\min_{w_g, w_l} \left\{ F\left(w_g, w_l\right) \triangleq \sum_{i=1}^{n} p_i F_i\left(w_g, w_{l,i}\right) \right\}, \tag{4}$$

where w_g is the global sub-model parameter shared by all the clients; $w_l := \left[w_l^1, \cdots, w_l^N\right]$ is a matrix whose k_{th} column is the local sub-model parameter for the k_{th} client; N is the number of clients; $p_k \geqslant 0$ and $\sum_{k=1}^{n} p_k = 1$ is the contribution ratio of each client; n_i is the data size of client i. By default, we can set $p_k = \frac{n_k}{n}$, where $a = \sum_{i=1}^{n} a_i$ is the total number of samples across all the clients. For the local data at client i, $F_i\left(\cdot\right)$ is the local (potentially non-convex) loss function, which is defined as

$$F_i\left(w_g, w_{l,i}\right) = \frac{1}{a_i} \sum_{j=1}^{a_i} \ell\left(w_g, w_{l,i}; x_{i,j}\right), \tag{5}$$

where $x_{i,j}$ is the j_{th} sample of client i. As shown in Algorithm 1, in the t round, the central server broadcasts the latest global sub-model parameter w_g^t to all the clients. Then, each

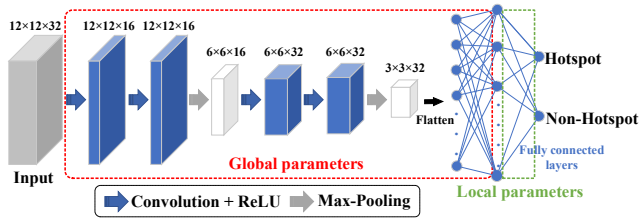


Fig. 5: An example of a CNN model in our framework.

client ($e.g.$, i_{th} client) starts with $w_{g+l,i}^{t} = w_{g,i}^{t} \cup w_{l,i}^{t}$ and conducts $E_l\ (\geqslant 1)$ local updates for sub-model parameters

$$w_{l,i}^{t+\frac{1}{2}} = w_{l,i}^{t} - \eta \sum_{j=0}^{E_l-1} \nabla_l F_i \left(w_g^t, \hat{w}_{l,i}^{t+j}; \xi_i^t \right), \tag{6}$$

where $\hat{w}_{l,i}^{t+j}$ denote the intermediate variables locally updated by client i in the t round; $\hat{w}_{l,i}^{t} = w_{l,i}^{t}$; ξ_i^t are the samples uniformly chosen from the local data in the t round of training. After that, the global and local sub-model parameters at client i become $w_{g+l,i}^{t+\frac{1}{2}} = w_g^t \cup w_{l,i}^{t+\frac{1}{2}}$ and are then updated by E_g steps of inner gradient descent as follows:

$$w_i^{t+1} = w_i^{t+\frac{1}{2}} - \eta \sum_{j=0}^{E_g-1} \nabla F_i \left(\hat{w}_{g+l,i}^{t+\frac{1}{2}+j}; \xi_i^t \right), \tag{7}$$

where $\hat{w}_{g+l,i}^{t+\frac{1}{2}+j}$ denote the intermediate variables updated by client i in the $t+\frac{1}{2}$ round; $\hat{w}_{g+l,i}^{t+\frac{1}{2}} = w_{g+l,i}^{t+\frac{1}{2}}$. Finally, the client sends the global sub-model parameters back to the server, which then aggregates the global sub-model parameters of all the clients, $i.e.$, $w_{g,1}^{t+1}, \cdots, w_{g,n}^{t+1}$, to generate the new global sub-model, w_g^{t+1}.

Algorithm 1 HFL-LA for Lithography Hotspot Detection

Server:

Initialize w_g^0, send w_g^0 to every client;
for each round $t = 0, 1, \cdots, T-1$ **do**
  $S_t \leftarrow$ (Randomly select S cilents);
  for each client $i \in S_t$ **do**
    $w_{g,i}^{t+1} \leftarrow$ ClientUpdate(i, w_g^t);
  $w_g^{t+1} \leftarrow \frac{a}{a_K} \sum_{i=1}^{S} p_i w_{g,i}^{t+1}$;
  Send w_g^{t+1} to every client.

Client:

ClientUpdate(i, w_g):
$\mathcal{B} \longleftarrow$ (Divide $\mathcal{D}_k$ according to the batch size of B);
for each local update $i = 0, 1 \cdots, E_l - 1$
  for batch $\xi_i \in \mathcal{B}$ **do do**
    $w_{l,i} \leftarrow w_{l,i} - \eta \nabla_l F_i(w_{l,i}; \xi_i)$;
for each global update $i = 0, 1 \cdots, E_g - 1$
  for batch $\xi_i \in \mathcal{B}$ **do do**
    for batch $\xi_i \in \mathcal{B}$ **do**
      $w_{g,i} \cup w_{l,i} \leftarrow w_g \cup w_{l,i} - \eta \nabla F_i(w_g \cup w_{l,i}; \xi_i)$;
return $w_{g,i}$ to server.

This figure displays the network architecture of each client involved in the experiment. The network has two convolution stages which are followed by two fully connected stages, with each stage featuring two convolution layers, a Rectified Linear Unit (ReLU) layer, and a max-pooling layer. The second fully connected layer serves as the output layer, with its outputs representing the predicted probabilities of hotspot and non-hotspot. It's also worth mentioning that the CNN-based model architecture shown in Fig. 5 is only one example for the target application, and the proposed framework can accommodate different CNN architectures in principle.

D. Communication Heterogeneity

Our framework accommodates for communication heterogeneity, meaning that clients can perform synchronized or asynchronous updates while still ensuring good convergence. In the case of synchronized updates, for each round, all clients participate in each global aggregation as:

$$w_g^{t+1} = \sum_{i=1}^{n} p_i w_{g,i}^{t+1}. \tag{8}$$

The round completes when the last client finishes its update process. In a practical scenario, however, each client's computational cost and schedule to participate in an update can vary greatly. Thus, it is more realistic to assume an asynchronous scenario where different clients will update at different rates. In this scenario, the central server can collect outputs from the first S clients that respond, with $1 \leqslant S < n$, and stop waiting for the remaining $(S+1)_{th}$ to n_{th} clients. The set of indices for the first S clients in the t_{th} round is represented as $S_t\ (|S_t| = S)$, and the global aggregation process can be rewritten as:

$$w_g^{t+1} = \frac{a}{a_S} \sum_{i \in S_t} p_i w_{g,i}^{t+1}, \tag{9}$$

where a_S is the sum of the sample data volume of the first S clients and $\frac{a}{a_S} \sum_{i \in S_t} p_i = 1$.

IV. Convergence Analysis

In this section, we study the convergence of the proposed HFL-LA algorithm. Unlike the conventional federated learning, our proposed HFL-LA algorithm for LHD works with fewer clients, smaller data volume, and non-IID datasets, making the convergence analysis more challenging. Before proceeding into the main convergence result, we provide the following widely used assumptions on the local cost functions $\{F_k\}$ and stochastic gradients [24].

Assumption 1. ***(Smoothness)*** *Each $F_i\left(\omega_g, \omega_{l,i}\right)$ is L-smooth in $\left(\omega_g, \omega_{l,i}\right) \in \mathbb{R}^{p+d_i}$.*

Assumption 2. ***(Bounded Variance)*** *For $\forall \omega_g \in \mathbb{R}^p$ and $\forall \omega_{l,i} \in \mathbb{R}^{d_i}$, there exist $\sigma_l^2, \sigma_g^2 \geqslant 0$ such that*

$$\mathbb{E}\left[\left\| \nabla_l F_i\left(w_g, w_{l,i}; \xi\right) - \nabla_l F_i\left(w_g, w_{l,i}\right) \right\|^2 \right] \leqslant \sigma_l^2,$$
$$\mathbb{E}\left[\left\| \nabla_g F_i\left(w_g, w_{l,i}; \xi\right) - \nabla_g F_i\left(w_g, w_{l,i}\right) \right\|^2 \right] \leqslant \sigma_g^2.$$

TABLE II: The ICCAD and Industry benchmark details.

Benchmarks	Size/Clip (μm^2)	Training Set		Testing Set	
		HS#	non-HS#	HS#	non-HS#
ICCAD	3.6×3.6	1204	17096	2524	13503
Industry	1.2×1.2	3629	80299	942	20412

Assumption 3. ***(Bounded gradient)*** *For* $\forall \omega_g \in \mathbb{R}^p$ *and* $\forall \omega_{l,i} \in \mathbb{R}^{d_i}$*, there exist* $D_l^2, D_g^2 \geqslant 0$ *such that*

$$\left\|\nabla_l F_i\left(w_g, w_{l,i}\right)\right\|^2 \leqslant D_l^2, \quad \left\|\nabla_g F_i\left(w_g, w_{l,i}\right)\right\|^2 \leqslant D_g^2.$$

With the above assumptions, we are ready to present the following main results of the convergence of the proposed algorithm. *The detailed proof can be found in the Appendix.*

Lemma 1. ***(Consensus Error)*** *Suppose Assumption 1-3 hold. Then, we have for all* $k \geqslant 0$*,*

$$\mathbb{E}\left[\left\|w_g^k - \mathbf{1}\bar{w}_g^k\right\|^2\right] \leqslant n\eta^2\left(E_g - 1\right)^2\left(D_g^2 + \sigma_g^2\right). \tag{10}$$

Theorem 1. *Suppose Assumption 1-3 hold. Let the step-size satisfies* $\eta \leqslant 1/L$*, we have for all* $T \geqslant 0$*,*

$$\begin{aligned} &\frac{1}{T+1}\sum_{t=0}^{T}\left(\frac{1}{n}\mathbb{E}\left[\left\|\nabla F\left(\mathbf{1}\bar{w}_g^{t\tau}, w_l^{t\tau}; \xi^k\right)\right\|^2\right]\right) \\ &\leqslant \frac{2\left(F\left(\bar{w}_g^0, w_l^0\right) - F^*\right)}{T\eta} + \eta\tau L\sigma_l^2 + \frac{\eta E_g L\sigma_g^2}{n} \\ &+ 2\tau\eta^2 L^2\left(E_g - 1\right)^2\left(D_g^2 + \sigma_g^2\right). \end{aligned} \tag{11}$$

Remark 1. *The above lemma guarantees that the global sub-model parameters of all the clients reach consensus with an error proportional to the learning rate* η*. Besides, the above theorem further shows that, with a constant step-size, the parameters of all clients converge to the* η*-neighborhood of a stationary point with a rate of* $\mathcal{O}\left(1/T\right)$*. It should be noted that the second term of the steady-state error will vanish when* $E_g = 1$*. This theorem sheds light on the relationship between design parameters and convergence performance, which helps guide the design of the proposed HFL-LA algorithm.*

V. Experiment Results

The proposed framework is implemented based on the PyTorch library [25]. In our experiments, we use the following hyperparameters to guide the training process of the CNN-based model on each client: We optimize our models with the Adam optimizer for $T = 50$ rounds. We select a learning rate $\eta = 0.001$, a batch size of 64 and L2 regularization strength of 0.00001. Furthermore, in each round, we perform local updates for $E_l = 500$ iterations, and global updates for $E_g = 1500$ iterations. Two distinct benchmarks (ICCAD and Industry) are used in our experiments to train and evaluate our framework. The test cases published in ICCAD 2012 contest [26] contain lithography patterns of the 28nm technology node. We combined all these patterns into a merged benchmark, denoted by ICCAD, and obtained the Industry benchmark using layout data at a 20nm technology node from our industrial partner. Table II provides details on the benchmarks, including the size of the training and testing sets and the layout clip size. The columns labeled "HS#" and "non-HS#" show the total number of hotspots and non-hotspots, respectively. The benchmark is divided at random into separate portions, with each client being allocated one distinct portion. Specifically, we ensure the size of each portion is uniformly distributed and the maximum size can be 4 times as large as the smallest one. This data partitioning strategy introduces a nuanced balance between flexibility and consistency, fostering a heterogeneous data distribution that mirrors real-world scenarios, thus enhancing the applicability and adaptability of our experiments. Given the limited access to public lithography data and intellectual property concerns from companies, it is our best effort to simulate data heterogeneity. This is achieved by adjusting quantities, clip sizes, and symbolizing distinct tech nodes of the lithography layout clips. Despite these constraints, our existing framework successfully mirrors a typical level of data heterogeneity often found in real-world scenarios.

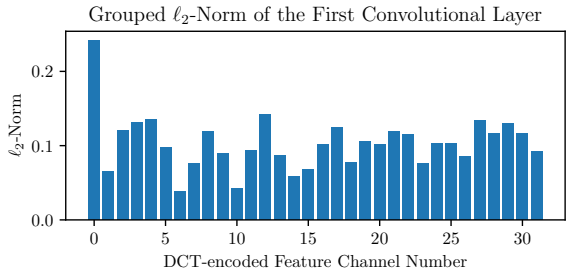


Fig. 6: The grouped ℓ_2-norm of the first convolution layer is presented. The range of the DCT-encoded channel number is from 0 to 31, with channel 0 representing the DC component of the spectral domain data and channels 1 to 31 representing the AC components in increasing frequency.

One minor issue about the data is that the sizes of the original layout clips from ICCAD and Industry are different. In order to achieve consistent clip sizes, the layout clips in the ICCAD benchmark are divided into nine blocks, ensuring that the size is consistent with the Industry benchmark. However, it's important to note that the two benchmarks have different feature representations due to differences in technology and design patterns. The Industry benchmark, in particular, has a higher degree of data heterogeneity with more diverse design patterns.

A. Feature Selection

This section presents the evaluation of the proposed feature selection method. As described in Sec. III-B, the ℓ_2-norm of the channel-wise groups in the first convolutional layer is related to the impact of the corresponding feature channels on model performance, as shown in Figure 6. Figure 6 intuitively proves the concept that different frequency components in the feature space have very different contributions to the model in terms of their weights during model inference. The feature channels were then sorted by their ℓ_2-norm and the model was retrained with only the top-c channels, where $c = 26$ in our experiment. To validate the effectiveness of the feature selection method, the performance of HFL-LA was tested with

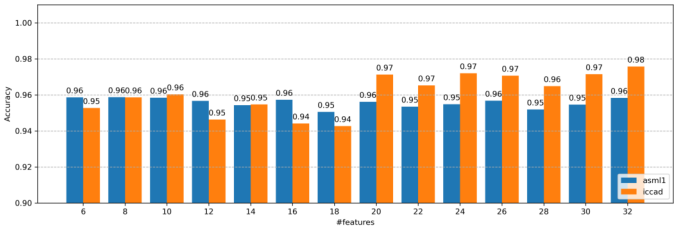


Fig. 7: Accuracy of HFL-LA on the validation set using a different number of selected features representing the layout clip.

different numbers of features representing the layout clips and compared on the validation set. Figure 6 demonstrates that HFL-LA achieves comparable or even higher accuracy with $c = 26$ features as suggested by the proposed selection method for both benchmarks, which represents a 18.75% reduction of computational cost for the subsequent learning compared to the original 32 features.

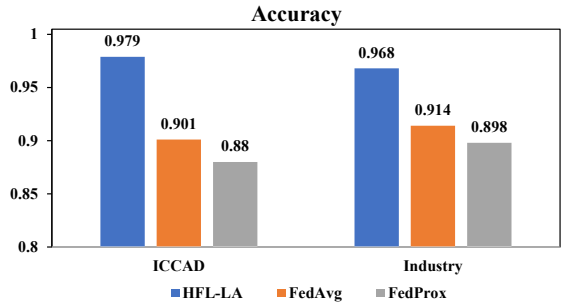
We also analyze the contribution of feature selection in our HFL-LA framework. Figure 7 shows the validation accuracy of HFL-LA when the model is trained with a different number of selected features representing the layout clips. When no feature selection is performed and all the 32 features are used for training, HFL-LA reports validation accuracy of 98% on the `ICCAD` dataset and 96% on the `Industry` dataset. And when we select the top-c ($c \geq 6$) features, for the `ICCAD` dataset, the HFL-LA framework achieves a comparable accuracy of 97% when $c = 20$, and even when $c = 6$, the accuracy is still 95% with a mere 3% drop. For the `Industry` dataset, we show that the most important 6 features provide enough information to achieve the same accuracy as the total 32 features. This result shown in Figure 7 proves the existence of unnecessary computation overhead in the ML model development with the DCT-based feature extraction method.

B. Heterogeneous Federated Learning with Local Adaptation

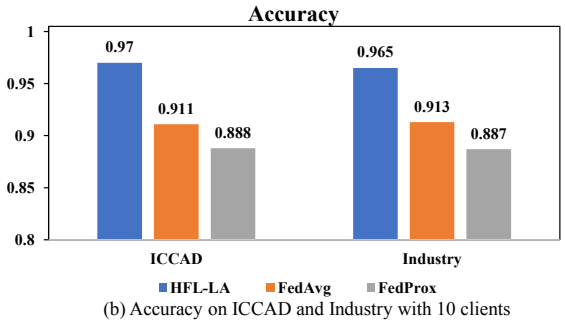
To evaluate the effectiveness of the proposed HFL-LA algorithm, we compare its results with the state-of-the-art federated learning algorithms FedAvg and FedProx, as well as with local and central learning methods, which were described in [4], [19], [20]. The following summarizes the algorithms compared:

- FedAvg: A conventional federated learning algorithm that averages the uploaded models [20].
- FedProx: A federated learning algorithm that handles heterogeneity by adding a proximal term to the objective [19].
- Local learning (denoted as "local"): A learning method that only uses the local data of each client [4].
- Central learning (denoted as "centralized"): A learning approach that trains a unified model using all available training sets [4].

In this experiment, the merged training sets of the `ICCAD` and `Industry` benchmarks were divided and assigned to different client numbers (2, 4, and 10) as their local data. The testing sets, as shown in Table II, were kept separate



(a) Accuracy on ICCAD and Industry with 4 clients



(b) Accuracy on ICCAD and Industry with 10 clients

Fig. 8: Accuracy comparison among HFL-LA, FedAvg, and FedProx on `ICCAD` and `Industry` with 4 and 10 clients using asynchronous model updates.

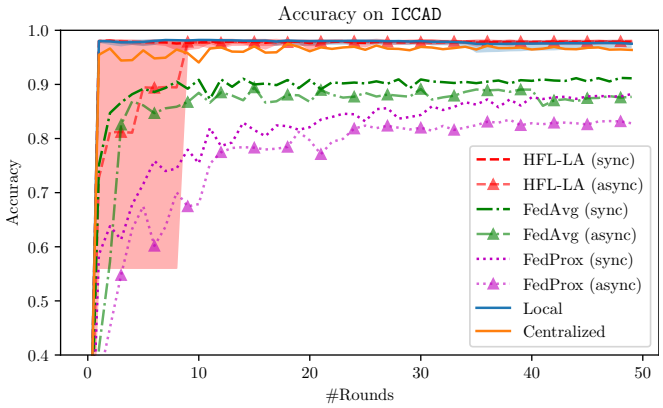
and used to evaluate the performance of the trained models. The algorithms were compared based on their True Positive Rate (TPR), False Positive Rate (FPR), and accuracy. Table III summarizes the results. For each experiment, we collect results from 5 parallel runs with different random seeds for model parameter initialization, and report the average and standard deviation. All clients communicated with the server following a synchronized schedule, and the average performance across all clients in the three scenarios (2, 4, and 10 clients) was calculated. The best performance in each scenario is marked in bold. The proposed HFL-LA algorithm showed an improvement of 7-11% in accuracy for TPR and FPR compared to FedAvg and FedProx. Although local learning, which only uses homogeneous local data, performed slightly better on the `ICCAD` benchmark, its performance quickly dropped when the data heterogeneity increased, as seen in the `Industry` benchmark, yielding a degradation of around 4% compared to HFL-LA.

We also compare the results when the model updates are done asynchronously for 4 and 10 client scenarios, where half of the clients are randomly selected for training and updating in each round. It's pivotal to underscore that only federated learning techniques mandate these model updates. Hence, our comparison predominantly zeroes in on HFL-LA versus the FedAvg and FedProx methods. As illustrated in Fig. 8, the HFL-LA method shines brightly, even in the face of inconsistent communication and variegated updates. When pitted against other federated learning techniques, HFL-LA showcases a marked performance enhancement, with accuracy figures rising by a notable 5-10%. This robustness and superior performance firmly position HFL-LA as a preferred choice when considering federated learning approaches.

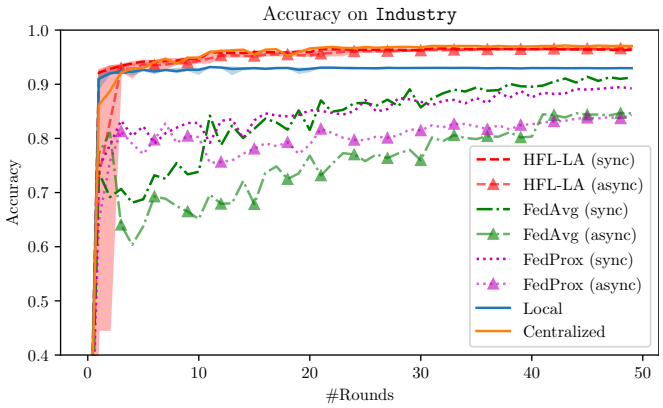
TABLE III: Inference performance (TPR, FPR and accuracy) comparison among HFL-LA, FedAvg, FedProx, local & central learning with standard deviation included. All experiments are repeated 5 times with different random seeds.

Methods	Number of clients	ICCAD			Industry		
		TPR	FPR	ACC	TPR	FPR	ACC
HFL-LA	2 clients	0.961±0.008	0.020±0.005	**0.981**±0.007	0.967±0.009	0.041±0.006	0.965±0.008
	4 clients	0.968±0.006	0.022±0.004	**0.980**±0.006	0.976±0.006	0.050±0.005	**0.969**±0.006
	10 clients	0.968±0.004	0.031±0.003	0.971±0.004	0.972±0.004	0.051±0.003	**0.966**±0.004
FedAvg	2 clients	0.975±0.012	0.111±0.015	0.893±0.013	0.815±0.014	0.011±0.008	0.870±0.012
	4 clients	0.972±0.010	0.102±0.012	0.902±0.010	0.884±0.011	0.017±0.007	0.915±0.010
	10 clients	0.970±0.006	0.091±0.008	0.912±0.006	0.882±0.007	0.017±0.004	0.914±0.006
FedProx	2 clients	0.978±0.015	0.135±0.018	0.869±0.016	0.855±0.017	0.015±0.010	0.896±0.015
	4 clients	0.974±0.013	0.122±0.015	0.881±0.014	0.860±0.014	0.018±0.009	0.899±0.013
	10 clients	0.959±0.007	0.114±0.008	0.889±0.007	0.844±0.008	0.017±0.005	0.888±0.007
Local	2 clients	0.974±0.006	0.022±0.004	0.979±0.006	0.977±0.006	0.040±0.004	**0.972**±0.006
	4 clients	0.967±0.005	0.022±0.005	0.979±0.007	0.972±0.007	0.072±0.006	0.958±0.007
	10 clients	0.926±0.005	0.025±0.004	**0.976**±0.005	0.955±0.005	0.124±0.004	0.931±0.004
Centralized	1 server	0.957±0.010	0.033±0.008	0.969±0.009	0.975±0.010	0.039±0.007	0.971±0.009

Lastly, we compare the accuracy of different methods with both synchronous (denoted as sync) and asynchronous (denoted as async) update mechanisms for 10 clients. For the `ICCAD` benchmark, as shown in Fig. 9(a), our HFL-LA method achieve the highest accuracy and converge much faster than the other methods in the scenario of synchronous updates. The convergence rate of HFL-LA is even comparable to local learning. Even with asynchronous updates, the HFL-LA method can still achieve a convergence rate and accuracy that are similar to those in the synchronous update scenario. As for the `Industry` benchmark, as shown in Fig. 9(b), the HFL-LA method also outperforms all the other methods in terms of accuracy (e.g., improvement of 3.7% over local learning). Furthermore, the HFL-LA method even reaches around 5× convergence speedup compared with the other federated learning methods, like FedAvg and FedProx, even adopting asynchronous updates.



(a) Accuracy on ICCAD testing set

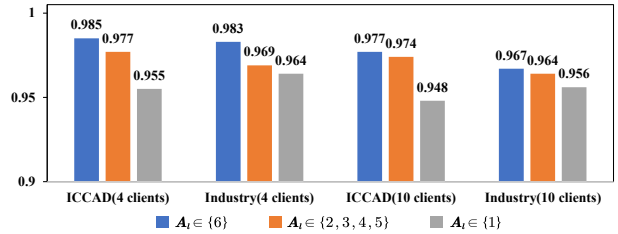


(b) Accuracy on Industry testing set

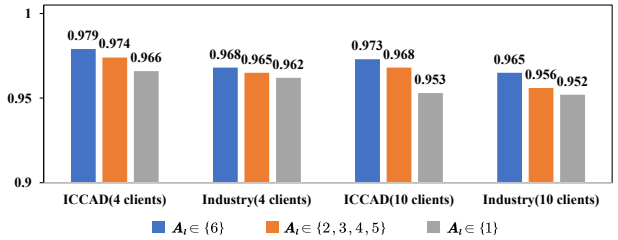
Fig. 9: The comparison of convergence between various methods during training, where model evaluation is performed on the testing sets for `ICCAD` and `Industry`.

C. Choice of Personalization Adaptation Layers

We further explore the effectiveness of the HFL-LA algorithm when using different CNN model layers as local parameters. As shown in Fig. 5, our CNN model has a total of 4 convolutional layers and 2 fully connected layers. Starting from the first convolutional layer, we number all the layers of the CNN model as {1,2,3,4,5,6}. We consider that the local parameters should be the classifier layer (the final fully connected layer). Since we describe local parameters in units of CNN model layers, with a slight abuse of notation, we can use A_l to denote the CNN model layers included in the local parameters. $A_l \in \{1\}$ refers to the first convolutional layer as the local parameters. $A_l \in \{2,3,4,5\}$ refers to the final fully connected layer as the local parameters. $A_l \in \{6\}$ refers to using the layers in the middle of the CNN model as the local parameters. Fig. 10 plots test accuracies (averaged across clients) comparison among $A_l \in \{1\}$, $A_l \in \{2,3,4,5\}$, and $A_l \in \{6\}$ on `ICCAD` and `Industry` with 4 and 10 clients using synchronous and asynchronous model update. Interestingly, there seem to be a clear correlation between A_l



(a) Accuracy using synchronous model updates.



(b) Accuracy using asynchronous model updates.

Fig. 10: Accuracy comparison among $A_l \in \{1\}$, $A_l \in \{2,3,4,5\}$, and $A_l \in \{6\}$ on `ICCAD` and `Industry` with 4 and 10 clients using synchronous and asynchronous update.

and the client averaged test accuracy at steady state. HFL-LA has the highest accuracy with $A_l \in \{6\}$, achieving 1% accuracy improvement from that of the other methods. As shown in Table II, the label distributions of `ICCAD` and `Industry` datasets are highly heterogeneous, so it is most reasonable to choose the final fully connected layer as the local parameter. As shown in the experimental results, even though the parameter scale of the final fully connected layer is only 0.68% of the parameter scale of the entire CNN model layers, it achieves the best accuracy.

D. CNN model heterogeneity

While it's accurate that the optimal CNN architecture can vary based on the characteristics of different datasets, a homogeneous architecture requirement in a federated learning environment can indeed limit individual performance. To address this, in our HFL-LA approach, we entertain the possibility of customizing CNN architectures for individual datasets.

For the `Industry` dataset, characterized by its complex feature expression, we enhanced the basic CNN model by adding an extra fully connected layer to the architecture depicted in Fig.5. The modified architecture, specifically tailored for the `Industry` dataset, is illustrated in Fig.11. Upon the aggregation of global sub-model parameters, the server disseminates this information to all clients. Each client then proceeds to train its local sub-model parameters using its private dataset. This creates a collaborative training environment where each model retains its unique architecture while benefiting from the shared insights. This leads to rapid model improvements, surpassing traditional federated learning baselines. Our experimental results, as displayed in Table IV, confirm this approach's efficacy. The customized models achieved test accuracies of approximately 97.5% on the `ICCAD` dataset and 96.2% on the `Industry` dataset.

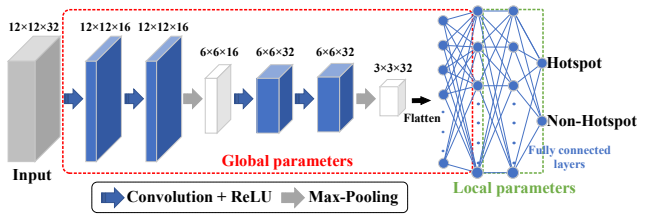


Fig. 11: The CNN model corresponding to Industry. This model differs from the one shown in Fig. 5 in that it shows a different configuration of the local parameters.

TABLE IV: Comparison of accuracy for HFL-LA and FedAvg [20]. Each entry provides the average value accompanied by the standard deviation (avg±std). All experiments are repreated 5 times with different random seeds.

Number of clients	ICCAD		Industry	
	FedAvg	HFL-LA	FedAvg	HFL-LA
4 (sync)	0.902±0.018	**0.980**±0.014	0.914±0.013	**0.964**±0.012
10 (sync)	0.908±0.006	**0.968**±0.006	0.913±0.004	**0.960**±0.003
4 (async)	0.878±0.021	**0.977**±0.010	0.892±0.017	**0.959**±0.008
10 (async)	0.892±0.009	**0.973**±0.005	0.882±0.006	**0.963**±0.003

This approach can be extended to further improve performance. For instance, clients could use additional layers, alternative activation functions, or different types of layers (such as convolutional, pooling, or normalization layers) based on the specific characteristics of their datasets.

E. Performance on different sizes of clients

We also explore the performance of HFL-LA when the size of each local client varies. Fig. 13 and Fig. 12 show the HFL-LA accuracy and local accuracy on different splitting of the `ICCAD` dataset and `Industry` dataset, respectively. Note that the size of each client's data is inversely proportional to the number of clients split from either dataset. When either of the two datasets is split into 10 clients, the HFL-LA accuracy is 97% on `ICCAD` and 96% on `Industry`, while the local accuracy is 97% on `ICCAD` and 91% on `Industry`. When the number of clients split from either dataset increases to 25, local accuracy on `Industry` decreases significantly to 87%, which is a 4.4% drop. On the other hand, the HFL-LA accuracy merely decreases from 96% to 95%, which is only a 1% drop. This result shows that when the data on each client is insufficient for successful local training, HFL-LA can utilize information gathered from decentralized clients and thus outperforms local training.

VI. Conclusion

We have proposed a new hotspot detection framework that uses heterogeneous federated learning with local adaptation. The framework takes advantage of an efficient feature selection method and domain expertise of lithography hotspot detection to handle heterogeneity in data, model, and communication. Experiment results demonstrate that our framework surpasses other methods in terms of performance and has better convergence compared to other federated learning methods, even when datasets are highly heterogeneous.

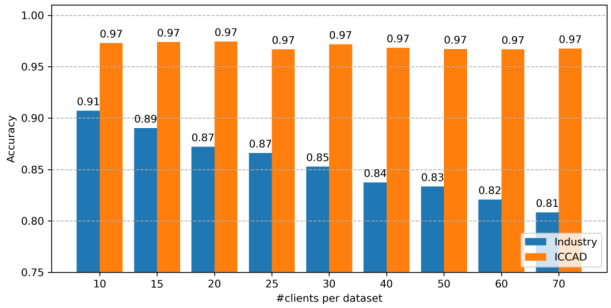


Fig. 12: Local accuracy of models trained on clients of different sizes.

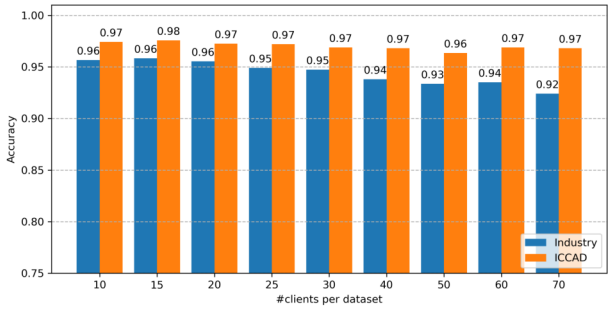


Fig. 13: Accuracy of HFL-LA on the validation set with different number of clients in the training set.

ACKNOWLEDGMENTS

The authors would like to thank the suggestions from the Editor and reviewers.

APPENDIX

In this section, we prove the lemmas and the theorem mentioned above. For brevity, we only consider the case where the number of sampled clients $S = n$. However, the techniques used to prove the main results can be extended to other cases with different updating strategies on the global and local model parameters. We use the following notations:

$$
\begin{aligned}
&G_l\left(w_g^k, w_l^k; \xi^k\right) := \left[\cdots, \nabla_l F_i\left(\omega_{g,i}^k, \omega_{l,i}^k; \xi_i^k\right), \cdots\right]^T, \\
&G_g\left(w_g^k, w_l^k; \xi^k\right) := \left[\cdots, \nabla_g F_i\left(\omega_{g,i}^k, \omega_{l,i}^k; \xi_i^k\right), \cdots\right]^T, \\
&\bar{\omega}_g^k := \frac{\mathbf{1}^T}{n}\omega_g^k, \ \tau = E_l + E_g, \ m := \lfloor (k-1)/\tau \rfloor,
\end{aligned}
$$

where k denotes the count of overall iterations, m denotes the number of global communications before k. Then, we can rewrite the proposed HFL-LA algorithm as follows:

$$
\begin{aligned}
\omega_l^{k+1} &= \omega_l^k - \eta G_l\left(w_g^k, w_l^k; \xi^k\right), \\
\omega_g^{k+1} &= W^k \omega_g^k - \alpha_k G_g\left(w_g^k, w_l^k; \xi^k\right).
\end{aligned}
$$

where

$$
W^k := \begin{cases} \mathbf{J}, & \text{if } \mathbf{mod}\left(k, E_l + E_g\right) = 0 \\ \mathbf{I}, & \text{else} \end{cases},
$$

$$
\alpha_k := \begin{cases} \eta, & \text{if } \mathbf{mod}\left(k, E_l + E_g\right) \geqslant E_l \\ 0, & \text{else} \end{cases}.
$$

A. Supporting lemmas

We first provide the proof for Lemma 1 which bounds the consensus error in expectation.

Proof: By the above rewritten algorithm, we have

$$
\begin{aligned}
&w_g^k - \mathbf{1}\bar{w}_g^k \\
&= \left(W_{k-1} - \mathbf{J}\right)\left(\omega_g^{k-1} - \mathbf{1}\bar{\omega}_g^{k-1}\right) \\
&- \alpha_k \left(W_{k-1} - \mathbf{J}\right) G_g\left(w_g^{k-1}, w_l^{k-1}; \xi^{k-1}\right) \\
&= \prod_{s=0}^{k-1-m\tau} \left(W_{k-1-s} - \mathbf{J}\right)\left(\omega_g^{m\tau} - \mathbf{1}\bar{\omega}_g^{m\tau}\right) \\
&- \sum_{t=k-1}^{m\tau} \alpha_t \left(\prod_{s=0}^{k-1-t} \left(W_{k-1-s} - \mathbf{J}\right)\right) G_g\left(w_g^t, w_l^t; \xi^t\right) \\
&= -\eta \sum_{t=k-1}^{m\tau+E_l} (\mathbf{I} - \mathbf{J}) G_g\left(w_g^t, w_l^t; \xi^t\right),
\end{aligned}
\tag{12}
$$

where $m := \lfloor (k-1)/\tau \rfloor$ and $\tau := E_l + E_g$. By the definitions of α_k, W^k and Assumption 2 and 3, we get

$$
\mathbb{E}\left[\left\|w_g^k - \mathbf{1}\bar{w}_g^k\right\|^2\right] \leqslant \eta^2 \left(E_g - 1\right)^2 n \left(D_g^2 + \sigma_g^2\right), \tag{13}
$$

which completes the proof. ■

Lemma 2. *Suppose Assumption 1-3 hold. Let the step-size satisfy $\eta \leqslant 1/L$. Then, we have for all $k \geqslant 0$,*

$$
\begin{aligned}
&\mathbb{E}\left[F\left(\bar{w}_g^{k+1}, w_l^{k+1}\right)\right] \\
&\leqslant \mathbb{E}\left[F\left(\bar{w}_g^k, w_l^k\right)\right] + \frac{\left(\alpha_k + \eta\right) L^2}{2n} \mathbb{E}\left[\left\|w_g^k - \mathbf{1}\bar{w}_g^k\right\|^2\right] \\
&- \frac{\alpha_k}{2n} \mathbb{E}\left[\left\|G_g\left(\mathbf{1}\bar{w}_g^k, w_l^k\right)\right\|^2\right] - \frac{\eta}{2n} \mathbb{E}\left[\left\|G_l\left(\mathbf{1}\bar{w}_g^k, w_l^k\right)\right\|^2\right] \\
&+ \frac{\eta^2 L \sigma_l^2}{2} + \frac{\alpha_k^2 L \sigma_g^2}{2n}.
\end{aligned}
\tag{14}
$$

Proof: Since each F_i is L-smooth, we have

$$
\begin{aligned}
&F_i\left(\bar{w}_g^{k+1}, w_{l,i}^{k+1}\right) \leqslant F_i\left(\bar{w}_g^k, w_{l,i}^k\right) \\
&+ \left\langle \nabla F_i\left(\bar{w}_g^k, w_{l,i}^k\right), \left(\bar{w}_g^{k+1}, w_{l,i}^{k+1}\right) - \left(\bar{w}_g^k, w_{l,i}^k\right) \right\rangle \\
&+ \frac{\eta^2 L}{2} \left\|\nabla_l F_i\left(w_{g,i}^k, w_{l,i}^k; \xi_i^k\right)\right\|^2 \\
&+ \frac{\alpha_k^2 L}{2} \left\| \frac{\mathbf{1}^T}{n} G_g\left(w_g^k, w_l^k; \xi^k\right) \right\|^2.
\end{aligned}
\tag{15}
$$

Then, we bound the inner product in the above inequality. Noticing that

$$
\begin{aligned}
&\left\langle \nabla F_i\left(\bar{w}_g^k, w_{l,i}^k\right), \left(\bar{w}_g^{k+1}, w_{l,i}^{k+1}\right) - \left(\bar{w}_g^k, w_{l,i}^k\right) \right\rangle \\
&= \left\langle \nabla_g F_i\left(\bar{w}_g^k, w_{l,i}^k\right), \alpha_k \frac{\mathbf{1}^T}{n} G_g\left(w_g^k, w_l^k; \xi^k\right) \right\rangle \\
&+ \left\langle \nabla_l F_i\left(\bar{w}_g^k, w_{l,i}^k\right), \eta \nabla_l F_i\left(w_{g,i}^k, w_{l,i}^k; \xi_i^k\right) \right\rangle,
\end{aligned}
\tag{16}
$$

by the smoothness of F_i, we can then obtain

$$\begin{aligned}
&\mathbb{E}\left[\left\langle \nabla_g F_i\left(\bar{w}_g^k, w_{l,i}^k\right), \alpha_k \frac{\mathbf{1}^T}{n} G_g\left(w_g^k, w_l^k; \xi^k\right)\right\rangle\right] \\
&\geqslant \frac{\alpha_k}{2}\left(\mathbb{E}\left[\left\|\nabla_g F_i\left(\bar{w}_g^k, w_{l,i}^k\right)\right\|^2\right] + \mathbb{E}\left[\left\|\frac{\mathbf{1}^T}{n} G_g\left(w_g^k, w_l^k\right)\right\|^2\right]\right) \\
&- \frac{\alpha_k}{2}\mathbb{E}\left[\left\|\nabla_g F_i\left(\bar{w}_g^k, w_{l,i}^k\right) - \frac{1}{n}\sum_{i=1}^n \nabla_g F_j\left(w_{g,i}^k, w_{l,i}^k\right)\right\|^2\right] \\
&\geqslant \frac{\alpha_k}{2}\left(\mathbb{E}\left[\left\|\nabla_g F_i\left(\bar{w}_g^k, w_{l,i}^k\right)\right\|^2\right] + \mathbb{E}\left[\left\|\frac{\mathbf{1}^T}{n} G_g\left(w_g^k, w_l^k\right)\right\|^2\right]\right) \\
&- \frac{\alpha_k L^2}{2n}\mathbb{E}\left[\left\|w_g^k - \mathbf{1}\bar{w}_g^k\right\|^2\right],
\end{aligned} \tag{17}$$

and

$$\begin{aligned}
&\mathbb{E}\left[\left\langle \nabla_l F_i\left(\bar{w}_g^k, w_{l,i}^k\right), \eta \nabla_l F_i\left(w_{g,i}^k, w_{l,i}^k; \xi_i^k\right)\right\rangle\right] \\
&= \eta\mathbb{E}\left[\left\langle \nabla_l F_i\left(\bar{w}_g^k, w_{l,i}^k\right), \nabla_l F_i\left(w_{g,i}^k, w_{l,i}^k\right)\right\rangle\right] \\
&\geqslant \frac{\eta}{2}\left(\mathbb{E}\left[\left\|\nabla_l F_i\left(\bar{w}_g^k, w_{l,i}^k\right)\right\|^2\right] + \mathbb{E}\left[\left\|\nabla_l F_i\left(w_{g,i}^k, w_{l,i}^k\right)\right\|^2\right]\right) \\
&- \frac{\eta L^2}{2}\mathbb{E}\left[\left\|w_{g,i}^k - \bar{w}_g^k\right\|^2\right].
\end{aligned} \tag{18}$$

By Assumption 2 and 3 and summing over i, we obtain

$$\begin{aligned}
&\mathbb{E}\left[F\left(\bar{w}_g^{k+1}, w_l^{k+1}\right)\right] \\
&\leqslant \mathbb{E}\left[F\left(\bar{w}_g^k, w_l^k\right)\right] + \frac{\left(\alpha_k + \eta\right) L^2}{2n}\mathbb{E}\left[\left\|w_g^k - \mathbf{1}\bar{w}_g^k\right\|^2\right] \\
&- \frac{\alpha_k}{2n}\left(\mathbb{E}\left[\left\|G_g\left(\mathbf{1}\bar{w}_g^k, w_l^k\right)\right\|^2\right]\right) - \frac{\eta}{2n}\left(\mathbb{E}\left[\left\|G_l\left(\mathbf{1}\bar{w}_g^k, w_l^k\right)\right\|^2\right]\right) \\
&+ \frac{\alpha_k^2 L - \alpha_k}{2n}\mathbb{E}\left[\left\|G_g\left(\mathbf{1}w_g^k, w_l^k\right)\right\|^2\right] \\
&+ \frac{\eta^2 L - \eta}{2n}\mathbb{E}\left[\left\|G_l\left(w_g^k, w_l^k\right)\right\|^2\right] + \frac{\eta^2 L\sigma_l^2}{2} + \frac{\alpha_k^2 L\sigma_g^2}{2n}.
\end{aligned} \tag{19}$$

Let the step-size satisfy $\eta \leqslant 1/L$, we complete the proof. ■

B. Proof of Theorem 1

Proof: Invoking Lemma 1 and 2, we get

$$\begin{aligned}
&\frac{1}{K}\sum_{t=0}^{K-1}\left(\frac{\alpha_t}{2n}\mathbb{E}\left[\left\|G_g\left(\mathbf{1}\bar{w}_g^k, w_l^k\right)\right\|^2\right] + \frac{\eta}{2n}\mathbb{E}\left[\left\|G_l\left(\mathbf{1}\bar{w}_g^k, w_l^k\right)\right\|^2\right]\right) \\
&\leqslant \frac{F\left(\bar{w}_g^0, w_l^0\right) - F^*}{K} + \frac{\eta^2 L\sigma_l^2}{2} + \frac{\eta^2 E_g L\sigma_g^2}{2n\tau} \\
&+ \frac{\eta L^2}{nK}\sum_{t=0}^{K-1}\mathbb{E}\left[\left\|w_g^t - \mathbf{1}\bar{w}_g^t\right\|^2\right] \\
&\leqslant \frac{F\left(\bar{w}_g^0, w_l^0\right) - F^*}{K} + \frac{\eta^2 L\sigma_l^2}{2} + \frac{\eta^2 E_g L\sigma_g^2}{2n\tau} \\
&+ \eta^3 L^2\left(E_g - 1\right)^2\left(D_g^2 + \sigma_g^2\right).
\end{aligned} \tag{20}$$

Letting T be the number of performing global consensus such that $T\tau \leqslant K \leqslant (T+1)\tau$, we get

$$\begin{aligned}
&\frac{1}{T+1}\sum_{t=0}^{T}\left(\frac{1}{n}\mathbb{E}\left[\left\|G\left(\mathbf{1}\bar{w}_g^{t\tau}, w_l^{t\tau}\right)\right\|^2\right]\right) \\
&\leqslant \frac{2\left(F\left(\bar{w}_g^0, w_l^0\right) - F^*\right)}{T\eta} + \eta\tau L\sigma_l^2 + \frac{\eta E_g L\sigma_g^2}{n} \\
&+ 2\tau\eta^2 L^2\left(E_g - 1\right)^2\left(D_g^2 + \sigma_g^2\right),
\end{aligned} \tag{21}$$

which completes the proof. ■

REFERENCES

[1] G. E. Moore, "Cramming more components onto integrated circuits," vol. 86, no. 1. Ieee, 1998, pp. 82–85.

[2] T. Matsunawa, B. Yu, and D. Z. Pan, "Optical proximity correction with hierarchical bayes model," in *Optical Microlithography XXVIII*, vol. 9426. SPIE, 2015, pp. 238–247.

[3] L. Liebmann, S. Mansfield, G. Han, J. Culp, J. Hibbeler, and R. Tsai, "Reducing dfm to practice: the lithography manufacturability assessor," in *Design and Process Integration for Microelectronic Manufacturing IV*, vol. 6156. SPIE, 2006, pp. 178–189.

[4] H. Yang and et al., "Layout hotspot detection with feature tensor generation and deep biased learning," *IEEE TCAD*, vol. 38, no. 6, pp. 1175–1187, 2018.

[5] W. Wen and et al., "A fuzzy-matching model with grid reduction for lithography hotspot detection," *IEEE TCAD*, vol. 33, no. 11, pp. 1671–1680, 2014.

[6] Y.-T. Yu, G.-H. Lin, I. H.-R. Jiang, and C. Chiang, "Machine-learning-based hotspot detection using topological classification and critical feature extraction," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 3, pp. 460–470, 2015.

[7] M. Shin and J.-H. Lee, "Accurate lithography hotspot detection using deep convolutional neural networks," *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 15, no. 4, p. 043507, 2016.

[8] H. Yang, Y. Lin, B. Yu, and E. F. Young, "Lithography hotspot detection: From shallow to deep learning," in *2017 30th IEEE International System-on-Chip Conference (SOCC)*, 2017, pp. 233–238.

[9] W. Ye, Y. Lin, M. Li, Q. Liu, and D. Z. Pan, "Lithoroc: lithography hotspot detection with explicit roc optimization," in *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, 2019, pp. 292–298.

[10] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang, "A secure federated transfer learning framework," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 70–82, 2020.

[11] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," *Advances in neural information processing systems*, vol. 30, 2017.

[12] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*, 2017, pp. 1126–1135.

[13] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, "Federated learning with personalization layers," *arXiv preprint arXiv:1912.00818*, 2019.

[14] P. P. Liang, T. Liu, L. Ziyin, N. B. Allen, R. P. Auerbach, D. Brent, R. Salakhutdinov, and L.-P. Morency, "Think locally, act globally: Federated learning with local and global representations," *arXiv preprint arXiv:2001.01523*, 2020.

[15] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," *arXiv preprint arXiv:2002.05516*, 2020.

[16] K. Pillutla, K. Malik, A. Mohamed, M. Rabbat, M. Sanjabi, and L. Xiao, "Federated learning with partial model personalization," *arXiv preprint arXiv:2204.03809*, 2022.

[17] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation," *arXiv preprint arXiv:1910.03581*, 2019.

[18] T. Shen, J. Zhang, X. Jia, F. Zhang, G. Huang, P. Zhou, K. Kuang, F. Wu, and C. Wu, "Federated mutual learning," *arXiv preprint arXiv:2006.16765*, 2020.

[19] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

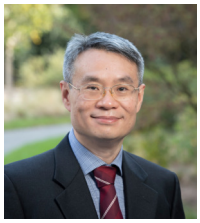
[20] B. McMahan and et al., "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[21] X. He, Y. Deng, S. Zhou, R. Li, Y. Wang, and Y. Guo, "Lithography hotspot detection with fft-based feature extraction and imbalanced learning rate," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 25, no. 2, pp. 1–21, 2019.
[22] W. Ye, M. B. Alawieh, M. Li, Y. Lin, and D. Z. Pan, "Litho-gpa: Gaussian process assurance for lithography hotspot detection," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019, pp. 54–59.
[23] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.
[24] H. Yu, S. Yang, and S. Zhu, "Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 5693–5700, 2019.
[25] A. Paszke and et al., "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019.
[26] J. A. Torres, "Iccad-2012 cad contest in fuzzy pattern matching for physical verification and benchmark suite," in *Proc. ICCAD*, 2012, pp. 349–350.

Jingyu Pan received the B.Eng. degree from Zhejiang University, Hangzhou, China, in 2020. He is currently a Ph.D. student in the Electrical and Computer Engineering department at Duke University. His research interests include machine learning applications in Electronics Design Automation and VLSI circuits and systems.

Xuezhong Lin received the Bachelor's degree from SiChuan University, ChengDu, China, in 2020. He received the Master's degree from Zhejiang University, Hangzhou, China, in 2023. His research interests include federated learning, deep learning, and distributed optimization.

Jinming Xu received the B.S. degree in mechanical engineering from Shandong University, China, in 2009 and the Ph.D. degree in Electrical and Electronic Engineering from Nanyang Technological University (NTU), Singapore, in 2016. He was a research fellow of the EXQUITUS center at NTU from 2016 to 2017; he also received postdoctoral training in the Ira A. Fulton Schools of Engineering, Arizona State University, from 2017 to 2018, and School of Industrial Engineering, Purdue University, from 2018 to 2019, respectively. Currently, he is an assistant professor with the College of Control Science and Engineering at Zhejiang University, China. His research interests include distributed optimization and control, machine learning and network science.



Yiran Chen (M'04-SM'16-F'18) received B.S. (1998) and M.S. (2001) from Tsinghua University and Ph.D. (2005) from Purdue University. After five years in the industry, he joined the University of Pittsburgh in 2010 as Assistant Professor and was promoted to Associate Professor with tenure in 2014, holding Bicentennial Alumni Faculty Fellow. He is now the John Cocke Distinguished Professor of Electrical and Computer Engineering at Duke University and serving as the director of the NSF AI Institute for Edge Computing Leveraging the Next-generation Networks (Athena), the NSF Industry-University Cooperative Research Center (IUCRC) for Alternative Sustainable and Intelligent Computing (ASIC), and the co-director of Duke Center for Computational Evolutionary Intelligence (DCEI). His group focuses on the research of new memory and storage systems, machine learning and neuromorphic computing, and mobile computing systems. Dr. Chen has published 1 book and about 600 technical publications and has been granted 96 US patents. He has served as the associate editor of more than a dozen international academic periodicals and served on the technical and organization committees of about 70 international conferences. He is now serving as the Editor-in-Chief of the IEEE Circuits and Systems Magazine. He received 11 best paper awards, 1 best poster award, and 15 best paper nominations from international conferences and workshops. He received numerous awards for his technical contributions and professional services such as the IEEE CASS Charles A. Desoer Technical Achievement Award, the IEEE Computer Society Edward J. McCluskey Technical Achievement Award, etc. He has been the distinguished lecturer of IEEE CEDA and CAS. He is a Fellow of the AAAS, ACM, and IEEE, and now serves as the chair of ACM SIGDA.

Cheng Zhuo (M'12–SM'16) received the Ph.D. degree from the University of Michigan, Ann Arbor, MI, USA. He is now a full professor in Zhejiang University with research focus on EDA, hardware acceleration, and power/signal integrity. Dr. Zhuo has published over 150 technical papers and received 7 best paper awards and nominations. He also received ACM/SIGDA Technical Leadership Award and Meritorious Service Award, JSPS Invitation Fellowship, and Humboldt Fellowship for Experienced Researchers, etc. Dr. Zhuo has served on the organization and technical program committees of over 30 international conferences and as Associate Editor for IEEE TCAD, ACM TODAES, and Elsevier Integration. He is IEEE CEDA Distinguished Lecturer, a senior member of IEEE and a Fellow of IET.