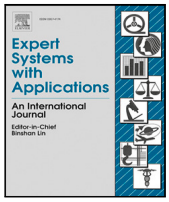
Expert Systems With Applications 242 (2024) 122847

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



# Model pruning enables localized and efficient federated learning for yield forecasting and data sharing

Andy Li [a], Milan Markovic [a,b], Peter Edwards [a], Georgios Leontidis [a,b,*]

[a] School of Natural and Computing Sciences, University of Aberdeen, Aberdeen, AB24 3UE, UK
[b] Interdisciplinary Centre for Data and AI, University of Aberdeen, Aberdeen, AB24 3FX, UK

ARTICLE INFO

*Keywords:*
Agri-food
Yield prediction
Neural network pruning
Federated learning

ABSTRACT

Federated Learning (FL) presents a decentralized approach to model training in the agri-food sector and offers the potential for improved machine learning performance, while ensuring the safety and privacy of individual farms or data silos. However, the conventional FL approach has two major limitations. First, the heterogeneous data on individual silos can cause the global model to perform well for some clients but not all, as the update direction on some clients may hinder others after they are aggregated. Second, it is lacking with respect to the efficiency perspective concerning communication costs during FL and large model sizes. This paper proposes a new technical solution that utilizes network pruning on client models and aggregates the pruned models. This method enables local models to be tailored to their respective data distribution and mitigate the data heterogeneity present in agri-food data. Moreover, it allows for more compact models that consume less data during transmission. We experiment with a soybean yield forecasting dataset and find that this approach can improve inference performance by 15.5% to 20% compared to FedAvg, while reducing local model sizes by up to 84% and the data volume communicated between the clients and the server by 57.1% to 64.7%. Our method demonstrates the potential to use efficient models that are more environmentally friendly to support the agri-food sector's transition to net zero. Future enhancements of this method could further optimize distributed learning in agri-food, enhancing sustainability and applicability.

## 1. Introduction

The agri-food supply chain involves the whole journey from farm to fork, including agriculture, food processing, warehousing systems, distribution and marketing. Data analytics hold the key to ensuring food security and sustainability. Machine learning has been widely adapted to provide technical solutions to analytical problems in agriculture and food sectors, such as crop yield prediction (Alhnaity et al., 2021; Jeong et al., 2016; Onoufriou, Hanheide, & Leontidis, 2023; van Klompenburg, Kassahun, & Catal, 2020), consumption demand forecasting (Anagnostis, Papageorgiou, & Bochtis, 2020; Ryu, Nasridinov, Rah, & Yoo, 2020), crop and disease detection (Kussul, Lavreniuk, Skakun, & Shelestov, 2017; Mohanty, Hughes, & Salathé, 2016), quality control and intelligent scheduling (Onoufriou, Bickerton, Pearson, & Leontidis, 2019; Rong, Xie, & Ying, 2019; Thota & Leontidis, 2021), and several others.

Typically, building such statistical models requires large amounts of data collected from various sources, i.e., different farms, supply chains, and other stakeholders. However, individually, they may not have adequate data to train competent machine learning models for the tasks. While combining their data into a centralized silo may improve data quality, collecting it may be challenging due to commercially sensitive information and reputational risks (Durrant et al., 2021). Federated Learning (FL) is a well-established training algorithm that addresses this by allowing a model to be trained decentrally without physically sharing the data but instead sharing the model information only (McMahan, Moore, Ramage, & y Arcas, 2016). Each participating device (referred to as a client) participates in training in an isolated environment and is coordinated by the central server. As a result, FL allows models to be collaboratively trained on large datasets while preserving data privacy. This approach is particularly useful in the

* Corresponding author at: School of Natural and Computing Sciences, University of Aberdeen, Aberdeen, AB24 3UE, UK.
*E-mail addresses:* a.li.21@abdn.ac.uk (A. Li), milan.markovic@abdn.ac.uk (M. Markovic), p.edwards@abdn.ac.uk (P. Edwards), georgios.leontidis@abdn.ac.uk (G. Leontidis).

https://doi.org/10.1016/j.eswa.2023.122847
Received 1 September 2023; Received in revised form 21 November 2023; Accepted 3 December 2023

agri-food industry, where training on a local dataset alone may be insufficient. For example, FL can enable coordinated crop yield prediction across multiple farms, allowing each farm to benefit from a more robust model without having to share sensitive or proprietary data.

FL requires transmission of the model between the server and the clients each round. However, with the progressive improvements in deep learning models, their number of parameters has increased exponentially (Menghani, 2023), and the communication cost often becomes a bottleneck (Konečnỳ, McMahan, Ramage, & Richtárik, 2016). Large models also make edge device deployment challenging as they consume more memory footprint and computational power. Moreover, in real-world scenarios, clients typically hold heterogeneous data, meaning the data distribution is different and can be diverse in nature even when the data measurements are held consistent. For example, when data from hundreds of farms is used to build a model for crop yield prediction, it often comes from different regions with inconsistent readings. Such inconsistencies could arise from disparities in sensor deployment or from variables not reflected in the data, such as differing crop genetics and soil types. Heterogeneous data can cause the model to generalize poorly, especially for clients whose datasets deviate significantly from the mean distribution.

In this paper, we propose a federated learning strategy, which we name Federated Pruning (FedPruning) to address these limitations. Neural networks are typically over-parameterized and there is much redundancy (Denil, Shakibi, Dinh, Ranzato, & de Freitas, 2014). It has been consistently shown that same inference performance can be reached with only a fraction of the original model size (Blalock, Ortiz, Frankle, & Guttag, 2020). We leverage the theoretical benefits of pruning, and remove redundant parameters from individual client models. Through our training algorithm, all client models have different connections and weight values by the end of training, and they become localized and tailored to their own data. The resulting models not only have better local inference performance compared to the previous method FedAvg, but they are also smaller in size, more energy and memory efficient. Communication is a common bottleneck in FL, and we are motivated by the usage of machine learning models in agri-food both in the traditional broadband and Internet of Things (IoT) settings where hardware capacity and internet are more limited. Reducing the number of parameters in the models can decrease the data exchange volume of the participants, resulting in lower communication costs and feasibility of edge device implementation, considering the internet and coverage may be more unreliable and limited in rural settings.

To showcase the potential benefits of FedPruning,[1] we employ an established dataset and a CNN architecture from a previous study for soybean yield forecasting (Khaki, Wang, & Archontoulis, 2020). We build a centralized model, local only models and a FL model as three baselines, and compare four variations of FedPruning with them. We demonstrate that our method improves inference performance of local models, reduces communication cost during training and results in smaller sizes compared to the FL baseline. In summary, this work describes the following contributions:

- To the best of our knowledge, this is the first study to conceptualize a communication-efficient machine learning methodology that is built upon data privacy and allows for decentralized training with neural network pruning in an agri-food setting.
- We propose a new pruning methodology that improves inference performance and communication efficiency at the same time in a FL setup.
- We demonstrate the applicability of our method on a real-world dataset and show that it outperforms both centralized and FedAvg baselines, suggesting it as a viable option in decentralized agri-food settings.

[1] See the project on GitHub: https://github.com/TensorStrike/soybean_fedpruning.

## 2. Background & related work

### 2.1. Federated learning

Federated Learning (FL) offers a decentralized approach for training models across multiple data sources. It addresses two key challenges that are often present in training local models using local data: preserving data privacy by keeping data on local devices and improving the overall performance of models, which might otherwise be limited by the volume and bias of local data (Bonawitz et al., 2016; McMahan et al., 2016). Recently, FL has been widely adapted into many fields, such as health care (Dayan et al., 2021; Huang et al., 2019; Ma et al., 2022; Nguyen et al., 2022; Pfohl, Dai, & Heller, 2019; Rieke et al., 2020; Xu et al., 2021; Zhang, Li, Ma, Luo, & Li, 2021), IoT (Imteaj, Thakker, Wang, Li, & Amini, 2021; Khan, Saad, Han, Hossain, & Hong, 2021; Nguyen et al., 2022; Yu et al., 2020) and finance (Imteaj & Amini, 2022; Long, Tan, Jiang, & Zhang, 2020; Yang, Zhang, Ye, Li and Xu, 2019).

However, it poses two notable challenges in practice setting. First, The frequent exchange of model updates between the server and clients often becomes a bottleneck in the training pipeline, preventing the effective training of the global model, particularly when the models are large (Bonawitz et al., 2019; Qiu et al., 2020; Yang, Liu, Chen and Tong, 2019). This issue is further exacerbated in rural settings where limited bandwidth can significantly hinder the aggregation process (Wu, Wu, Lyu, Huang, & Xie, 2022). Additionally, the varying hardware capabilities and dataset sizes among clients complicate the timing and efficiency of model aggregation (Shi & Radu, 2021; Wang, Liu, Liang, Joshi, & Poor, 2020). Efforts have been made to reduce the communication cost. Some approaches compress the size of the updates with reduced dimensionality of the gradient (Konečnỳ et al., 2016; Li & Han, 2019), quantization (Amiri, Gunduz, Kulkarni, & Poor, 2020; Konečnỳ et al., 2016; Prakash et al., 2022; Reisizadeh, Mokhtari, Hassani, Jadbabaie, & Pedarsani, 2020) or knowledge distillation (Jeong et al., 2018; Wu et al., 2022) before sending them to the server, while others focus on optimizing update schemes or strategically excluding less impactful devices to improve efficiency (Caldas, Konečny, McMahan, & Talwalkar, 2018; Hamer, Mohri, & Suresh, 2020; Paragliola & Coronato, 2022; Tao & Li, 2018). Our paper contributes to this ongoing effort and falls into the category of model compression with the use of pruning before aggregation.

Heterogeneity is yet another challenge in FL. While it can take on many forms, statistical heterogeneity, especially in the form of non-IID (non-Independently and Identically Distributed) data distributions is the most common and extensively researched issue (Li, Sahu, Talwalkar, & Smith, 2020a; Shi & Radu, 2021; Yang, Liu et al., 2019). This non-uniform distribution of data among clients often leads to biased and suboptimal model performance. This issue can be explained by weight divergence, where the global model, influenced by the non-IID data, deviates from the ideal model that would be obtained if the data were IID. Over time, and through successive communication rounds, this divergence tends to worsen, adversely affecting both model convergence and overall performance (Zhu, Xu, Liu, & Jin, 2021). While several solutions (Dinh, Tran, & Nguyen, 2022; Karimireddy et al., 2019; Li, Jiang, Zhang, Kamp and Dou, 2021; Li et al., 2018; Xu, Chen, Quek, & Chong, 2022) address this and improve the overall performance, they often do not take into account the efficiency aspect and in some cases come at the cost of increased communication or extended training times, making them less practical for real-time applications. Each method inevitably comes with trade-offs - Table 1 collects various mainstream methods mentioned in this section with their advantages and limitations.

**Table 1**
Established federated learning methods.

| Method | Key feature | Contribution | Limitation |
|---|---|---|---|
| FedBN (Li, Jiang et al., 2021) | It treats batch norm separately during aggregation and keep these parameters local. Doing so allows each client to keep its unique data distribution. | It effectively address feature skew in heterogeneous data. | It does not focus on communication efficiency; requires the model to use batch norm to take advantage of this. |
| SCAFFOLD (Karimireddy et al., 2019) | It estimates the update directions of the server and local model updates and then 'corrects' the local updates with the difference between the server and local model. | It reduces client drift in the presence of heterogeneity and improves the learning consistency across clients. | The process of estimating and correcting based on the server and local model discrepancies may introduce additional computational overhead. |
| FedProx (Li et al., 2018) | It adds a penalty term to FedAvg, providing a balance between learning from local data and adhering to the global model. | It mitigates the impact of statistical heterogeneity and client drift, leading to more stable convergence. | The added penalty term may increase the computation overhead and it does not address communication cost. |
| pFedME (Dinh et al., 2022) | It employs Moreau envelopes as clients' regularized loss functions, which encourages localization in FL. | It achieves significant speedups on convergence. | It requires extra computation and does not improve communication efficiency; regularization may need extra tuning. |
| MOON (Li, He and Song, 2021) | It works to minimize the difference between what the local and global models learn, while maximizing the difference between the current and previous learning of the local model. | It addresses heterogeneity, improves training stability as well as reduces the number of communication rounds. | It involves comparing representations learned by different models may increase the computational complexity; the effectiveness may be diminished if the global representation is suboptimal. |
| FedCorr (Xu et al., 2022) | Additional steps are designed to identify noisy clients, and fine-tune on clean clients | It addresses issues of having high noise clients and addresses heterogeneity. | The additional steps are computationally expensive and not suitable for IoT. |
| McMahan et al (Konečnỳ et al., 2016) | It proposes an algorithm tailored to a setting where a large number of nodes with uneven data distribution is involved. It focuses on incurring more computation on local devices, thus reducing the number of communication rounds. | It reduces the communication cost by reducing the rounds of communication. | The paper's focus is more on communication efficiency in distributed settings rather than directly addressing non-IID data challenges; may not be applicable to cross-silo scenarios with fewer clients. |
| FedKD (Wu et al., 2022) | It compresses the updates with the use of knowledge distillation and gradient compression techniques. | It saves a significant amount of communication cost while achieving close to centralized performance. | It does not directly address heterogeneity; achieving the right balance between learning from the distilled model and retaining representations from local data can be challenging, especially in highly heterogeneous environments. |
| FedBoost (Hamer et al., 2020) | It trains an ensemble of base predictors, which work together to improve the overall accuracy reducing the need for communication. | It focuses on minimizing per-round communication costs for both server-to-client and client-to-server. | It is computationally expensive and not suitable for IoT. |
| GWEP (Prakash et al., 2022) | It uses joint quantization and model pruning to compress models in FL. | It significantly reduces communication cost and model size, making it suitable for IoT deployment. | It does not address heterogeneity and high compression may sacrifice performance. |

### 2.2. Pruning

Pruning techniques are pivotal for efficient neural network deployment, especially in resource-constrained applications like crop yield prediction (Han, Mao and Dally, 2015). Early work suggests that pruning aids model generalization by balancing the bias–variance trade-off (LeCun, Denker, & Solla, 1989; Rasmussen & Ghahramani, 2000). Recent studies confirm that moderate pruning can even improve model accuracy (Han, Pool, Tran and Dally, 2015). The primary motivation for contemporary pruning methods is to enable energy efficiency for real-time operation on mobile devices and reduce the model size for easier storage and transmission (Han, Pool et al., 2015).

Most modern pruning algorithms stem from Han et al.'s three-step process: initial training, weight removal based on importance, and fine-tuning (Han, Pool et al., 2015). Pruning effectively reduces redundant weights without compromising performance. For instance, AlexNet and VGG-16 can be pruned by 9x and 13x, respectively, without loss in performance (Han, Pool et al., 2015). Various criteria exist for weight removal, with magnitude-based pruning being widely adopted (LeCun et al., 1989). The recent Lottery Ticket Hypothesis introduces a method that optimal sub-networks can be found by re-initializing weights after pruning, and offers an alternative to fine-tuning (Frankle & Carbin, 2018). The sub-networks can be trained in isolation from scratch to reach the performance no worse than the original with equal or less training.

## 3. Materials and methods

### 3.1. Data

Our demonstration focuses on collaborative federated forecasting using an accessible open-source dataset, given limited real data availability in agri-food. We use the tabular data analyzed from a previous work of Khaki et al. (2020) for the same task of predicting the yield of soybean (bushels per acre). The dataset is composed of weather, soil and management data of soybean from 9 states and their counties from 1980 to 2018. In order to maintain consistency with the previous study, we use data from 1980 to 2015 to predict yield for the final three years 2016, 2017, and 2018.

**Table 2**
Data sample breakdown for the states used for prediction year 2016, 2017 and 2018. Each state represents a silo used in FL.

| Location | 2016 | | 2017 | | 2018 | |
|---|---|---|---|---|---|---|
| | Train | Val | Train | Val | Train | Val |
| Illinois | 2977 | 67 | 3044 | 72 | 3116 | 64 |
| Indiana | 2630 | 52 | 2682 | 54 | 2736 | 61 |
| Iowa | 3132 | 94 | 3226 | 90 | 3316 | 86 |
| Kansas | 2443 | 17 | 2460 | 19 | 2479 | 15 |
| Minnesota | 2134 | 55 | 2189 | 55 | 2244 | 43 |
| Missouri | 2395 | 18 | 2413 | 19 | 2432 | 15 |
| Nebraska | 2274 | 52 | 2326 | 43 | 2369 | 39 |
| North dakota | 574 | 12 | 586 | 12 | 598 | 12 |
| South dakota | 1164 | 22 | 1186 | 21 | 1207 | 20 |
| Combined | 19 723 | 389 | 20 112 | 385 | 20 479 | 355 |

- The weather data includes the weekly average of 6 attributes: precipitation, solar radiation, snow water equivalent, maximum temperature, minimum temperature and vapor pressure. This data was acquired from Daymet (Thornton et al., 2020).
- The soil data includes 10 attributes: bulk density, cation capacity exchange capacity at 7 pH, course fragments, clay percentage, total nitrogen, organic carbon density, organic matter percentage, pH in water, sand, silt, soil organic carbon, all measured at 6 depths. The data was acquired from Gridded Soil Survey Geographic Database (gSSURGO, 2023) for the United States and is generally the most detailed level of soil geographic data in accordance with the national survey standards.
- The management data includes the cumulative percentage of planted fields within each state. This data is acquired from National Agricultural Statistics Service of the United States (USDA-NASS, 2019).

The soil data is uniform throughout the period for each county while the weather and management data change over time. The data is distributed into 9 silos, with each representing a corresponding state. Following a cleaning and data processing approach similar to that described by Khaki et al. (2020), we end up with silos containing a diverse range of samples, as detailed in Table 2. Such an imbalance in data distribution is common in practical scenarios. Clients with a large training data volume account for a larger proportion of the overal training data, and can reduce the accuracy of clients with fewer samples (He & Garcia, 2009). To address this, we implemented a strategy akin to random oversampling. This method aims to balance the dataset by adjusting the sample size across silos without losing information. It involves replicating samples from under-represented silos more frequently, and those from over-represented silos less so, resulting in a more uniform distribution among all silos.

### 3.2. Federated pruning

The ultimate objective of FL (McMahan et al., 2016) is to find a set of parameters, denoted by $\theta_{global}$ that minimizes the global loss function $\mathcal{L}$ across all clients, $K$, such that:

$$\theta_{global} \in \arg\min_{\theta} \mathcal{L}(\theta) := \frac{1}{K}\sum_{i}^{K} \ell_i(\theta) \tag{1}$$

This is particularly relevant in the agri-food contexts where individual silos often lack sufficient data to train an accurate model and data holders are often reluctant to share the data due to privacy and security concerns. However. the heterogeneity in data can lead to performance degradation on clients experiencing distribution shifts.

To address these challenges, we leverage pruning as described in Eq. (2). $\|\theta_p\|_0$ the L0 norm of the pruned parameters $\theta_p$, represents the number of non-zero elements. To obtain the pruned network $\theta_p$, the model is pruned until $\|\theta_p\|_0$ is less than a preset threshold $N$.

$$\arg\min_{\theta_p} \mathcal{L}(x;\theta_p) \quad \text{subject to: } \|\theta_p\|_0 < N \tag{2}$$

During FL, we prune local weights at the end of each local training cycle. This results in localized models with unique sets of parameters tailored to their own data distributions. The pruned models are not only more accurate but also smaller in size, thereby reducing the communication overhead. This is especially beneficial in agri-food settings like farms, where bandwidth is often limited. The server then aggregates these sparse models and disseminates the updated weights to the clients. This iterative process continues for $T$ communication rounds, ultimately yielding localized and compact models. We refer to this method as Federated Pruning (FedPruning) throughout this paper. FedPruning can be described as in Eq. (3). We obtain client parameters $\theta_k$ by averaging client $\theta_p$ (obtained with a prune function $P$) in aggregation during federated learning (see Fig. 1).

$$\theta_k \in \arg\min_{\theta_p} \mathcal{L}(\theta_p) := \frac{1}{K}\sum_{i}^{K} \ell_i(x;\theta_p) \quad \text{where } \theta_p = P(x;\theta) \text{ s.t. } \|\theta_p\|_0 < N \tag{3}$$

Unlike typical FL scenarios involving random client selection, our approach is specifically designed for case scenarios in agri-food where the number of clients, such as farms or silos, is finite and well-defined. In such contexts, random selection is not only unnecessary but also counterproductive, as each client contributes valuable, albeit heterogeneous, data that is crucial for the global model. Therefore, in our FedPruning method, all available clients are included in each communication round for model aggregation. This ensures that the global model benefits from the full spectrum of data distributions, making it more robust and applicable to the specific challenges of agri-food systems.

### 3.3. Localization-preserving aggregation

The primary incentive to participate in FL is to have a global model that is better performing than the individual local models. However, in practice, local clients can outperform the global model due to data heterogeneity (Hanzely & Richtárik, 2020) or not independent and identically distributed (Non-IID) data, and it defeats the purpose of FL. For a supervised learning task on client $k$, the data is in the form of (X,y) where $x$ is the input features and $y$ is the label, and it follows a local distribution $P_k(X, y)$. By Non-IID, the $P_k(X, y)$ differs from client to client (Zhu et al., 2021). We may experience this from different types of skews. First, the conditional distribution $P_k(y|X)$ may be different across the clients although $P_k(X)$ is the same. In the agri-food sector, this could be resulted from the different measuring devices or sensitivities of sensors used to capture the local data. Second, for time-series data such as ours, it can happen when clients have uneven distribution across the years. Some may have more data points towards later years while others have more from the early years.

While the widely popular FedAvg can work with non-IID to some degree, it ultimately lacks the theoretical guarantee to converge for all clients (Li, Sahu, Talwalkar, & Smith, 2020b). In the agri-food sector, feature shifts, often caused by variations in local measurement devices or measurement sensitivity are the primary reason for non-IID data across local data silos. Such shifts in feature distribution can cause performance degradation, as local models are trained on distributions that are not aligned with those of other clients. During communication rounds, FedAvg aggregates the gradients of the local models by taking the weighted average of the local gradients and returning it back to the clients (McMahan et al., 2016). It results in handling all the various data distributions with one single global model. In our method, instead of attempting to obtain a "one model to fit all", each client learns a localized sparse network that is tailored to its own data distribution.

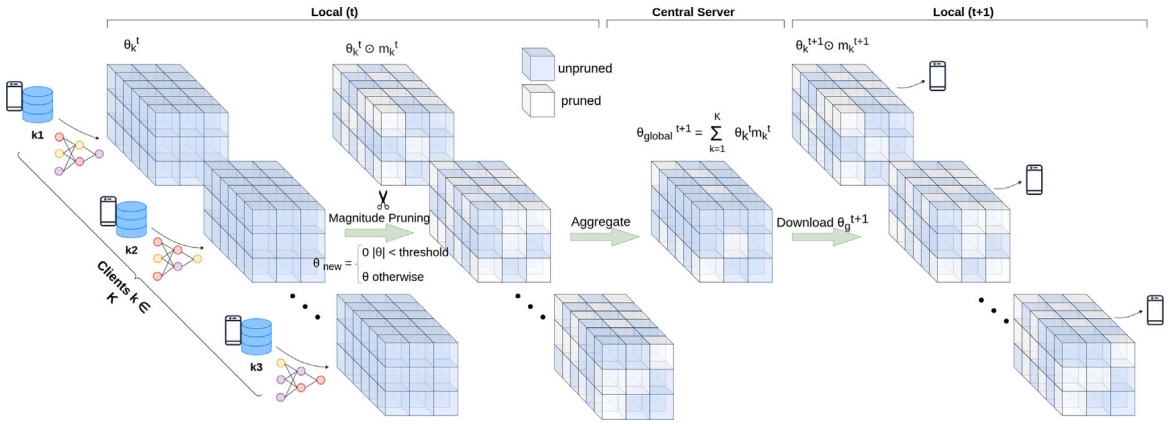



**Fig. 1.** Federated Pruning during one round of federated learning.

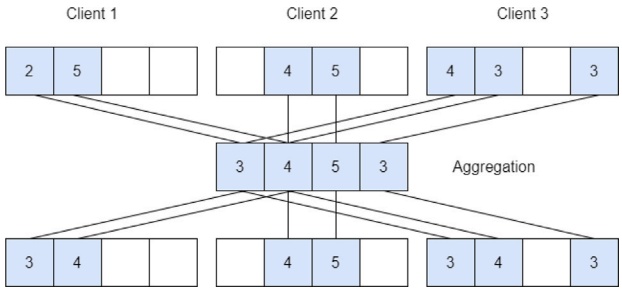



**Fig. 2.** Localization-preserving aggregation. The original weights (top) are aggregated (middle) to become new weights (bottom), which are updated with on the clients.

Following the idea of FedAvg where the average weights are proportional to the number of participants, we use the aggregation algorithm proposed in LotteryFL (Li, Sun et al., 2021), which is designed to aggregate sparse networks. Largely pruned networks may overlap with each other on some connections while having unique or infrequently overlapped connections due to the non-IID data distribution across the clients. This aggregation strategy updates on only the overlapped connections of the sub-networks while keeping the non-overlapped parts unchanged. Fig. 2 illustrates how the averages of sparse tensors are computed. For instance, at the leftmost position, we take the average from client 1 and client 3 since client 2 has it pruned. The aggregated weights are then sent back to the clients and the unpruned weights are updated. This aggregation strategy allows the server to maximally preserve weights that are important to individual clients.

### 3.4. Communication cost

In the field of agri-food, a persistent issue is the insufficient availability of robust wireless connectivity in rural regions. Any delay in data exchange or loss of connection among IoT devices such as sensors or electronic devices could directly hinder farming operations (van Hilten & Wolfert, 2022). The application of sparsity to neural networks is a widely adopted technique for minimizing the number of parameters transmitted to the server, and thereby reducing communication costs (Aji & Heafield, 2017; Alistarh et al., 2018). Communication constitutes a common bottleneck in FL since the participating clients are regularly required to send and receive updates from the server. By removing a portion of parameters per update, the communication cost is reduced to the compact size of a sub-network from the full network.

### 3.5. Iterative magnitude pruning

We obtain a sparse neural network on each client $k \in K$ by training the network and pruning its smallest-magnitude weights. Consider a dense neural network $f(x;\theta_k)$ with parameters $\theta_k$, when optimizing with gradient descent on a training set, the validation loss $l_k$ converges. In the implementation, pruning a percentage of the weights leads to the generation of a mask $m_k \in \{0,1\}$. If the magnitude of parameter is smaller than the quantile, the corresponding mask entry is set to 0. The mask is combined with the state of the network to produce a sub-network $f(x;\theta_k \odot m_k)$, which is then trained again to recover $l_k$ (If LTH is used, the weights of the sub-network are reset to $\theta_g$ at initialization before re-training). The network is trained and pruned over $T$ rounds; each round prunes $p\%$ of the remaining weights that survived the round. This iterative magnitude pruning step makes the basis of the client updates presented in our approach.

We define the sparsity of a network as $P_m = \frac{\|m\|_0}{|\theta|}$, with $\|m\|_0$ being the number of zeros in a mask and $|\theta|$ being the number of weights in a network, i.e., $P_m = 75\%$ means that 75% of weights have been pruned. $P_m$ is used as the metric to evaluate the compactness of the models throughout this paper. Our approach produces sub-networks $\exists m$ for which $l' \approx l(comparable\ losses)$ and $\|m\|_0 \ll \theta(fewer\ parameters)$ for all clients $K$.

### 3.6. Training algorithm

The general training algorithm is formally described in Algorithm 1. The steps are taken as follow:

1. Global weights are downloaded from central server to clients.
2. Perform local training on each client for $e$ epochs.
3. If $loss < loss_{best}$, prune $p\%$ of the parameters $\theta$ by magnitude, generating mask $m$. Train for $e$ epochs to recover the loss.
4. The current sub-networks $\theta_k^t$ of the clients are sent to the global server for aggregation. Update the global model $\theta_k^t$ to $\theta_k^{t+1}$.

The above steps are iterated for $T$ rounds. We use magnitude as our pruning criterion to determine the importance of weights. To give networks enough time to converge, we avoid pruning prematurely by running regular FL without pruning for 2 communication rounds — we find that it greatly reduce the risks of networks failing to recover, and increases the pruning potential in future rounds.

In our test setting, clients converge at different rates and their eventual losses vary. Thus, we perform pruning based on their own convergence and use their best loss as an indicator for whether they are ready to be pruned. We ensure that the loss recovers to its previous best

**Algorithm 1** Training Algorithm. T rounds are indexed by t; The clients are indexed by $k$; r is the prune ratio; m is the local mask; $\eta$ is the learning rate; $\ell$ is the loss function

initialize global model $\theta_{global}$ with $\theta_0$
**while** *round* $t < T$ **do**
  **ClientUpdate**($\theta_{global}$) :
  **for** *each client k1, k2* … **do**
    $\theta_k^t \leftarrow \theta_{global} \odot m_k^t$
    $\theta_k^t \leftarrow \text{Train}(\theta_k^t)$
    ▷ If loss has recovered; target sparsity has not reached; no pruning in final rounds
    **if** $loss < loss_{best}$ *and* $r^t < r_{target}$ *and* $t < T - 3$ **then**
      $m_k^t \leftarrow$ *Prune p% of* $\theta_k^t$
      $\theta_k^t \leftarrow \theta_0$ ▷ Reset weights to initialization (if LTH is used)
      $\theta_k^t \leftarrow \text{Train}(\theta_k^t \odot m_k^t)$ ▷ recover loss
    **end if**
    Return $\theta_k^t$ to server
  **end for**
  **Server Executes:**
  $\theta_{global}^{t+1} \leftarrow \text{aggregate}(\theta_{k1}^t, \theta_{k2}^t, \dots)$ ▷ Note: client weights have already been masked
**end while**
**function** TRAIN($\theta_k^t$)
  **for** *epoch* $e = 1, 2, \dots,$ **do**
    **for** *batch* $b \in B$ **do**
      $\theta_k^{t+1} \leftarrow \theta_k^t - \eta \nabla \theta_k^t \ell(\hat{y}, y)$
      **return** $\theta_k^{t+1}$
    **end for**
  **end for**
**end function**

before it can be pruned again. In practice, we set this threshold to 10%–20% over its best loss to accelerate pruning early on as it is likely that the loss will recover in later rounds. Obtaining smaller models early on means that there is less communication cost throughout the entire FL training process. To allow losses to recover and stabilize following pruning, a number of non-pruning round is added.

### 3.7. Model architecture

The previous work by Khaki et al. (2020) tested five different models for this forecasting task — CNN–LSTM, Random Forest, Deep fully connected neural network (DFNN) and LASSO. The CNN–LSTM was most effective in predicting yields of soybean with RMSE for the validation data being approximately 8%. However, the line of research for iterative magnitude pruning is almost exclusively based on convolutional and fully connected layers (Liang, Glossner, Wang, Shi, & Zhang, 2021), so we build a model based on these layers to realize the theoretical benefits of pruning. We utilize the previously established convolutional layers to capture the temporal structure of the data for weather, soil and management, but concatenate them along with yield data from the previous dependent years into the fully connected layers, which work as our regressor. Batch normalization is also used after the non-linearity to accelerate training and improve the accuracy (Santurkar, Tsipras, Ilyas, & Madry, 2018). We experimented with different configurations and found that 3 fully connected layers gave an accuracy comparable to the DFNN model from the previous work. Rectified linear unit (ReLU) activation function is used for all convolutional and fully connected layers.

### 3.8. Metrics

We use several key metrics to evaluate performance and efficiency in this study. First, we use the Root Mean Square Error (RMSE) as the primary performance metric to assess the accuracy of our yield predictions. RMSE provides a reliable measure of the model's prediction errors, allowing us to understand the accuracy of regression model.

For assessing model compression, we relied on the sparsity metric, previously defined as $P_m = \frac{\|m\|_0}{|\theta|}$, and this is the ratio between zero and non-zero weights. Additionally, we measure the theoretical size of the pruned models based on the number of parameters, first converting this count into size in bits, and then a more tangible metric of KB.

An important aspect of our study was evaluating the communication cost savings achieved through model compression during transmission in the FL process. We calculated this by measuring the reduction in bandwidth usage due to model compression. For example, if a model is compressed by 50% at a certain communication round, subsequent rounds would require correspondingly less bandwidth and have a compounding effect. This effect accumulates over the course of the process, and the result shows the theoretical amount saved in MB during the whole FL process.

## 4. Results

### 4.1. Experiment setup

Our experiments were conducted on a laptop with an Intel Core i7 processor, 16 GB RAM, and an NVIDIA GTX2080 Super GPU, using Python 3.7, PyTorch 1.13.1 with CUDA 11.7 support, and additional dependencies listed in our repository. To evaluate the inference accuracy of the models, the data for 2016, 2017 and 2018 are used as validation years and their yields are predicted in bushels per acre. We implemented three baselines to make a fair comparison.

- **Centralized** is trained on the combined data of the clients.
- **Local only** is trained locally by each client.
- **FedAvg** (McMahan et al., 2016) is the classic FL approach where clients download the global model from the server, train using local data, and then send updates to the server to update the global model through aggregation.

In addition to the baselines, we implement and evaluate FedProx and FedBN, two popular mainstream methods that address non-IID data. These methods are benchmarked for a comparison in efficacy in handling the complexities inherent in non-IID dataset in real life, such as ours.

- **FedProx** (Li et al., 2018) introduces a proximal term that penalizes large deviations from the global model to encourage stability and less divergence among client updates. We set $\mu$ in the proximal term definition in Li et al. (2018) to 0.01 for our experiment.
- **FedBN** (Li, Jiang et al., 2021) keeps batch norm layers specific to clients while updating other parameters. In the implementation, we revise the update logic such that the batch norm parameters are not updated from the server.

To make local models more accurate and compact in our own method, each sub-network prunes the least important parameters after local training. The hypothesis behind this is that the surviving parameters are those that are most important to the local clients (not other clients), and the aggregation of pruned models will preserve the localized parameters derived from pruning. To evaluate the effectiveness of our proposed method, we implement four variations, and compare them with the baselines.

- **Federated Pruning (FedPruning)** is the method described in Algorithm 1 where individual models are pruned on the client's end and sub-networks are aggregated on only the overlapped connections on the server's end. The surviving sub-networks are fine-tuned and their losses are recovered.

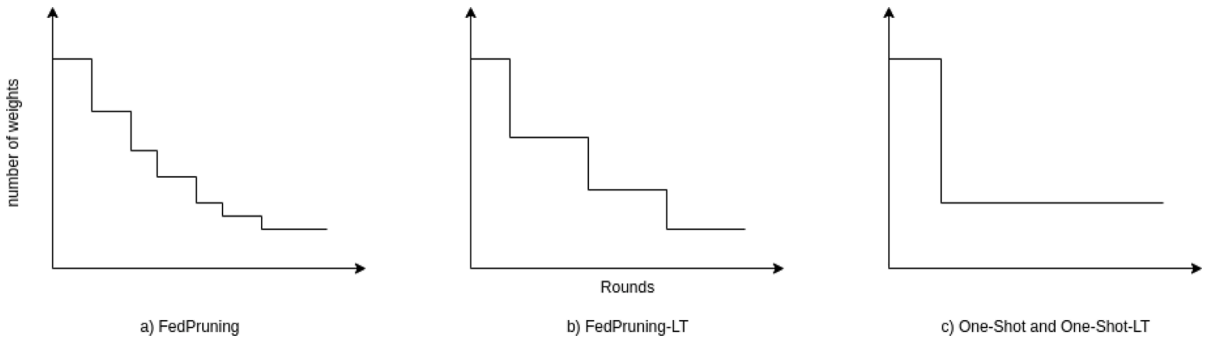



**Fig. 3.** Sparcification schedules.

**Table 3**
Settings tested in this experiment. The pruning rate denotes the percentage of weights pruned each time pruning is executed, and target sparsity represents the intended level of sparsity achieved at the end of training. Notably, the sparsity achieved by iterative pruning may slightly deviate from the target sparsity depending on the training performance.

| Method | Rounds/Local epochs | Learning rates | Pruning rate | Target sparsity |
|---|---|---|---|---|
| Centralized | 1/300 | 5e−5, 1e−5, 2e−6 | 0.0 | 0.0 |
| Local only | 1/300 | 2e−5, 4e−6, 8e−7 | 0.0 | 0.0 |
| FedAvg | 40/5 | 2e−5, 4e−6, 8e−7 | 0.0 | 0.0 |
| FedProx | 40/5 | 2e−5, 4e−6, 8e−7 | 0.0 | 0.0 |
| FedBN | 40/5 | 5e−5, 1e−5, 2e−6 | 0.0 | 0.0 |
| FedPruning (ours) | 40/6 | 2e−5, 4e−6, 8e−7 | 0.25 | 0.80 |
| FedPruning-LT (ours) | 40/8 | 2e−5, 2e−6, 4e−7 | 0.415 | 0.80 |
| One-shot (ours) | 40/5 | 2e−5, 4e−6, 8e−7 | 0.70 | 0.70 |
| One-shot-LT (ours) | 40/5 | 1e−5, 2e−6, 4e−7 | 0.70 | 0.70 |

- **Federated Pruning with Lottery Ticket (FedPruning-LT)** is trained on the same algorithm as FedPruning, but with LTH (Frankle & Carbin, 2018), which resets the remaining weights to initialization, creating a winning ticket. Both FedPruning approaches prune iteratively, which repeatedly train, prune, re-train, and aggregate over T rounds. Each round prunes $p\%$ of the survived weights until target sparsity is reached.
- **One-shot** prunes all client networks by a large $p\%$ at once (Lee, Ajanthan, & Torr, 2019) and proceeds with federated learning with the same aggregation strategy as FedPruning.
- **One-shot with Lottery Ticket (one-shot-LT)** executes a one-shot pruning, but with weights reset to initialization following pruning.

For all methods, we use a batch size of 50, and the weights are initialized with the Kaiming method (He, Zhang, Ren, & Sun, 2015). Adam optimizer is used with a learning rate decaying at around 5 and 10 by a factor of 0.2. L2 regularization can be used to enforce sparsity during training by encouraging smaller weights (Han, Pool et al., 2015), and therefore we set a weight decay of 0.0001. For federated methods, the model is trained for a maximum of 40 communication rounds with 5–8 epochs trained locally. Local early-stopping is also used to prevent overfitting. Table 3 includes additional parameter settings used in this experiment.

**Sparsification Schedules**. Model sparsity is trained with a schedule. As shown in Fig. 3, FedPruning (a) prunes a relatively small percentage iteratively, whereas FedPruning-LT (b) prunes a maximum of three times but a larger portion each time. This is because, in our testing, LT requires more iterations to recover due to the weight reset. If a network is pruned prematurely before the loss is recovered, it can impede its ability to recover its former loss and also prevent further pruning. Hence, FedPruning-LT is given a minimum of 7 rounds to recover following a pruning round, while FedPruning is given a minimum of 3 rounds to recover. In addition to these mandatory recovery rounds, clients are evaluated before pruning and pruning is only executed if their losses have been recovered. Pruning is also prohibited in the final rounds to ensure the best final inference performance. Both one-shot approaches (c) prune a significant portion as soon as the network converges and proceed with training and aggregating sparse networks.

**Table 4**
RMSE (bushels per acre) of the 9 states trained using different training procedures. The values are recorded using the average of 3 runs each year with random initialization seeds. The final average sparsity for our methods are in the bracket. The best performance is highlighted in bold.

| Method | 2016 | 2017 | 2018 | Average |
|---|---|---|---|---|
| Centralized baseline | 8.74 | 6.04 | 5.83 | 6.8 |
| Local only | 10.10 | 7.46 | 7.96 | 8.84 |
| FedAvg | 9.70 | 6.53 | 6.85 | 7.69 |
| FedProx | 9.94 | 5.92 | 6.25 | 7.37 |
| FedBN | 10.06 | 5.98 | 6.48 | 7.51 |
| FedPruning (ours) | 8.87 (0.75) | 5.02 (0.84) | 5.62 (0.78) | 6.50 (0.79) |
| FedPruning -LT (ours) | 9.52 (0.73) | 5.24 (0.76) | 5.52 (0.76) | 6.75 (0.75) |
| One-shot (ours) | 8.50 (0.7) | 5.54 (0.7) | 5.55 (0.7) | 6.53 (0.7) |
| One-shot-LT (ours) | **8.39 (0.7)** | **4.85 (0.7)** | **5.26 (0.7)** | **6.17 (0.7)** |

### 4.2. Inference performance, communication cost and size

The centralized baseline pools all training and testing data together, and demonstrates a commensurate level of inference performance when compared to the DFNN model from the previous study (Khaki et al., 2020). We find that the local models, which only use data from their respective silos, exhibit poor performance compared to the centralized baseline, as indicated in Table 4. This is expected as the local clients have limited training data. FedAvg addresses the data limitation by leveraging all data from all silos via aggregation and model updates. We evaluate the performance of FedAvg, FedProx and FedBN using the global model directly following the aggregation. Our findings indicate that FedAvg outperforms the local models for all years by approximately 10.5%. However, FedAvg's performance still falls short compared to the centralized. When compared to FedAvg, the impact of FedProx and FedBN appears limited as they perform marginally better for some years but worse in others.

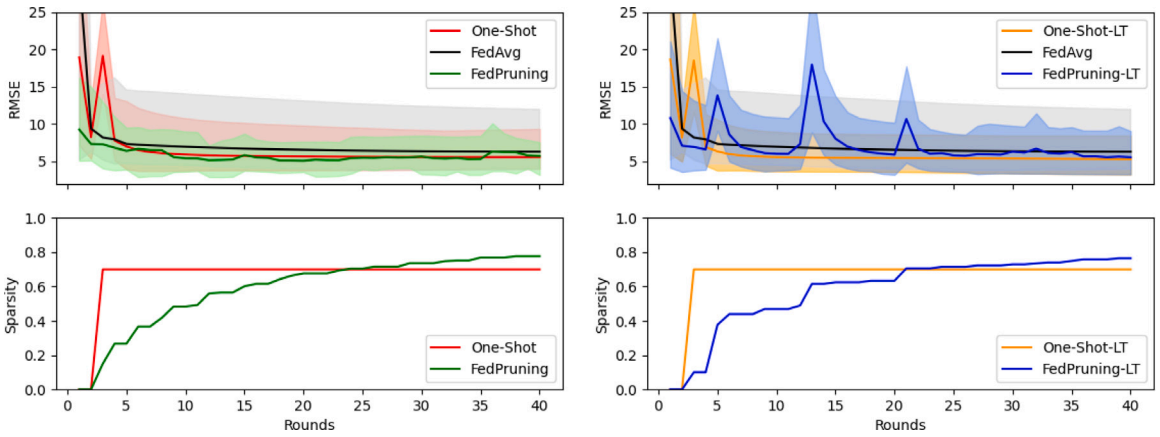



**Fig. 4.** RMSE performance and model sparsity of FedPruning, FedPruning-LT, One-Shot, and One-Shot-LT compared to the FedAvg basedline over 40 communication rounds for 2018. The solid lines at top represent the average RMSE of clients. The shades correspond to the RMSE between the highest and lowest clients. The graph is produced using the average of 3 runs with random initialization seeds.

Since FedPruning produces different, localized models for each client, we evaluate the individual client models instead of the global model like we do with FedAvg. Therefore, to evaluate the performance of FedPruning — both with iterative and one-shot pruning, with and without LTH, we assess the inference performance using local test data from each silo at the end of local training. The most direct way for the method to be demonstrably useful is for it to be a drop-in replacement for FedAvg — that is it must be able to reach the inference performance no worse than FedAvg on all clients on average and result in fewer parameters within a specific number of communication rounds. Symbolically, $\mathcal{L}(\mathscr{A}_{\text{fp}}^{0\to T}(\theta_p)) \leq \mathcal{L}(\mathscr{A}_{\text{fa}}^{0\to T}(\theta))$ and $|\theta_p| < |\theta|$, where $|\theta_p|$ and $|\theta|$ are the numbers of parameters in pruned and unpruned models respectively, $\mathcal{L}$ is the average loss across all clients, $\mathscr{A}_{\text{fp}}^{x\to y}$ and $\mathscr{A}_{\text{fa}}^{x\to y}$ are the FedPruning and FedAvg procedure for training from round $x$ to round y, and $T$ is the final round.

**Assessing the inference performance of local models.** The results indicate that FedPruning under all four settings significantly outperform FedAvg, with performance improvements ranging between 15.5% to 19.8%. The iterative pruning methods FedPruning and FedPruning-LT outperform the centralized baseline for 2017 and 2018, while exhibiting slightly inferior performance for 2016. Overall, these methods demonstrate comparable inference performance to the centralized baseline. One-shot surprisingly shows a similar performance compared to its iterative counterpart, despite being slightly less pruned. However, when LTH is applied to one-shot, it shows a small but noticeable increase in performance across all years, amounting to a 5.5% improvement overall. We also make the observation that these variations collectively are on par with the centralized baseline or even marginally outperform it. One-shot-LT shows the biggest difference by approximately 10%.

Fig. 4 shows how the pruning approaches perform in terms of RMSE and sparsity over 40 communication rounds. It is observed that FedPruning and One-shot (left graph) have a narrower client RMSE range than FedAvg, with the upper bound of client RMSE being lower. Additionally, FedPruning exceeds the final performance at a lower sparsity (approximately $P_m$ = 30%–60%) and decreases slightly as we prune, forming Occam's Hill, which suggests that if the model is either too simple or too complex, performance on an independent test set will suffer (Rasmussen & Ghahramani, 2000). FedPruning-LT (right graph) prunes more each time but fewer times in total. We tested it with the same schedule as FedPruning, and observed that when client models are pruned before they recover, they may permanently lose the performance. This not only affects itself but also other clients since some of their weights are shared. We also find that the overall performance may be better when the clients are pruned and recovered together, as opposed to independently. The RMSE spikes on the graph indicate the impact of weight reset after pruning, and we allow them to recover fully before the next pruning round. Despite the effort, it does not appear to match the performance of FedPruning and One-Shot-LT. One-Shot-LT on the other hand, although simpler to implement, consistently shows superior performance compared to both one-shot and FedPruning-LT.

**Assessing the Cost of Communication and Model Size.** Communication costs in federated learning are primarily influenced by the volume of data transmitted between clients and the server. This volume is affected by two factors: the sparsity of the model which dictates the compression ratio and the timing of model pruning. Sparse models, which are compressed more effectively due to their reduced size, incur lower transmission costs. Moreover, the earlier the pruning occurs, the larger the cumulative savings in communication costs. For example, inducing significant compression early on results in reduced transmission costs in all subsequent rounds. On the other hand, compressing the model in the final round gives no transmission savings in the rounds leading up to it. The bottom graphs of Fig. 4 illustrate these concepts, contrasting the data consumption patterns of iterative versus one-shot pruning approaches. Initially, iterative pruning methods, such as FedPruning, lead to higher communication costs that decrease over time as the models become sparser. In contrast, one-shot pruning methods cut down the model size in a single step, leading to immediate and consistent communication savings, affecting all subsequent rounds. Nevertheless, all pruning techniques successfully reduce the model size while maintaining inference performance. Notably, one-shot pruning achieves a 3.22X reduction in model size, while iterative methods like FedPruning and FedPruning-LT achieve reductions of 4.76X and 4X, respectively (see Table 5).

## 5. Discussion

Existing works (Frankle & Carbin, 2018; Han, Pool et al., 2015) demonstrate that neural networks can be represented by substantially fewer parameters. We drew inspiration from the benefits of network pruning and implemented the fundamental pruning steps, which are training the initial network, removing connections, and fine-tuning the model for the participating local clients in the FL process. Subsequently, we aggregated the overlapping connections of the sub-networks instead of the full network. Through testing with the soybean yield prediction dataset, we observed that the proposed method consistently outperformed the benchmark of the classic FedAvg approach across all years tested. Additionally, this method efficiently produced more compact models for edge device utilization, thereby introducing new options

**Table 5**
Communication cost and size of an average client model during the FL process. The values are recorded using the average of 3 years, with 3 runs per year with random initialization seeds.

| Method | Communication cost (MB) | Communication saved (%) | Client model size (kB) |
|---|---|---|---|
| FedAvg | 50.76 | 0 | 634.64 |
| FedPruning | 21.78 | 57.1 | 133.27 |
| FedPruning-LT | 20.96 | 58.7 | 158.66 |
| One-shot | 17.92 | 64.7 | 196.95 |
| One-shot-LT | 17.92 | 64.7 | 196.95 |

for the implementation of FL in practical agri-food settings. These findings suggest that the proposed method could potentially improve the efficiency and feasibility of FL in agri-food settings, and may offer practical advantages over the classic FedAvg approach.

**Model localization.** In theory, if all client datasets are identically and independently distributed, and the overall data volume is large enough, then FedAvg and the centralized model using combined data could achieve similar inference performance because the client stochastic gradient is an unbiased estimate of the full gradient and the average model weights of the client models will approximate the centralized model (Bottou, 2010; Rakhlin, Shamir, & Sridharan, 2012). However, this assumption nearly never holds in practice, as we can see from our experiment. The non-IID data happens in the presence of inconsistent data distributions when there is an attribute imbalance of the training data across clients due to perturbations. When the number of data collection points becomes large, it is difficult to keep the measurements consistent. For instance, the measurement of temperature may vary between farms due to the deployment of sensors in different positions within the polytunnels. Different farms may also use different fertilizers and be exposed to different climates, or elements not captured in the datasets but affecting the yield. By producing sparse neural networks, the weights important to the client itself are retained. As sparsity $P_m$ increases, the number of parameters shared with other clients via aggregation decreases. A percentage of these remaining weights are nearly unique or shared with a few clients who also consider these weights important to themselves, and these parameters attribute to the localization of client models. A higher sparsity may reduce the divergence from non-IID data, which causes poor performance. This can be seen from our results, as FedPruning consistently outperform FedAvg on individual client models.

**Pruning.** For best results, rather than pruning all weights at once, the common practice is to repeat the train-prune-retrain procedure until the target sparsity is reached. As LeCun et al. (1989) put "A simple strategy consists in deleting parameters with small "saliency", i.e. those whose deletion will have the least effect on the training error... After deletion, the network should be retrained. Of course this procedure can be iterated". The concept of iterative magnitude pruning is also realized by many contemporary research works. Han, Pool et al. (2015), who modernized this method states "Our pruning method ... learns the network connectivity via normal network training... The second step is to prune the low-weight connections... The final step retrains the network... This step is critical. If the network is used without retraining, accuracy is significantly impacted.". Modern literature seems to agree that pruning should be performed iteratively for best inference performance. However, in our study, we did not observe a significant disparity between the iterative and one-shot approaches. This may suggest that the true potential of iterative FedPruning has yet to be realized. Local losses degrade after the models are pruned, and it is observed that they may not recover to their pre-pruning state before they are passed to the central server. Although they typically recover in subsequent rounds, passing these models for aggregation may result in 'hiccups', a temporary degradation in the overall loss. This can be observed from the jagged loss curve in Fig. 4 compared to the smooth curves of FedAvg and the one-shot methods. We have established a generalizable algorithm that effectively leverages pruning during FL. However, there may be unexplored approaches to identifying more optimal settings for iterative pruning, such as determining the ideal granularity, schedule, and other related factors.

When we applied LTH to FedPruning, we observed that the weight reset (required to find the ticket) resulted in a significant deterioration (increase in the local loss), returning it to the initial training state. Unlike the isolated experiments by Frankle and Carbin (2018), our local models required weight sharing with each other. When local models performed weight resetting, retraining and averaging independently and frequently, it would adversely affect the aggregated weights of other clients, causing the overall loss to cease improving. We reduced the noise caused by this by changing the schedule of sparsification — nearly simultaneous pruning across all clients and less frequently. Our experiments demonstrated that applying LTH to the iterative FedPruning was arduous and less advantageous compared to the other variations, though still outperforming the FedAvg baseline. However, LTH was more effective when applied only once, as shown in the comparison between one-shot and one-shot-LT. This finding may suggest that the property of LTH reemerges when scheduling of sparsification becomes less of an issue. It invites a number of follow-up questions and may be explored empirically in future research.

**Sustainability and Inference Efficiency.** Large models should be compressed to effectively participate in FL and fit on edge devices, as they require more computations, energy consumption and carbon footprint. The efficiency of machine learning inference is dictated by memory locality — if a large model cannot be held in on-chip storage (SRAM), references need to be made to access off-chip memory (DRAM), and accessing DRAM memory (640 pJ) is significantly more energy-intensive than accessing SRAM memory (5 pJ) (Horowitz). When compared to the energy cost of 32-bit float multiplications (3.7 pJ), memory locality dominates. Sparse models with compatible hardware or framework require less computations and data movement during inference than their dense counterparts and form a step towards creating sustainable solutions in agri-food and beyond.

**Use Cases.** As our goal is to propose technological solutions to facilitate data sharing and enable the development of efficient and 'green' machine learning models at scale, as well as to potentially encourage those in agri-food sector to adopt such technologies, we provide example use cases that could benefit from such a methodology, beyond the soybean case we are considering in this paper. Especially in areas where we see data sharing in agri-food via distributed training to be most applicable. In this paper, we focused on forecasting for collaborative federations for our empirical demonstration given the accessibility to suitable open-source datasets. However, the proposed methodology is directly applicable to the other use cases. We describe two key use cases observed in the agri-food sector that we believe data sharing and distributed/collaborative training with efficient machine learning models can assist; this list is not exhaustive.

- *Strawberry yield forecasting for collaborative consortia*
  The aggregation of more data from a variety of sources and multiple farms can vastly improve the performance of machine learning models and other decision-support systems. In the agri-food sector, soft fruit growers, for instance, can be limited by their data collection processes, yet they may wish to employ decision-support frameworks to improve not only profits but also their sustainability (net-zero targets). Another aspect of this relates

to contractual agreements between growers and large retail supermarkets; over-/under- estimating the amount of produce can lead to fruit waste or fruit shortages respectively, which can have both financial and environmental repercussions for growers and the sector. Having a technical solution in place that allows the creation of federations like that explored in Durrant et al. (2021) facilitated through our proposed efficient federated learning methodology can enable multiple growers to share data in a trustworthy and transparent manner to improve their own processes and production systems towards achieving financial and environmental sustainability. A decentralized pruned model can also be deployed on edge and/or other devices for more efficient real-time inference.

- *Plant diseases and pest detection from crop images*
  One of the most devastating factors that affect yield and the quality of plants relates to plant diseases and pests. Recognizing early signs of such events is paramount towards damage limitation. Object recognition systems and remote monitoring can be useful tools that can help to identify such adverse events as early as possible, but they require lots of representative images to be used for training large-scale models. In practice, it may be unlikely that a single grower or farmer will have adequate data that can be used to train a single local model, performing well enough to be practically useful. Aggregating image datasets from various sources, including infrared cameras, depth cameras, and simple color cameras, can be transformational in developing robust plant disease and pest identification systems. This can aid in early problem recognition and helps reduce waste, thereby contributing to the financial viability of growers and farmers in the agri-food sector. Our proposed methodology can be applied to such settings and be trained with multimodal data, therefore enabling decentralized training with efficient pruned neural network models. Such a lightweight model can be deployed on edge devices for real-time decision support.

## 6. Conclusion

As machine learning models grow rapidly in size, they demand more memory and energy footprint, and make it especially challenging for FL on edge devices in agricultural settings where network and hardware capacities are even more limited. Moreover, the non-IID data adversely affects the local accuracy of clients. Our proposed solution, which involves local pruning of models followed by global aggregation addresses these challenges effectively. The key advantages include the superior local inference as a result of the localized models, reduced model sizes for more efficient deployment and reduced communication costs during training.

Our method's effectiveness was validated using various pruning policies on a real-world agri-food dataset, focusing on inference performance, communication costs, and model sizes. We found that our approach consistently outperformed the FedAvg baseline across all tested settings and years. Notably, it often matched or even marginally surpassed the centralized baseline in performance. We have repeatedly seen in literature that moderately pruned models (before reaching extreme sparsities) tend to perform better than the unpruned counterpart (Han, Pool et al., 2015; Suzuki et al., 2018), and this coincides with our finding even in a distributed setting. However, we acknowledge the need for further empirical studies across diverse datasets and models to generalize this finding. Therefore, direct future work aims to explore this behavior and the effects of different pruning policies with other open-source datasets and problem settings. Furthermore, model sparsification may be used in conjunction with other techniques and hardware to maximize the compression effects. Moving forward we aim to develop a more comprehensive pipeline for maximizing efficiency in federated learning, especially in resource-constrained environments.

## CRediT authorship contribution statement

**Andy Li:** Conceptualisation, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Visualization. **Milan Markovic:** Validation, Investigation, Funding acquisition, Supervision, Writing – review & editing. **Peter Edwards:** Resources, Funding acquisition, Supervision, Investigation, Writing – review & editing. **Georgios Leontidis:** Conceptualisation, Methodology, Investigation, Resources, Funding acquisition, Supervision, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

All datasets used in this paper are openly available via the original sources (gSSURGO, 2023; Thornton et al., 2020; USDA-NASS, 2019).

## Acknowledgments

The work described here was funded by the EPSRC 'Enhancing Agri-Food Transparent Sustainability' (EATS) project, United Kingdom (grant number: EP/V042270/1) and by a University of Aberdeen Ph.D. studentship, United Kingdom . We also thank the University of Aberdeen's HPC facility Maxwell.

## References

Aji, A. F., & Heafield, K. (2017). Sparse communication for distributed gradient descent. In *Proceedings of the 2017 conference on empirical methods in natural language processing*. Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/d17-1045.

Alhnaity, B., Kollias, S., Leontidis, G., Jiang, S., Schamp, B., & Pearson, S. (2021). An autoencoder wavelet based deep neural network with attention mechanism for multi-step prediction of plant growth. *Information Sciences*, *560*, 35–50.

Alistarh, D., Hoefler, T., Johansson, M., Khirirat, S., Konstantinov, N., & Renggli, C. (2018). The convergence of sparsified gradient methods. http://dx.doi.org/10.48550/ARXIV.1809.10505, URL https://arxiv.org/abs/1809.10505.

Amiri, M. M., Gunduz, D., Kulkarni, S. R., & Poor, H. V. (2020). Federated learning with quantized global model updates. arXiv preprint arXiv:2006.10672.

Anagnostis, A., Papageorgiou, E., & Bochtis, D. (2020). Application of artificial neural networks for natural gas consumption forecasting. *Sustainability*, [ISSN: 2071-1050] *12*(16), http://dx.doi.org/10.3390/su12166409, URL https://www.mdpi.com/2071-1050/12/16/6409.

Blalock, D., Ortiz, J. J. G., Frankle, J., & Guttag, J. (2020). What is the state of neural network pruning?

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., et al. (2019). Towards federated learning at scale: System design. http://dx.doi.org/10.48550/ARXIV.1902.01046, URL https://arxiv.org/abs/1902.01046.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., et al. (2016). Practical secure aggregation for federated learning on user-held data.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th international conference on computational statisticsparis France, August 22-27, 2010 keynote, invited and contributed papers* (pp. 177–186). Springer.

Caldas, S., Konečny, J., McMahan, H. B., & Talwalkar, A. (2018). Expanding the reach of federated learning by reducing client resource requirements. arXiv preprint arXiv:1812.07210.

Dayan, I., Roth, H. R., Zhong, A., Harouni, A., Gentili, A., Abidin, A. Z., et al. (2021). Federated learning for predicting clinical outcomes in patients with COVID-19. *Nature Medicine*, *27*(10), 1735–1743.

Denil, M., Shakibi, B., Dinh, L., Ranzato, M., & de Freitas, N. (2014). Predicting parameters in deep learning.

Dinh, C. T., Tran, N. H., & Nguyen, T. D. (2022). Personalized federated learning with moreau envelopes.

Durrant, A., Markovic, M., Matthews, D., May, D., Leontidis, G., & Enright, J. (2021). How might technology rise to the challenge of data sharing in agri-food? *Global Food Security*, [ISSN: 2211-9124] *28*, Article 100493. http://dx.doi.org/10.1016/j.gfs.2021.100493, URL https://www.sciencedirect.com/science/article/pii/S2211912421000031.

Frankle, J., & Carbin, M. (2018). The lottery ticket hypothesis: Finding sparse, trainable neural networks. http://dx.doi.org/10.48550/ARXIV.1803.03635, URL https://arxiv.org/abs/1803.03635.

gSSURGO (2023). Gridded soil survey geographic (gssurgo) database. URL https://www.nrcs.usda.gov/resources/data-and-reports/gridded-soil-survey-geographic-gssurgo-database.

Hamer, J., Mohri, M., & Suresh, A. T. (2020). Fedboost: A communication-efficient algorithm for federated learning. In *International conference on machine learning* (pp. 3973–3983). PMLR.

Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. http://dx.doi.org/10.48550/ARXIV.1510.00149, URL https://arxiv.org/abs/1510.00149.

Han, S., Pool, J., Tran, J., & Dally, W. J. (2015). Learning both weights and connections for efficient neural networks. http://dx.doi.org/10.48550/ARXIV.1506.02626, URL https://arxiv.org/abs/1506.02626.

Hanzely, F., & Richtárik, P. (2020). Federated learning of a mixture of global and local models. CoRR, abs/2002.05516. URL https://arxiv.org/abs/2002.05516.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(9), 1263–1284. http://dx.doi.org/10.1109/TKDE.2008.239.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. http://dx.doi.org/10.48550/ARXIV.1502.01852, URL https://arxiv.org/abs/1502.01852.

Horowitz, M. Energy table for 45 nm process, Stanford VLSI wiki.

Huang, L., Shea, A. L., Qian, H., Masurkar, A., Deng, H., & Liu, D. (2019). Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. *Journal of Biomedical Informatics*, *99*, Article 103291.

Imteaj, A., & Amini, M. H. (2022). Leveraging asynchronous federated learning to predict customers financial distress. *Intelligent Systems with Applications*, *14*, Article 200064.

Imteaj, A., Thakker, U., Wang, S., Li, J., & Amini, M. H. (2021). A survey on federated learning for resource-constrained IoT devices. *IEEE Internet of Things Journal*, *9*(1), 1–24.

Jeong, E., Oh, S., Kim, H., Park, J., Bennis, M., & Kim, S.-L. (2018). Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. arXiv preprint arXiv:1811.11479.

Jeong, J. H., Resop, J. P., Mueller, N. D., Fleisher, D. H., Yun, K., Butler, E. E., et al. (2016). Random forests for global and regional crop yield predictions. *PLoS One*, *11*(6), Article e0156571.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., & Suresh, A. T. (2019). SCAFFOLD: Stochastic controlled averaging for federated learning. http://dx.doi.org/10.48550/ARXIV.1910.06378, URL https://arxiv.org/abs/1910.06378.

Khaki, S., Wang, L., & Archontoulis, S. V. (2020). A CNN-RNN framework for crop yield prediction. *Frontiers in Plant Science*, *10*, http://dx.doi.org/10.3389/fpls.2019.01750.

Khan, L. U., Saad, W., Han, Z., Hossain, E., & Hong, C. S. (2021). Federated learning for internet of things: Recent advances, taxonomy, and open challenges. *IEEE Communications Surveys & Tutorials*, *23*(3), 1759–1799.

Konečnỳ, J., McMahan, H. B., Ramage, D., & Richtárik, P. (2016). Federated optimization: Distributed machine learning for on-device intelligence. arXiv preprint arXiv:1610.02527.

Kussul, N., Lavreniuk, M., Skakun, S., & Shelestov, A. (2017). Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, *14*(5), 778–782. http://dx.doi.org/10.1109/LGRS.2017.2681128.

LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. In *Advances in neural information processing systems, Vol. 2*.

Lee, N., Ajanthan, T., & Torr, P. H. S. (2019). SNIP: Single-shot network pruning based on connection sensitivity.

Li, H., & Han, T. (2019). An end-to-end encrypted neural network for gradient updates transmission in federated learning. arXiv preprint arXiv:1908.08340.

Li, Q., He, B., & Song, D. (2021). Model-contrastive federated learning.

Li, X., Jiang, M., Zhang, X., Kamp, M., & Dou, Q. (2021). Fedbn: Federated learning on non-iid features via local batch normalization. arXiv preprint arXiv:2102.07623.

Li, T., Sahu, A. K., Talwalkar, A., & Smith, V. (2020a). Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, *37*(3), 50–60.

Li, T., Sahu, A. K., Talwalkar, A., & Smith, V. (2020b). Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, *37*(3), 50–60. http://dx.doi.org/10.1109/msp.2020.2975749.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., & Smith, V. (2018). Federated optimization in heterogeneous networks. http://dx.doi.org/10.48550/ARXIV.1812.06127, URL https://arxiv.org/abs/1812.06127.

Li, A., Sun, J., Wang, B., Duan, L., Li, S., Chen, Y., et al. (2021). Lotteryfl: Empower edge intelligence with personalized and communication-efficient federated learning. In *2021 IEEE/ACM symposium on edge computing (SEC)* (pp. 68–79). http://dx.doi.org/10.1145/3453142.3492909.

Liang, T., Glossner, J., Wang, L., Shi, S., & Zhang, X. (2021). Pruning and quantization for deep neural network acceleration: A survey. http://dx.doi.org/10.48550/ARXIV.2101.09671, URL https://arxiv.org/abs/2101.09671.

Long, G., Tan, Y., Jiang, J., & Zhang, C. (2020). Federated learning for open banking. In *Federated learning: Privacy and incentive* (pp. 240–254). Springer.

Ma, Z., Zhang, M., Liu, J., Yang, A., Li, H., Wang, J., et al. (2022). An assisted diagnosis model for cancer patients based on federated learning. *Frontiers in Oncology*, *12*, Article 860532.

McMahan, H. B., Moore, E., Ramage, D., & y Arcas, B. A. (2016). Communication-efficient learning of deep networks from decentralized data. CoRR, abs/1602.05629. URL http://arxiv.org/abs/1602.05629.

Menghani, G. (2023). Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *ACM Computing Surveys*, *55*(12), 1–37.

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, [ISSN: 1664-462X] *7*, http://dx.doi.org/10.3389/fpls.2016.01419, URL https://www.frontiersin.org/articles/10.3389/fpls.2016.01419.

Nguyen, D. C., Pham, Q.-V., Pathirana, P. N., Ding, M., Seneviratne, A., Lin, Z., et al. (2022). Federated learning for smart healthcare: A survey. *ACM Computing Surveys*, *55*(3), 1–37.

Onoufriou, G., Bickerton, R., Pearson, S., & Leontidis, G. (2019). Nemesyst: A hybrid parallelism deep learning-based framework applied for internet of things enabled food retailing refrigeration systems. *Computers in Industry*, *113*, Article 103133.

Onoufriou, G., Hanheide, M., & Leontidis, G. (2023). Premonition net, a multi-timeline transformer network architecture towards strawberry tabletop yield forecasting. *Computers and Electronics in Agriculture*, *208*, Article 107784.

Paragliola, G., & Coronato, A. (2022). Definition of a novel federated learning approach to reduce communication costs. *Expert Systems with Applications*, *189*, Article 116109.

Pfohl, S. R., Dai, A. M., & Heller, K. (2019). Federated and differentially private learning for electronic health records. arXiv preprint arXiv:1911.05861.

Prakash, P., Ding, J., Chen, R., Qin, X., Shu, M., Cui, Q., et al. (2022). IoT device friendly and communication-efficient federated learning via joint model pruning and quantization. *IEEE Internet of Things Journal*, *9*(15), 13638–13650. http://dx.doi.org/10.1109/JIOT.2022.3145865.

Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., & Huang, X. (2020). Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, *63*(10), 1872–1897.

Rakhlin, A., Shamir, O., & Sridharan, K. (2012). Making gradient descent optimal for strongly convex stochastic optimization.

Rasmussen, C., & Ghahramani, Z. (2000). Occam's razor. In *Advances in neural information processing systems, Vol. 13*.

Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A., & Pedarsani, R. (2020). Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *International conference on artificial intelligence and statistics* (pp. 2021–2031). PMLR.

Rieke, N., Hancox, J., Li, W., Milletari, F., Roth, H. R., Albarqouni, S., et al. (2020). The future of digital health with federated learning. *NPJ Digital Medicine*, *3*(1), 119.

Rong, D., Xie, L., & Ying, Y. (2019). Computer vision detection of foreign objects in walnuts using deep learning. *Computers and Electronics in Agriculture*, *162*, 1001–1010.

Ryu, G.-A., Nasridinov, A., Rah, H., & Yoo, K.-H. (2020). Forecasts of the amount purchase pork meat by using structured and unstructured big data. *Agriculture*, [ISSN: 2077-0472] *10*(1), http://dx.doi.org/10.3390/agriculture10010021, URL https://www.mdpi.com/2077-0472/10/1/21.

Santurkar, S., Tsipras, D., Ilyas, A., & Madry, A. (2018). How does batch normalization help optimization? http://dx.doi.org/10.48550/ARXIV.1805.11604, URL https://arxiv.org/abs/1805.11604.

Shi, H., & Radu, V. (2021). Towards federated learning with attention transfer to mitigate system and data heterogeneity of clients. In *Proceedings of the 4th international workshop on edge systems, analytics and networking* (pp. 61–66). New York, NY, USA: Association for Computing Machinery, ISBN: 9781450382915, http://dx.doi.org/10.1145/3434770.3459739.

Suzuki, T., Abe, H., Murata, T., Horiuchi, S., Ito, K., Wachi, T., et al. (2018). Spectral pruning: Compressing deep neural networks via spectral analysis and its generalization error. arXiv preprint arXiv:1808.08558.

Tao, Z., & Li, Q. (2018). eSGD: Commutation efficient distributed deep learning on the edge. In *HotEdge* (p. 6).

Thornton, M., Shrestha, R., Wei, Y., Thornton, P., Kao, S., & Wilson, B. (2020). Daymet: Daily surface weather data on a 1-km grid for north america, version 4. http://dx.doi.org/10.3334/ORNLDAAC/1840, URL https://daac.ornl.gov/cgi-bin/dsviewer.pl?ds_id=1840.

Thota, M., & Leontidis, G. (2021). Contrastive domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 2209–2218).

USDA-NASS (2019). Nass - national agricultural statistics service. URL https://www.nass.usda.gov/.

van Hilten, M., & Wolfert, S. (2022). 5G in agri-food - a review on current status, opportunities and challenges. *Computers and Electronics in Agriculture*, [ISSN: 0168-1699] *201*, Article 107291. http://dx.doi.org/10.1016/j.compag.2022.107291, URL https://www.sciencedirect.com/science/article/pii/S0168169922006032.

van Klompenburg, T., Kassahun, A., & Catal, C. (2020). Crop yield prediction using machine learning: A systematic literature review. *Computers and Electronics in Agriculture*, [ISSN: 0168-1699] *177*, Article 105709. http://dx.doi.org/10.1016/j.compag.2020.105709, URL https://www.sciencedirect.com/science/article/pii/S0168169920302301.

Wang, J., Liu, Q., Liang, H., Joshi, G., & Poor, H. V. (2020). Tackling the objective inconsistency problem in heterogeneous federated optimization.

Wu, C., Wu, F., Lyu, L., Huang, Y., & Xie, X. (2022). Communication-efficient federated learning via knowledge distillation. *Nature Communications*, *13*(1), http://dx.doi.org/10.1038/s41467-022-29763-x.

Xu, J., Chen, Z., Quek, T. Q., & Chong, K. F. E. (2022). Fedcorr: Multi-stage federated learning for label noise correction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10184–10193).

Xu, J., Glicksberg, B. S., Su, C., Walker, P., Bian, J., & Wang, F. (2021). Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research*, *5*, 1–19.

Yang, Q., Liu, Y., Chen, T., & Tong, Y. (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, *10*(2), 1–19.

Yang, W., Zhang, Y., Ye, K., Li, L., & Xu, C.-Z. (2019). Ffd: A federated learning based method for credit card fraud detection. In *Big data–bigdata 2019: 8th international congress, held as part of the services conference federation, SCF 2019, San Diego, CA, USA, June 25–30, 2019, proceedings 8* (pp. 18–32). Springer.

Yu, T., Li, T., Sun, Y., Nanda, S., Smith, V., Sekar, V., et al. (2020). Learning context-aware policies from multiple smart homes via federated multi-task learning. In *2020 IEEE/ACM Fifth international conference on internet-of-things design and implementation (IoTDI)* (pp. 104–115). IEEE.

Zhang, W., Li, X., Ma, H., Luo, Z., & Li, X. (2021). Federated learning for machinery fault diagnosis with dynamic validation and self-supervision. *Knowledge-Based Systems*, *213*, Article 106679.

Zhu, H., Xu, J., Liu, S., & Jin, Y. (2021). Federated learning on non-IID data: A survey. *Neurocomputing*, [ISSN: 0925-2312] *465*, 371–390. http://dx.doi.org/10.1016/j.neucom.2021.07.098, URL https://www.sciencedirect.com/science/article/pii/S0925231221013254.