future internet

Article

Towards a Decentralized Collaborative Framework for Scalable Edge AI

Ahmed M. Abdelmoniem [1,*], Mona Jaber [1], Ali Anwar [2], Yuchao Zhang [3] and Mingliang Gao [4]

[1] School of Electronic Engineering and Computer Science, Queen Mary University of London, London E1 4NS, UK; m.jaber@qmul.ac.uk
[2] Department of Computer Science and Engineering, University of Minnesota Twin-Cities, Minneapolis, MN 55455, USA; aanwar@umn.edu
[3] School of Computer Science, Beijing University of Posts and Telecommunication, Beijing 100876, China; yczhang@bupt.edu.cn
[4] School of Electrical and Electronic Engineering, Shandong University of Technology, Qingdao 255049, China; mlgao@sdut.edu.cn
* Correspondence: ahmed.sayed@qmul.ac.uk

Abstract: Nowadays, Edge Intelligence has seen unprecedented growth in most of our daily life applications. Traditionally, most applications required significant efforts into data collection for data-driven analytics, raising privacy concerns. The proliferation of specialized hardware on sensors, wearable, mobile, and IoT devices has led to the growth of Edge Intelligence, which has become an integral part of the development cycle of most modern applications. However, scalability issues hinder their wide-scale adoption. We aim to focus on these challenges and propose a scalable decentralized edge intelligence framework. Therefore, we analyze and empirically evaluate the challenges of existing methods, and design an architecture that overcomes these challenges. The proposed approach is client-driven and model-centric, allowing models to be shared between entities in a scalable fashion. We conduct experiments over various benchmarks to show that the proposed approach presents an efficient alternative to the existing baseline method, and it can be a viable solution to scale edge intelligence.

Keywords: collaborative computing; distributed systems; data-driven intelligence; machine learning; model distillation

check for updates

Citation: Abdelmoniem, A.M.; Jaber, M.; Anwar, A.; Zhang, Y.; Gao, M. Towards a Decentralized Collaborative Framework for Scalable Edge AI. *Future Internet* **2024**, *16*, 421. https://doi.org/10.3390/fi16110421

Academic Editors: Stefano Rinaldi and Alan Oliveira De Sá

Received: 30 June 2024
Revised: 5 November 2024
Accepted: 12 November 2024
Published: 14 November 2024

1. Introduction

Networks are experiencing massive data traffic increases due to the digital revolution in different industries, smart cities, e-commerce, and social platforms. Furthermore, enormous volumes of data are constantly generated with the growth of Edge-centric applications, e.g., the Internet of Things (IoT), sensors, wearables, and mobile device deployments. Thus, advanced analytics, which can analyze this voluminous data arising in complex scenarios in a decentralized manner, became a fundamental back-end technology of user products and applications and the primary income source for most enterprises [1].

These data-driven analytics rely on the data produced by edge devices for training Artificial Intelligence (AI) and Machine Learning (ML) models [1–4]. Historically, data were gathered and kept in the cloud, where analytics and intelligence are located. However, with the advances and abundance of Edge Intelligent Devices, there has been a growing tendency to keep the data at the source/edge for better security, privacy, and cost-effective data transfer [2,5]. This is especially motivated by recent security and privacy concerns over collecting or storing sensitive user information [6].

To this end, Edge AI has emerged as a paradigm that supports the data-driven applications in IoT environments to train AI/ML models via distributed learning coordinated via a central server or peer-to-peer communications between the distributed clients without clients needing to share their private data [6]. Accordingly, each client trains a local model with their own private data, and leverages available communication channels to securely exchange the model parameters (or updates) to create a common (or global) model. Since AI/ML models perform better with larger training data, it is anticipated that the global model would have better generalization abilities compared to the local model of the clients [7].

Several methods have evolved under the Edge AI paradigm to accomplish decentralized learning objectives, such as 1. Federated Learning (FL) [8], which involves clients that conduct training of local models on their own devices, and then they engage in sending model updates to the server, which aggregates these models to produce a global model and takes care of coordinating the FL rounds; 2. Decentralized Learning (DL) [9], which leverages peer-to-peer coordination for the model exchange among the edge devices; and 3. Transfer Learning (TL) [10], which aims to share the knowledge between different models of the same or different tasks.

These approaches are made possible by harnessing the advances in the AI/ML accelerators embedded in Edge devices [11] and the high-throughput and low-latency 5G and 6G technologies [12,13]. Nonetheless, several obstacles arise, diminishing the efficacy of these technologies and rendering them unsuitable for large-scale decentralized Edge AI. Among the key challenges for these approaches are the lack of ability to maintain high statistical and system efficiency due to the highly heterogeneous devices, configurations, and environments, and the strict synchronization requirements. This results in models with low quality and long training time and hinders the existing approaches from scaling with a large number of learners [11,14]. It should come as no surprise that low-quality models can be quite expensive for various companies and organizations. For example, only recently, the real estate company Zillow wrote down around USD 304M worth of inventory owing to the low accuracy of its Zestimate AI algorithm [15]. Modern data-driven intelligence needs systems that can produce accurate and timely models in a scalable manner [6,11].

We highlight the fundamental issues existing centralized and decentralized approaches face and propose an innovative architectural design for decentralized collaborative learning. In this view, the trained AI/ML models are treated as a commodity that can be exchanged between learning entities to meet global or personal objectives. This view resembles most online delivery services like Uber (passengers) or Deliveroo (food). For example, Uber's task is merely connecting the passengers wanting to be transported from point A to point B with the appropriate drivers. Similarly, the proposed idea facilitates the learning entities' discovery and exchange of the trained models [16]. In this work, we make the following contributions:

- We analyze centralized, federated, and decentralized methods and show their drawbacks.
- We propose a novel design for transforming the learning task into a collaborative knowledge transfer system that allows the learning parties to share (or trade) the trained models based on mutual benefits or needs.
- We present the design of a Model Discovery and Distillation (MDD) service and show its efficacy in improving learning performance. (MDD is open-sourced and available in the following link: https://github.com/ahmedcs/MDD Accessed on 5 November 2024).
- We seize an opportunity to decouple a common model's training task from the models' exchange task. This would facilitate the exchange of models among the best learning parties to improve their model performance collaboratively.

This approach is anticipated to significantly improve AI/ML-based analytics of data generated by various connected devices and sensors that are now an integral part of our daily life applications. We hope this work will be a stepping stone towards a scalable and efficient design for collaborative learning between decentralized entities in the IoT environments.

In the remainder of this work, we give a background on the existing learning methods in Section 3. We then discuss the main limitations of existing paradigms and motivate our proposed design in Section 4. We discuss the proposed architecture design in Section 5. We then present the evaluation results in Section 6. Finally, we conclude this paper in Section 7.

2. Related Work

The field of decentralized learning and model-sharing has explored various approaches and methods to overcome the challenges and limitations of traditional learning paradigms [8,11].

FL is often seen as a machine learning paradigm where a central server distributes the training process to a group of decentralized clients. These clients then train a shared global model using their individual datasets, which are never shared [8,14,17]. The FL method has been utilized to improve the predictive accuracy of virtual keyboards, among other applications [7,18].

Several research papers have focused on Federated Learning (FL), examining its advances, limitations, and the need for new designs and approaches [7,17,19]. Other studies have investigated the impact of heterogeneity on model accuracy and performance in decentralized learning scenarios [14,20]. Additionally, research has been conducted on communication efficiency, privacy preservation, collaboration, and knowledge transfer in decentralized learning systems [10,21–23].

One key approach for transfer learning is knowledge distillation (KD). Beyond model compression, knowledge distillation's applicability extends to scenarios where teacher and student models are pre-trained on different data subsets. This application of knowledge distillation aims to consolidate the distinct knowledge each model possesses, a concept that is increasingly relevant in decentralized collaborative learning [24,25] and Federated Learning [25–27].

The proposed Model Discovery and Distillation (MDD) module in the current research builds upon these works to improve efficiency, scalability, privacy, collaboration, and knowledge transfer among learning entities.

3. Background

We present the various methodologies traditionally used for training AI/ML models on big data generated from large-scale IoT, sensors, and mobile devices. Figure 1 depicts the common four approaches for learning largely decentralized data. We discuss them as follows:

(a) Centralized Learning (CL)—Figure 1a: In most cases, the end devices continually feed the generated data into the servers running in the cloud, where they are kept to be processed and analyzed later. These data are evaluated to identify properties that will assist in training AI and ML models using the cloud. High-end servers are used to train these models in the cloud providers' data centers. ML-as-a-service is becoming popular for training models on large amounts of data at scale provided by various cloud providers, such as Google Cloud, Azure, and AWS. Unfortunately, centralized learning (CL) is seeing weaker adoption due to the gathering of user data, which poses serious threats to users' personal information and privacy [28,29]. Further, the scalability is impeded due to high communication costs associated with data transfer [7,23]. Even though various works have aimed at reducing the communication overhead in the context of distributed data-parallel learning [30], various challenges remain unsolved [11].

(b) Federated Learning (FL)—Figure 1b: In the FL paradigm, the clients are users with end devices (e.g., mobile, IoT, or sensors) using apps generating the data. The client devices maintain the data and never share it with others. So, the clients train local models and engage with the FL server to create a common model. The server assists the clients with the task of aggregating the clients' models and coordinating the FL rounds [6–8]. The training procedure consists of a series of rounds managed by the server until a specific goal is achieved (e.g., target accuracy). In each cycle, the server selects a small set of clients for participating in the FL round who perform training of the model on their local data and share their new models to create a new global model. The server plays a crucial task in performing this aggregation step. However, this synchronous update step results in issues related to synchronization, dependability, and communication costs [11]. When a training round begins, available clients log in and express their desire to participate with the server. Typically, a client is available if it is fully connected (through IEEE 802.11/WiFi network, for instance), has a power source, and is idle [7]. Some recent works have explored battery-powered scenarios [20].

The server then executes a selection algorithm to choose a subset of the large client population. Then, the server sends the selected clients the FL task (i.e., the ML model and hyper-parameter configurations). All the clients run the same number of optimization steps determined by the FL task configuration. Then, the clients upload their new models to the server. The FL server then combines the client models to generate a new global model, which is check-pointed to the server's storage [7]. However, because of the non-IID distribution of the data, FL tends to be less efficient than CL due to the risk of divergence and lower model accuracy and the role of malicious clients aiming to diverge the model [28].

(c) Decentralized Learning (DL)—Figure 1c: DL is an alternative way to build generic models based on decentralized data in edge-device environments [9]. In DL, the learners use peer-to-peer communication to coordinate their training of a model tailored to their shared tasks [23]. Thus, device groups can train a shared model while maintaining the data of individual devices. The devices must always be present to repeat the training procedure in lockstep, and stragglers slow down the training because there is no central coordination. As a result, scalability and efficiency are impeded, since devices cannot train independently of slow learners [11].

(d) Transfer Learning (TL)—Figure 1d: Transfer Learning is an emerging ML method that transfers information from one domain to another. TL relies on previously learned information that ML algorithms can re-purpose, according to [31]. Additionally, TL supports using IID data when training data are insufficient [31]. And TL can maintain efficiency because it no longer relies on IID data availability. However, in TL, scalability is hampered as it is not easy to find and choose which model to use for the transfer as there is a large pool of models to choose from [10,22]. It is also challenging to perform scalable and secure TL as it requires active participation over secure channels between the two parties and their data during transfer learning (especially in IoT scenarios where data involves sensitive information) [6,31].

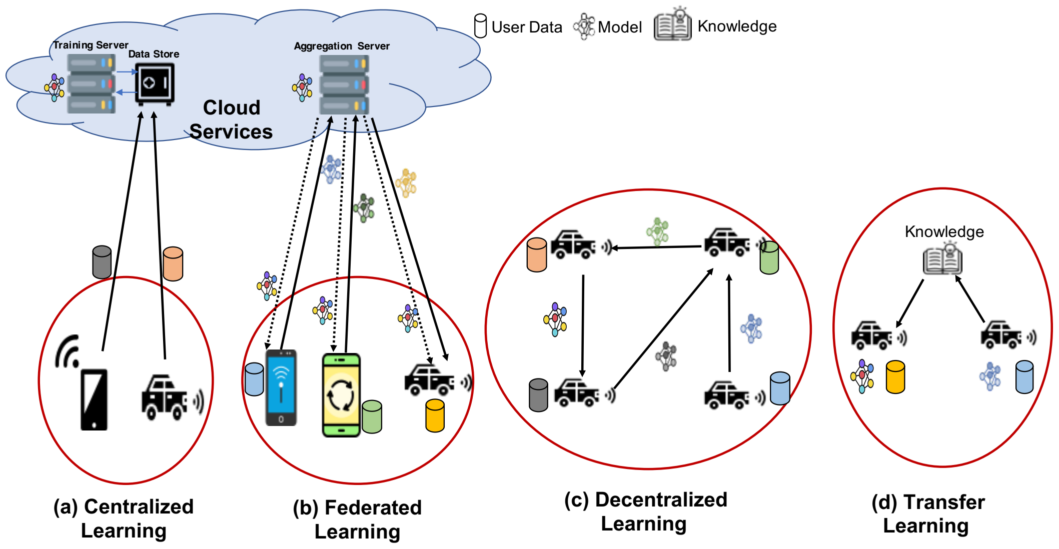


Figure 1. Comparison of the existing traditional learning paradigms. (**a**) Centralized Learning involves data collection and then training. (**b**) Federated Learning involves clients training local models and aggregating them to a global model via a server. (**c**) Decentralized Learning involves clients collaboratively training on a common model via peer-to-peer exchange and coordination. (**d**) Transfer Learning involves knowledge within models being used to train other models via distillation.

4. Challenges and Motivation

We discussed traditional learning paradigms in the previous section, and analyzed their primary contrasts. Now, we focus on the significant difficulties presented by these paradigms, which lead to inefficient learning, limited scalability, or privacy concerns [6,11,28].

In Table 1, we qualitatively compare the existing approaches over efficiency, scalability, and privacy metrics. Efficiency refers to the ability of a mechanism to achieve good accuracy (or convergence) in a short time. Scalability refers to the ability of a scheme to scale the learning process over a larger number of clients with minimal costs. Privacy refers to the ability of a method to maintain or help enforce the privacy preservation of the client's data. Regarding efficiency, methods with greater control over data distribution (e.g., centralized) would result in model convergence in a short time [19]. On the other hand, methods that can coordinate the operations for many participants (e.g., federated) would result in better abilities for scale [21]. Regarding privacy, methods that keep the user's data private without sharing it (e.g., decentralized or federated) would result in better privacy guarantees [2].

Table 1. Comparison of existing approaches.

Method	Efficiency	Scalability	Privacy
Centralized Learning	High	Medium	Low
Federated Learning	Low	Medium	High
Decentralized Learning	Low	Low	High
Transfer Learning	Medium	Low	Medium

While current centralized methods are the most efficient (due to the lack of non-IID data distribution problems), they are not desirable due to the non-existent or low privacy guarantees. The decentralized and federated methods overcome this issue by providing high privacy guarantees (due to the non-sharing of the data), but they fail to provide satisfactory convergence results quickly. Transfer Learning provides a good comprise, but it typically fails to scale well with large scenarios.

Unfortunately, the current paradigms, such as federated, decentralized, and transfer learning, are not efficient or scalable enough to meet the growing demand and adoption of decentralized services and applications [11]. One of the detrimental challenges that the face of the traditional paradigms, which hinders efficiency and scalability, is the heterogeneity of the users and learning environment [6,7,11]. When training on decentralized parties, the common types of heterogeneity can be one or a combination of:

- **Data**: non-uniform data points of the learners in number, type, and distribution.
- **Device**: a diverse set of device hardware configurations and communication network settings.
- **Behavioral**: dynamic participation availability driven by learners' behavior.

We find that heterogeneity, regardless of type, can significantly impact the model quality (as well as fairness), and cause model divergence in the worst case [14]. In Section 6, we support this with experiments that cover many scenarios on five different benchmarks (i.e., models and datasets) and large populations of thousands of users [14].

Since heterogeneity is endemic to decentralized approaches, mitigating its impact on the performance of decentralized methods is desirable. Hence, we identify a pressing need to manage, process and analyze the increasing amount of decentralized data generated by collaborative parties to support the growth of ML and data-driven applications [7,9,11,13]. We seize an opportunity from the proliferation of connected devices and edge computing, which create abundant data at the network's edge. Regardless, there is a lack of frameworks that can do so efficiently, at a low cost, and with fast training time, while observing privacy requirements. Without questions, any centralized approach that collects user data becomes infeasible. Nowadays, it is widely accepted that data are processed locally on edge devices, and models will be trained using collaborative approaches to decentralized data. This will be the key driver for high-quality, personalized, and intelligent services for a customized user experience, including personalized healthcare, gaming, and recommendation services [1,11,13]. These forms of decentralized learning must protect data privacy and efficiently use the intelligence and computational resources available at the edge.

Furthermore, practical deployment aspects such as heterogeneity, communication costs, coordination, synchronization, and availability to participate in training are all factors that either cause a hindrance to the scalability, slow down the FL rounds, or impact the overall model performance (i.e., generalization) [11,14,17]. As a side effect, when training is not sufficiently fast, the models can not adapt to localized shifts in data trends [11]. Therefore, addressing the above concerns guided our design of the following architecture.

5. The Proposed Architecture

This work addresses the following central question: **What would be the architectural design for allowing scalable collaborative learning over distributed data?** We position this work as a call for architectural designs that can:

- Efficiently enable various types of decentralized training between learning parties;
- Securely store a representation of the models in vaults, which ensure only authorized access to clients' models;
- Develop scalable model discovery methods by providing a distributed service to resolve clients' model requests by finding the best models matching their search criteria.

Next, we will discuss a potential design for a decentralized architecture that aims to capture the essence of the aforementioned goals and objectives.

The proposed architecture is presented in Figure 2. The system introduces a Model Discovery and Distillation (MDD) service that aids any party with the network in requesting and attaining knowledge within other models shared with the network. The service helps discover the requested model meeting the requirements and the distillation process with servers suited at the edge. Specifically, these servers and the target client of distillation can arrange for the distillation of the discovered knowledge to the target client's model. This process can be any form of knowledge transfer learning method [10], such as knowledge distillation [32], domain adaptation [33], mutual learning [34] fine-tuning, and personalization [35]. This process could take a few rounds of knowledge transfer until the target client's objective is met (i.e., target accuracy is achieved).

The advantage of the proposed architecture is that it has no single point of control and failure (like FL), no explicit synchronization among devices (like DL), and no expensive movement of private data (like CL). This is because the MDD process is client-driven, where learners create an initial model with their local datasets and then asynchronously seek models in the network to boost the model quality via distillation. The system discovers the model for the requester without involving other learners, which helps mitigate any influence of the clients' heterogeneity. The proposed architecture may also introduce incentive mechanisms (e.g., based on monetary income or mutual interest [22]) to enable the sharing of high-quality models in the network. There are various ways to realize incentives in the MDD architecture to encourage knowledge dissemination and sharing of high-quality models [36]. For instance, deep reinforcement learning (DRL) agents could be leveraged to learn the optimal pricing strategy the MDD service uses, incentivizing the clients towards model training [37]. Similarly, clients can also share the utility of each shared model they receive to guide the system for distributing incentives [38]. The utility is calculated as the percentage of average accuracy improvement over the model trained solely on the client's data.

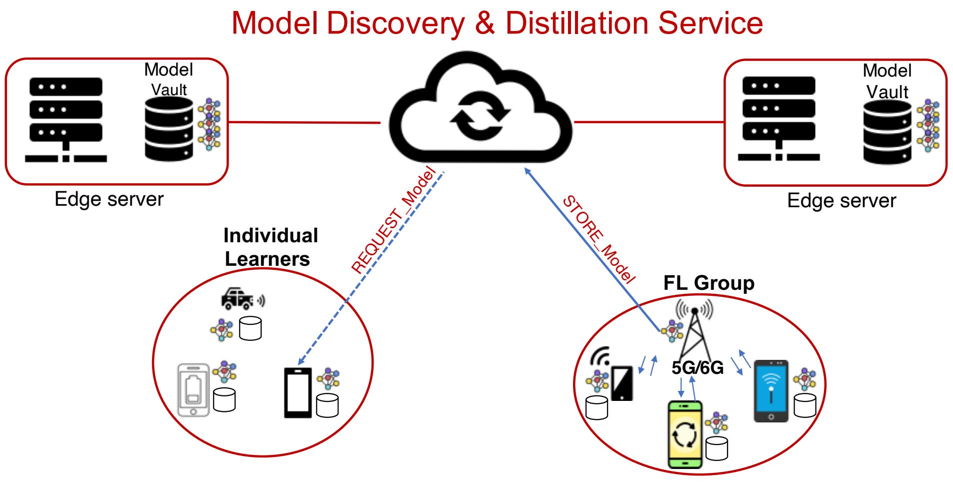


Figure 2. The proposed architectural design involves decentralized learning parties, secure model vaults to store the models hosted by edge servers and discovery service for model exchange hosted in the cloud.

In the proposed architecture, as shown in Figure 2, the data owners train an initial model individually or in groups (e.g., via Federated Learning). Then, the learner(s) request to store the model in private and secure model stores (or vaults). The system will evaluate the model either on a public dataset by the service or via requesting testing parties to obtain the quality metrics of the model. Whenever they require improvements on their model, the learning parties request a trained model to the discovery service specifying certain qualities of the requested model (e.g., a classifier needs to improve its accuracy to reach at least 90% of accuracy for class D). This requires novel discovery algorithms to find the best models in the network matching the requested knowledge. We provide the details of the algorithm supporting the proposed architecture in Algorithm 1.

Algorithm 1: Model Discovery and Distillation (MDD)

```
1  Function Request Model (r)
2      if $r \in Search(Knowledge\_Vaults)$ then
3          //Retrieve the model from the matching knowledge vault;
4          $M_{knowledge} = Knowledge\_Vaults(r)$;
5      else
6          //Search for a matching model in global population;
7          if $r \in Search(Global\_Learners)$ then
8              $M_{knowledge} = Global\_Learners(r)$;
9      //Perform Knowledge Distillation;
10     $M_{new} = Distill(M_{Knowledge}, M_{learner})$;
11 Function Store Model (M)
12     $r = Extract\_representation(M)$;
13     $V = Find\_vault(r)$;
14     $Store\_vault(V, r, M)$;
```

Therefore, the key innovation in this architecture is the introduction of discovery service design based on distributed architecture. This distributed service will act as the arbiter of clients' requests, leveraging decentralized edge servers for the discovery process. In particular, we envision the MDD service as a distributed service leveraging cloud resources or serverless computing to handle client requests [4]. The computing service that handles the knowledge discovery would rely on a distributed ledger (e.g., name or directory service) to keep track of the knowledge produced in the network. The MDD service relies on the unused resources of the edge servers to aid with knowledge distillation for updating clients' models [5].

However, achieving scalable and efficient discovery requires investigating novel discovery algorithms and protocols to search the network for the models that fulfill the requirements of clients' requests (the exploration of scalable discovery algorithms is part of our future work). The discovery service works by employing a service similar to Named Data Networking (NDN) [39] used for finding content in content-distribution networks (CDN) or Information-Centric Networks [40]. Similar to NDN, the models are treated as content that requires retrieval, and they are addressed using their key features, such as the task they solve, the dataset they were trained on, etc. Moreover, other performance metrics are collected about the model. Then, this information is stored in the distributed ledger (or directory service) and used to forward client requests to the destination (i.e., the model owner). Moreover, optimal routing algorithms can handle the dynamic and mobile IoT environments [41].

Upon completion of discovery, the requester either obtains a copy of the model and applies knowledge transfer technique (e.g., model distillation) [10,32] to integrate the knowledge of the discovered model(s) into its model and enhance the model quality (e.g., classification accuracy for class D), or the client can share its model with an Edge or Cloud server to allow for the knowledge distillation to occur on the server. In our design, we envision that the clients can seek full or partial assistance from edge server(s) to conduct the knowledge transfer operations to integrate the knowledge into their own models. This can be proven useful, especially in IoT use cases where the clients are typically limited in computational resources. In the following, we conduct experiments aiming at showcasing (1) the heterogeneity's detrimental impact on the collaboratively produced models, and (2) the effectiveness of the new system in improving model performances via model discovery and distillation.

6. Experimental Evaluation

In this section, we conduct preliminary experiments to evaluate the proposed MDD service. To this end, we design simulation experiments: (1) to showcase the impact of heterogeneity on the model performance as presented in Section 6.2; and (2) to compare the proposed MDD approach with the existing baseline methods in terms of model performance using several benchmarks, such as Logistic Regression with Synthetic dataset, CCN with FEMINIST dataset, and RNN with Reddit dataset, as presented in Section 6.3.

6.1. Experimental Setup

We leverage the FLASH framework [42], which covers FL benchmarks of five different benchmarks involving tasks such as logistic regression, image classification, and natural language processing. These tasks are reminiscent of common AI/ML tasks in scenarios involving federated IoT data [6,11]. FLASH provides the necessary infrastructure to simulate traditional FL, and defines a set of hyper-parameters for controlling the experiments. The clients' device hardware configurations regarding computation are based on realistic device profiles [42]. Moreover, a real-world user trace reflects real availability dynamics during the training process. The datasets involve various tasks, such as the Logistic Regression model trained on the Synthetic Dataset, the CNN model trained on the FEMINIST dataset, and the RNN model trained on the Reddit dataset. The data are distributed non-IID among the clients, reflecting the real-world data distributions among independent data owners or producers in typical mobile and IoT scenarios (for the source code and datasets, we refer the reader to our public repository: https://github.com/ahmedcs/MDD Accessed on 5 November 2024).

As performed in prior works [8,14,42], the training and test data are partitioned to make unique partitions of the data assigned to each client. The clients are sampled to use the allocated training and test partition during training and test rounds, respectively. We instrument the framework with information logging and configurations of the generated scenarios to be run on the server's GPU. For more details, we refer interested readers to [14,42].

6.2. Heterogeneity Impact

As discussed previously, one of the main problems in decentralized IoT scenarios is the device, configurations, and environment heterogeneity [7,11,14]. We extensively run experiments in FL settings covering over one thousand five hundred different heterogeneity settings, FL configurations, and learning hyper-parameters. The average model accuracy is evaluated by invoking a test round after every 10 training rounds. During test rounds, the model accuracy is evaluated by every client using its own test dataset, and then these accuracy figures are averaged to produce the average test accuracy of the model. Figure 3 shows a bar plot for the achieved test accuracy from all the experiments (normalized to the maximum baseline accuracy for each of the five benchmarks) and contrasts the following cases: (i) **U**: uniform or homogeneous situations in which all clients have identical hardware and network settings and are constantly accessible. (ii) **BH**: a behavior heterogeneity case where the devices have variable availability patterns based on real-world trace; (iii) **DH**: a device heterogeneity case where each device is assigned a different compute and communication profile from real-world trace; and (iv) **H**: a mixed heterogeneity case with both device and behavior heterogeneity.

As Figure 3 shows, all the heterogeneity cases (**BH**, **DH**, and **H**) significantly impact the test accuracy compared to the homogeneous case (**U**). The results show that the median accuracy (the light green line) is lowered by up to 12%, and the variance is severely high in the heterogeneous cases. What makes matters worse is that heterogeneity can prevent training from converging. This reinforces the motivation of this work, that heterogeneity is endemic to decentralized learning and to scale the training should cope with heterogeneity to produce high-quality models (c.f. [14] for more detailed results).

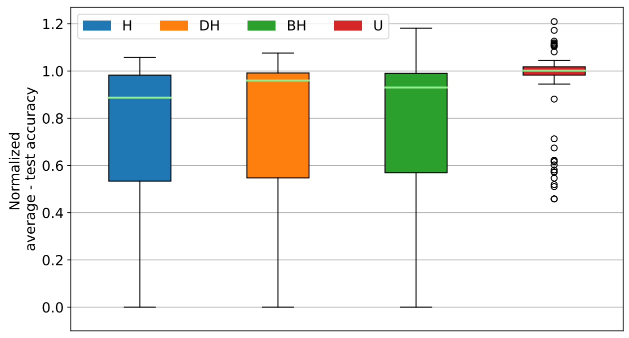


Figure 3. The quality impact of heterogeneity. We find that the values are zero in heterogeneous cases when the models diverge. The values are greater than one when an experiment results in better model accuracy than the baseline with the default configurations.

6.3. Efficacy of the Proposed Approach

We extend the FLASH framework with the MDD module to aid the clients in discovering and distilling the models. We also use FL as the most common baseline for decentralized learning methods. To have a fair comparison, we run the training for FL in an ideal scenario to produce its best results. To this end, we use the uniform (or homogeneous) setting (U) for Federated Learning in which devices have the same hardware configurations and are available throughout the training. We aim to demonstrate that different clients in the network can learn within their own groups at their own pace (i.e., FL group vs. individual clients' group, as in Figure 2). Then, the MDD service will prove to be effective if discovering and distilling the models for the individual clients or groups via the MDD service results in improvements of the models produced by other parties. In this case, the individual clients request the FL groups' model and use it as their initial model before training.

Experimental setting: Similarly to Figure 2, the experiments involve a small group of 10 independent parties individually conducting local training on their own data (**IND**) for a variable number of epochs and a large group consisting of the remaining clients training a global model via FL for 500 rounds (**FL**). We contrast the model quality of these two groups with the case when the proposed architecture is used (**MDD**). The 10 individual learners will request a global model trained by the FL group from the discovery service, and then each of the 10 learners will distill the FL model with their own model. The result plots show the test accuracy averaged by testing a model on the 10 individual clients (**IND**). The x-axis shows the local epochs used for training the model using the isolated 10 individual clients' datasets. Therefore, this number would be 0 in the FL case.

Traditional AI models: First, we conduct the experiments using 10 clients in an individual group and 9990 clients in an FL group to train using logistic regression as the model on synthetic dataset distributed in non-IID fashion over all the clients in both groups. As shown in Figure 4, training the individual clients for a few epochs attains high average accuracy (**IND**) due to the simplicity of the LR model. The FL model (**FL**) shows lower accuracy when tested on the test datasets of the 10 individual parties because of the large population size, which confirms with observations about FL in non-IID settings [19]. However, when the 10 clients request and distill the FL model via (**MDD**), their average test accuracy improves by 12 accuracy points.

Deep Neural Network (DNN) models: Next, we leverage DNN models to conduct experiments in a device heterogeneous setting. Specifically, both Recurrent Neural Network (RNN) and Convolution Neural Network (CNN) models are trained by over 10 clients who learn individually, and 3990 and 803 clients belong to the FL group. The FEMNIST and REDDIT datasets, used to train the CNN and RNN models, respectively, are distributed in a non-IID fashion among all the clients in both groups.

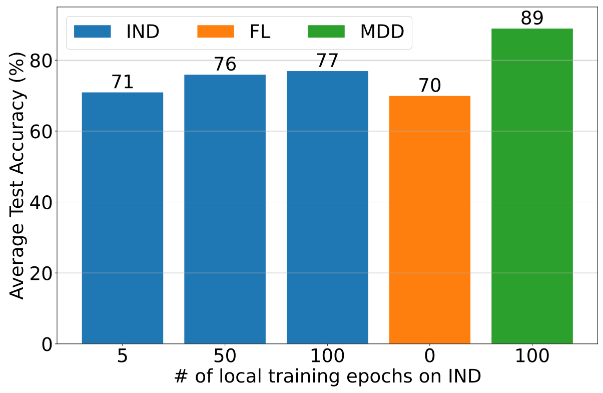
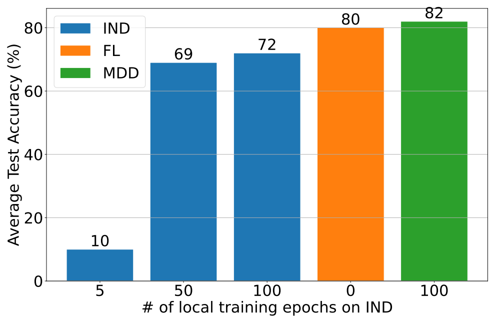


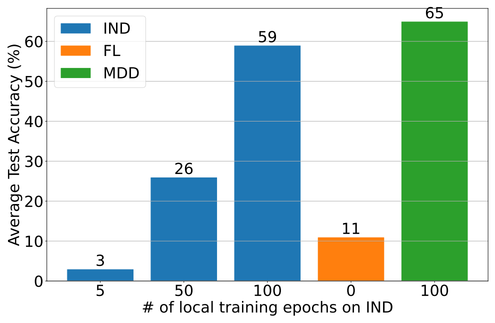
Figure 4. Performance comparison in the LR-Synthetic scenario.

Next, in Figure 5a, we observe that training for the model locally for 100 epochs does improve the model for the individual parties (**IND**). The FL model (**FL**) shows good accuracy when tested on the 10 individual clients. This might be due to the CNN models' ability to generalize well on samples of the Feminist dataset (which consists of easy-to-classify image classes of digits and letters). Moreover, the model is trained on many samples from 3390 clients in the FL group. We note then that if the individual parties (**IND**) request the FL model and distill their local models with the FL model, then the result is their local models improve in accuracy by 10 accuracy points (i.e., **MDD**).

Next, we observe in Figure 5b that in the RNN-Reddit case, the model trained by the FL group (**FL**) on its own shows very low accuracy values on the test set of the clients. In contrast, the model trained on the individual clients (**IND**) improves in accuracy over the training epochs. This might be due to the difficulty of text-based datasets such as Reddit, which is difficult to generalize well, especially with a small client population. Therefore, the FL model could not perform well when tested on the **IND** group, which confirms observations about FL in non-IID settings [19]. Further, we observe improvement in the individual clients' model accuracy by 6 accuracy points, which are achieved when they discover and distill the FL model with their own local model (i.e., the **MDD** approach).



(**a**) CNN-FEMNIST.



(**b**) RNN-REDDIT.

Figure 5. Performance comparison in scenarios involving DNNs.

In summary, the results obtained using a variety of benchmarks and case studies demonstrate that the developed MDD approach is effective. However, there are technical limitations and research gaps that require further attention:

- **Discovery mechanism:** The effectiveness of the MDD approach largely depends on the efficacy of the discovery protocol. In this work, we have relied on assumptions of the availability of discovery information. However, we have not explored the overhead of the discovery method and how to implement such a protocol at the Internet scale. We believe this can be overcome by borrowing approaches from content (or information)-centric network [40].
- **Distillation method:** Similarly, the effectiveness of the MDD approach largely depends on the efficiency of the distillation methods. In this work, we have used a simple model fusion based on average to achieve knowledge distillation. However, we have not explored the effectiveness of various distillation methods and their computational overhead. We believe there is a recent line of work on achieving efficient knowledge distillation between pre-trained models [43].

With this work, we set the first steps toward additional research into some technical areas that are required for this strategy to become practically viable.

7. Conclusions

This paper examines the challenges of main decentralized learning paradigms, and demonstrates how they fail to facilitate effective collaboration in complex situations, such as IoT use cases.

Our contribution in this work is that we underline the need for the research community to produce new designs based on model sharing. This promotes the services for identifying models owned by various network participants to support large-scale, decentralized learning collaboration. This paper uses several benchmarks and case studies to demonstrate the benefit of the established MDD strategy.

We believe the current work has some limitations, specifically in studying optimal discovery and distillation methods to achieve an effective and scalable MDD system. The methods used in this work are trivial and are meant to demonstrate the potential of the proposed approach; however, the methods for discovery and distillation require extensive evaluation.

For future research, we recommend conducting more research into the technical, practical, and efficient implementation of discovery and distillation techniques. Specifically, we believe for discovery methods, there are extensive lines of work related to content-centric network that could be leveraged. For instance, several Internet content delivery methods have been proposed and studied. On the distillation approaches, there is recent interest in efficient knowledge distillation in decentralized environments. These methods could be incorporated into the MDD approach, but their overhead must be analyzed. In our ongoing research, we plan to develop protocols that enable scalable and secure model storage, discovery, and transfer to achieve the objectives of the proposed architecture.

Author Contributions: Conceptualization, methodology, software, formal analysis, investigation, visualization, project administration, and writing—original draft preparation, A.M.A.; validation and writing—review and editing, A.M.A., M.J., A.A., Y.Z. and M.G.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The dataset used in this study is publicly available via the following LEAF framework: LEAF is a benchmarking framework for learning in federated settings, with applications including Federated Learning, multi-task learning, meta-learning, and on-device learning found in this link https://leaf.cmu.edu/ (Accessed on 5 November 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Xu, D.; Li, T.; Li, Y.; Su, X.; Tarkoma, S.; Jiang, T.; Crowcroft, J.; Hui, P. Edge Intelligence: Empowering Intelligence to the Edge of Network. *Proc. IEEE* **2021**, *109*, 1778–1837. [CrossRef]
2. Singh, R.; Gill, S.S. Edge AI: A survey. *Internet Things Cyber-Phys. Syst.* **2023**, *3*, 71–92. [CrossRef]
3. Ahmed, I.; Jeon, G.; Chehri, A. A Smart IoT Enabled End-to-End 3D Object Detection System for Autonomous Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 13078–13087. [CrossRef]
4. Golec, M.; Gill, S.S.; Parlikad, A.K.; Uhlig, S. HealthFaaS: AI based Smart Healthcare System for Heart Patients using Serverless Computing. *IEEE Internet Things J.* **2023**, *10*, 18469–18476. [CrossRef]
5. Iftikhar, S.; Gill, S.S.; Song, C.; Xu, M.; Aslanpour, M.S.; Toosi, A.N.; Du, J.; Wu, H.; Ghosh, S.; Chowdhury, D.; et al. AI-based fog and edge computing: A systematic review, taxonomy and future directions. *Internet Things* **2023**, *21*, 100674. [CrossRef]
6. Zhang, T.; Gao, L.; He, C.; Zhang, M.; Krishnamachari, B.; Avestimehr, A.S. Federated Learning for the Internet of Things: Applications, Challenges, and Opportunities. *IEEE Internet Things Mag.* **2022**, *5*, 24–29. [CrossRef]
7. Bonawitz, K.; Eichner, H.; Grieskamp, W.; Huba, D.; Ingerman, A.; Ivanov, V.; Kiddon, C.; Konečný, J.; Mazzocchi, S.; McMahan, H.B.; et al. Towards Federated Learning at Scale: System Design. In Proceedings of the SysML Conference, Stanford, CA, USA, 31 March–2 April 2019.
8. McMahan, H.B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), Lauderdale, FL, USA, 20–22 April 2017.
9. Daga, H.; Nicholson, P.K.; Gavrilovska, A.; Lugones, D. Cartel: A System for Collaborative Transfer Learning at the Edge. In Proceedings of the ACM Symposium on Cloud Computing (SoCC), Santa Cruz, CA, USA, 20–23 November 2019.
10. Renggli, C.; Pinto, A.S.; Rimanic, L.; Puigcerver, J.; Riquelme, C.; Zhang, C.; Lucic, M. Which Model to Transfer? Finding the Needle in the Growing Haystack. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022.
11. Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Nitin, A.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. Advances and Open Problems in Federated Learning. *Found. Trends Mach. Learn.* **2021**, *14*, 1–210. [CrossRef]
12. Jaber, M.; Imran, M.A.; Tafazolli, R.; Tukmanov, A. 5G Backhaul Challenges and Emerging Research Directions: A Survey. *IEEE Access* **2016**, *4*, 1743–1766. [CrossRef]
13. Letaief, K.B.; Chen, W.; Shi, Y.; Zhang, J.; Zhang, Y.A. The Roadmap to 6G: AI Empowered Wireless Networks. *IEEE Commun. Mag.* **2019**, *57*, 84–90. [CrossRef]
14. Abdelmoniem, A.M.; Ho, C.Y.; Papageorgiou, P.; Canini, M. A Comprehensive Empirical Study of Heterogeneity in Federated Learning. *IEEE Internet Things J.* **2023**, *10*, 14071–14083. [CrossRef]
15. Metz, R. Zillow's Home-Buying Debacle Shows How Hard It Is to Use AI to Value Real Estate, 2021. Available online: https://edition.cnn.com/2021/11/09/tech/zillow-ibuying-home-zestimate/index.html (accessed on 5 November 2024).
16. Lu, X.; Zhang, Y.; Zou, H.; Liang, Y.; Wang, W.; Crowcroft, J. FedCOM: Efficient Personalized Federated Learning by Finding Your Best Peers. In Proceedings of the 2nd ACM Workshop on Data Privacy and Federated Learning Technologies for Mobile Edge Network, Madrid, Spain, 6 October 2023.
17. Abdelmoniem, A.M.; Sahu, A.N.; Canini, M.; Fahmy, S.A. REFL: Resource-Efficient Federated Learning. In Proceedings of the ACM EuroSys, Rome, Italy, 8–12 May 2023; pp. 215–232.
18. Yang, T.; Andrew, G.; Eichner, H.; Sun, H.; Li, W.; Kong, N.; Ramage, D.; Beaufays, F. Applied Federated Learning: Improving Google Keyboard Query Suggestions. *arXiv* **2018**, arXiv:1812.02903.

19. Zhu, H.; Xu, J.; Liu, S.; Jin, Y. Federated learning on non-IID data: A survey. *Neurocomputing* **2021**, *465*, 371–390. [CrossRef]
20. Arouj, A.; Abdelmoniem, A.M. Towards Energy-Aware Federated Learning on Battery-Powered Clients. In Proceedings of the ACM FedEdge Workshop-ACM MobiCom, Sydney, Australia, 17 October 2022.
21. Hu, C.; Jiang, J.; Wang, Z. Decentralized Federated Learning: A Segmented Gossip Approach. *arXiv* **2019**, arXiv:1908.07782.
22. Cui, S.; Liang, J.; Pan, W.; Chen, K.; Zhang, C.; Wang, F. Collaboration Equilibrium in Federated Learning. In Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, 14–18 August 2022.
23. Pappas, C.; Papadopoulos, D.; Chatzopoulos, D.; Panagou, E.; Lalis, S.; Vavalis, M. Towards Efficient Decentralized Federated Learning. In Proceedings of the IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW), Bologna, Italy, 10 July 2022; pp. 79–85. [CrossRef]
24. Sattler, F.; Marban, A.; Rischke, R.; Samek, W. CFD: Communication-Efficient Federated Distillation via Soft-Label Quantization and Delta Coding. *IEEE Trans. Netw. Sci. Eng.* **2021**, *9*, 2025–2038. [CrossRef]
25. Roth, K.; Thede, L.; Koepke, A.S.; Vinyals, O.; Hénaff, O.; Akata, Z. Fantastic Gains and Where to Find Them: On the Existence and Prospect of General Knowledge Transfer between Any Pretrained Model. *arXiv* **2023**, arXiv:2310.17653.
26. Kalra, S.; Wen, J.; Cresswell, J.C.; Volkovs, M.; Tizhoosh, H.R. Decentralized federated learning through proxy model sharing. *Nat. Commun.* **2023**, *14*, 2899. [CrossRef]
27. Itahara, S.; Nishio, T.; Koda, Y.; Morikura, M.; Yamamoto, K. Distillation-Based Semi-Supervised Federated Learning for Communication-Efficient Collaborative Training With Non-IID Private Data. *IEEE Trans. Mob. Comput.* **2021**, *22*, 191–205. [CrossRef]
28. Kumar, A.; Khimani, V.; Chatzopoulos, D.; Hui, P. FedClean: A Defense Mechanism against Parameter Poisoning Attacks in Federated Learning. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 23–27 May 2022.
29. Liu, Y.; Kang, Y.; Xing, C.; Chen, T.; Yang, Q. A Secure Federated Transfer Learning Framework. *IEEE Intell. Syst.* **2020**, *35*, 70–82. [CrossRef]
30. Abdelmoniem, A.M.; Canini, M. DC2: Delay-aware Compression Control for Distributed Machine Learning. In Proceedings of the IEEE INFOCOM, Vancouver, BC, Canada, 10–13 May 2021.
31. Pan, S.J.; Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]
32. Lan, X.; Zhu, X.; Gong, S. Knowledge Distillation by On-the-Fly Native Ensemble. In Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), Montreal, QC, Canada, 3–8 December 2018.
33. Gan, S.; Mathur, A.; Isopoussu, A.; Kawsar, F.; Berthouze, N.; Lane, N.D. FRuDA: Framework for Distributed Adversarial Domain Adaptation. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 3153–3164. [CrossRef]
34. Zhang, Y.; Xiang, T.; Hospedales, T.M.; Lu, H. Deep Mutual Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4320–4328.
35. Guo, Y.; Shi, H.; Kumar, A.; Grauman, K.; Rosing, T.; Feris, R. SpotTune: Transfer Learning Through Adaptive Fine-Tuning. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 4800–4809.
36. Zhan, Y.; Zhang, J.; Hong, Z.; Wu, L.; Li, P.; Guo, S. A Survey of Incentive Mechanism Design for Federated Learning. *IEEE Trans. Emerg. Top. Comput.* **2022**, *10*, 1035–1044. [CrossRef]
37. Zhan, Y.; Li, P.; Qu, Z.; Zeng, D.; Guo, S. A Learning-Based Incentive Mechanism for Federated Learning. *IEEE Internet Things J.* **2020**, *7*, 6360–6368. [CrossRef]
38. Wang, X.; Le, Q.; Khan, A.F.; Ding, J.; Anwar, A. A Framework for Incentivized Collaborative Learning. *arXiv* **2023**, arXiv:2305.17052.
39. Afanasyev, A.; Burke, J.; Refaei, T.; Wang, L.; Zhang, B.; Zhang, L. A Brief Introduction to Named Data Networking. In Proceedings of the IEEE Military Communications Conference (MILCOM), Los Angeles, CA, USA, 29–31 October 2018; pp. 1–6.
40. Ascigil, O.; Sourlas, V.; Psaras, I.; Pavlou, G. A Native Content Discovery Mechanism for the Information-Centric Networks. In Proceedings of the 4th ACM Conference on Information-Centric Networking, Paris, France, 26–28 September 2017; pp. 145–155.
41. Cong, P.; Zhang, Y.; Liu, Z.; Baker, T.; Tawfik, H.; Wang, W.; Xu, K.; Li, R.; Li, F. A deep reinforcement learning-based multi-optimality routing scheme for dynamic IoT networks. *Comput. Netw.* **2021**, *192*, 108057. [CrossRef]
42. Yang, C.; Wang, Q.; Xu, M.; Chen, Z.; Bian, K.; Liu, Y.; Liu, X. Characterizing Impacts of Heterogeneity in Federated Learning upon Large-Scale Smartphone Data. In Proceedings of the Web Conference, Ljubljana, Slovenia, 19–23 April 2021.
43. Alballa, N.; Canini, M. Practical Insights into Knowledge Distillation for Pre-Trained Models. *arXiv* **2024**, arXiv:2402.14922.