# Mitigating Concept Drift in Distributed Contexts with Dynamic Repository of Federated Models

1st Elena Tsiporkova
*EluciDATA Lab of Sirris*
Brussels, Belgium
elena.tsiporkova@sirris.be

2nd Michiel De Vis
*EluciDATA Lab of Sirris*
Brussels, Belgium
michiel.devis@sirris.be

3rd Sarah Klein
*EluciDATA Lab of Sirris*
Brussels, Belgium
sarah.klein@sirris.be

4th Anna Hristoskova
*Digital Services Lab of Sirris*
Brussels, Belgium
anna.hristoskova@sirris.be

5th Veselka Boeva
*Department of Computer Science*
*Blekinge Institute of Technology*
Karlskrona, Sweden
veselka.boeva@bth.se

***Abstract*—This paper proposes a novel federated learning methodology, called FedRepo, that copes with concept drift issues in a statistically heterogeneous distributed learning environment. The proposed horizontal federated learning methodology, based on random forest (RF), can be used for collaborative training and maintenance of a dynamic repository of federated RF models, each one customized to a group of clients/devices. The clients are grouped together if their performance patterns with respect to the global RF model are similar. The performance of the customized RF global models is continuously monitored during the inference phase and the repository is accordingly adapted to mitigate the detected concept drift. The proposed methodology is studied and evaluated against an electricity consumption forecasting use case. The evaluation results demonstrate clearly that the proposed methodology is able to deal with concept drift issues in an efficient and adequate fashion without compromising the overall performance of the distributed environment.**

***Index Terms*—federated learning, concept drift, clustering, random forest, particle swarm optimization, distributed learning**

## I. Introduction

Federated learning is already widely used paradigm in very diverse distributed and privacy-sensitive application contexts [1], [2]. Distributed environments are typically characterised by very dynamic and heterogeneous in nature data sources and thus often suffer from concept drift, i.e., an evolution of the data that invalidates the data model [3]. This may lead to substantial performance degradation and needs to be dealt with via the development of dedicated concept drift detection and mitigation strategies [4]. Dealing with concept drift artifacts is however, still not widely explored area of research in federated learning (FL) context. Algorithm adaptability, particularly in the temporal dimension [5], [6] is underrepresented in the literature.

Concept drift is defined by change in the underlying distribution generating the data [3], [7]. This underlying distribution can often not be directly measured, but can be characterized as the context in which an asset is performing. In this paper, we will study how FL can be used to build and maintain robust electricity consumption forecasting models. The electricity consumption is influenced by external factors, like the seasons, which are known but also by hidden contextual factors, like the inhabitants' occupation or the replacement of household appliances that alter the overall consumption and temporal fingerprint of the consumption. Furthermore, the consumption patterns of different households are very different and therefore, training only one single model for all households might result in a too complex model to be trained on a distributed device with limited resources. Hence, in order to forecast electricity consumption in a federated fashion, model retraining needs to be performed on regular basis.

In this paper, we introduce a concept drift detection and mitigation methodology in a horizontally federated fashion based on RF regression models. In general, they are distinguished to be either vertically or horizontally federated. In the case of horizontally federated models, data sets share the same feature space but differ in data samples, whereas in the vertical scenario, the federated models are applicable to the cases that the data sets share the same sample space but differ in feature space [8]. The proposed methodology, called FedRepo, can collaboratively train and continuously adapt a repository of federated RF models, each one associated with a group of similar clients. Two clients are considered to be similar if they demonstrate similar global model performance behaviour.

Even though the FedRepo is described and evaluated in a regression task scenario, the same methodology can be used for classification by using a proper evaluation metric. The obtained evaluation results show that the FedRepo is able to handle adequately concept drift phenomena in electricity consumption forecasting tasks.

The work in this paper results from MIRAI, a project labelled by ITEA3 under project no 19034, with funding support from Innoviris Brussels, Belgium. This research also received funding from the Flemish Government (AI Research Program). Veselka Boeva's research is funded partly by the Knowledge Foundation, Sweden, through the Human-Centered Intelligent Realities (HINTS) Profile Project (contract 20220068).

The paper is organized as follows. We first describe related approaches in the field in Section II and general preliminaries and definitions in Section III, before describing our own approach in detail in Section IV. In Section V the implementation details and the results are described, before concluding in Section VI.

## II. RELATED WORK

In this section, we explore some existing approaches in the field of federated learning. An extensive overview of the current state-of-the-art in the field of distributed ML can be found in [9]. Another recent comprehensive surveys of the existing research and future research directions of FL are provided in [10], [11]. Our further discussion in this section is focused on three topics related to our work: (1) random forest based FL approaches, (2) federated load forecasting and (3) concept drift detection and mitigation in FL.

### A. Federated Random Forest

FL based on random forest is nowadays a very intensive research field due to the parallel nature of random forest models. In [12], a federated learning approach for intrusion detection based on random forest (RF) is studied. For each client, a RF model is trained on its specific data, and subsequently, all clients send their RF models to the centralized server to be combined in a global RF. Finally, the global RF is sent back to the clients. In [13], a vertical tree-based FL model, called Federated Forest, is proposed. The authors develop a secure cross-regional ML system that allows a learning process to be jointly trained over different regions' clients with the same user samples but different attribute sets.

The authors of [14] propose a new privacy-protected federated personalized RF model based on the federated forest approach of [13]. In their work, local sensitivity hashing is used for calculating the similarity between different users. Based on this similarity, a subset of the top-$k$ most similar users is selected for training the federated forest model in iterative fashion. In [15], a distributed machine learning system based on local random forest algorithms created with shared decision trees through the blockchain is introduced.

### B. Federated load forecasting

With the implementation of smart grid applications, electricity consumption forecasting became more and more important. In [16], Fernandez *et al.* compare the performance of different FL architectures combined with different privacy preserving techniques on the same data set as the one used for validating our approach (see Section IV-A). Similarly, in the work of Briggs *et al.* [17] different federated clustering methods for neural network based load forecasting are discussed. For both publications, the clustering is static and is not adapted over time as it is assumed that the underlying distributions will not change. This where our approach is different by dynamically maintaining (retraining) the models affected by concept drift.

### C. Concept Drift in Distributed Context

In [18], a novel FL algorithm introducing an extension of the most widely used Federated Averaging (FedAvg) algorithm [2], is proposed. The algorithm, called Adaptive-FedAvg, is able to operate with non-stationary data generating processes affected by concept drifts by varying the learning rate accordingly.

The authors in [6] used the Adaptive-FedAvg algorithm as baseline to compare against their FedDrift approach, where they consider distributed concept drift in a sense that the time of drift can be different in different devices as well as the source and target distributions can differ across clients. They identify this distributed concept drift by evaluating newly arrived data in the distributed devices against a set of federated models locally and then performing a hierarchical clustering on the losses in order to re-cluster the federated devices.

In [5], the authors introduce another extension of the FedAvg algorithm to be able to learn under concept drift. The algorithm is extended to deal with continual single-task problems, i.e. all the client devices share the same goal, but the underlying joint distribution of data might be non-IID (independent and identically distributed) and can change over time.

In [19], the problem of drift adaptation is identified as a time-varying clustering problem. The authors propose two new clustering algorithms for reacting to drifts based on local drift detection and hierarchical clustering.

Notice that most of the above mentioned FL solutions are aimed at the concept drift detection and mitigation during the training phase. Our work instead introduces a FL methodology that is able to deal efficiently with concept drift issues at the inference phase.

## III. PRELIMINARIES AND DEFINITIONS

In this section, we first briefly describe some preliminaries of a baseline FL method [1]. Further, we introduce the different methods used in our concept-drift mitigation methodology, namely Random Forest models, Particle Swarm Optimisation (PSO) and its binary extension. Finally, we define the problem which we attempt to tackle in this work.

### A. Baseline FL algorithm

Federated learning or collaborative learning aims to leverage information in a distributed system in the learning procedure while still keeping the data distributed across the devices in the system [1], [2]. The general idea is to transfer model information instead of the actual data samples collected on the devices. With this approach, it is possible to train a common model even on privacy-sensitive data, e.g., on usage data, without violating privacy regulations. In the general approach the aim is to train a unified model, that performs equally well on each client's private data. However, the heterogeneity of data across devices can lead to degraded performance of traditionally built FL models. Therefore, for increased computational efficiency and predicting power in the context of heterogeneous devices it is useful to personalize the derived

global model [20]. Most of these approaches are based on gradient-based deep-learning approaches [20], [21], [22].

### B. Random Forest

For supervised regression and classification tasks, random forest models [23] have been widely used. A random forest model is a ensemble of decision tree predictors while for each of these trees, the split features and values are determined from random subsets of the overall data, the so-called bagging approach. In that way the single trees in the forest are decoupled and over-fitting is prevented.

### C. Particle Swarm Optimization

Particle swarm optimization (PSO) is an evolutionary computation method introduced in [24]. In order to find a (near-)optimal solution, PSO updates the current generation of particles (each particle being a candidate solution) using the information about the best solution obtained by each particle and the entire population. Each particle is a point in an $n$-dimensional space. The $i$-th particle is initialized with random positions $X_i$ and velocities $V_i$ at time point $t = 0$. The basic update equations for the $d$-th dimension of the $i$-th particle are

$$v_{id}(t+1) = w \cdot v_{id}(t) + c_1 \cdot \varphi_1 \cdot (p_{id} - x_{id}(t)) + c_2 \cdot \varphi_2 \cdot (p_{gd} - x_{id}(t)) \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1). \quad (2)$$

The variables $\varphi_1$ and $\varphi_2$ are uniformly generated random numbers in the range $[0, 1]$, $c_1$ and $c_2$ are acceleration constants, $w$ is the inertia weight [25]. $P_g$ is the best particle position found so far within the population and $P_i$ is the best position discovered so far by the corresponding particle. The first part of equation (1) represents the inertia of the previous velocity, the second is the *personal experience of the particle*, the third part represents the *cooperation among particles*.

In [26], Kennedy *et al.* describe an adapted version of the classical PSO algorithm, called *binary PSO*, in which the components of $x_{ij} \in X_i, j = 1, ..., n$ are restricted to the set $\{0, 1\}$. In the binary PSO, the velocity defines the probability that the particle position component $x_{ij}$ is switched from 1 to 0 and vice versa. In [26], the authors suggest to use the sigmoid function $sig(x) = (1 + \exp(x))^{-1}$ for calculating the velocity.

In the standard approach of PSO, the number of clusters is constant and needs to be predefined. For most real-world problems though, it is impossible to determine the number of clusters upfront and the assumption that this number is constant for all times is not reasonable.

In this paper, we make use of the approach described by Omar *et al.* [27] that dynamically determines the best number of clusters. In that approach, a subset of data vectors is randomly selected as cluster centroids. The length of this subset is defined by the maximum number of clusters, which is a parameter to set. This parameter also sets the dimension of the particle's lengths. The actual position $x_{ij}$ of particle $i$ equals 1 if cluster $C_j$, determined by the cluster centroid corresponding to this position, is part of the solution. Every particle thus proposes a clustering solution.

The binary approach is then applied multiple times as in the original PSO approach (cf. III-C), converging every time to an optimal clustering solution. For this, a cluster evaluation metric is used. In between two iterations, the cluster centroids which are not part of the optimal solution are removed and replacements are randomly chosen from the data. Two stopping criteria should be defined: one for the binary PSO itself, and another for the amount of times it should be applied. A patience value is set for both in order to make use of early stopping in case of convergence.

### D. Problem Definition

This work is focusing on the detection and mitigation of concept drift in a distributed setup in a privacy-preserving and scalable fashion. Different devices collect the same kind of data while their distributions are highly heterogeneous and might vary over time. In contrast to the traditional supervised FL setting, the here proposed concept drift mitigation strategy does not have as a goal to find a stationary global model that performs well for all the workers (devices), but aims at training and continuously adapting a repository of global models, one per a group of similar devices, instead. The main challenge in this setup is how to dynamically maintain this repository in order to ensure that each global federated model performs well for each device in the cluster associated with this model.

The proposed method is able to identify similar devices in a dynamic manner without sharing the devices' data with one another and storing it on a central device.

## IV. MATERIALS AND METHODS

### A. Data and Use Case

Our approach is evaluated against the regression problem of forecasting the electricity consumption of a heterogeneous set of households. Electricity consumption data contains a lot of private information and can be used for classifying households into their socio-economic situation [28]. For this reason, a central collection of electricity consumption data with a comparably high temporal resolution (in the study of [28] a resolution of 30 minutes was used) is not favorable but instead, a federated approach should be used for gathering insights.

In this paper, the data collected by the UK Power Networks led Low Carbon London project [29] is used. It consists of 5,567 households in London representing a balanced sample representative of the Greater London population with a 30-minutes granularity between November 2011 and February 2014. For validating our methodology, we have randomly selected 300 households for which we have ensured that the data is available until at least 01/2014. For these households, a repository of federated models will be trained in order to forecast the consumption within the next 30 minutes.

### B. Concept Drift Detection and Mitigation Methodology

In this section, we consider a set of $M$ distributed devices (referred below as clients/workers) for which a repository of federated RF models will be trained. This repository will be dynamically adapted in order to best reflect the changing local operating context of the workers.

Our methodology is actually built around the dynamic maintenance and adaptation of three repositories residing in a central node, e.g., in the cloud:

- $\Theta$ is a repository of workers, which contains at any moment in time the workers for which new federated models need to be constructed.
- $\Phi$ is a repository of global federated RF models, which contains at any moment in time the active (deployed) federated models. More concretely, for any $K < M$:

$$\Phi = \{(\Phi_k, \Theta_k, s_k)| \quad k = 0, \ldots, K\},$$

  where $\Theta_k$ contains the workers for which the RF federated model $\Phi_k$ is active and $s_k = \#\Theta_k/M$ is the model support, i.e., it expresses the relative spread of the model among the whole population of workers.
- $\Gamma$ is a repository of tree models, which aggregates at any moment in time subsets of tree models $\Gamma_i$ (for $i = 1, \ldots, M$), one subset per worker, i.e. $\Gamma = \{\Gamma_1, \ldots, \Gamma_M\}$. $\Gamma_i$ is a subset of randomly selected trees from the local RF models of each worker $\theta_i$ and supposed to be used for the construction/update of the global federated model.

The proposed concept drift detection and mitigation methodology, referred to as FedRepo, will continuously update the above described repositories during the use of the federated models based on continuous monitoring and evaluation of the models' performance. The FedRepo consists of four main steps: initialization, model training, context-aware inference and dynamic model maintenance. These are elaborated below.

1) *Initialisation*: This step is performed in the central node. The repository of federated RF models is empty since no RF models have been constructed yet, i.e., $\Phi = \emptyset$. Analogously, the workers' repository contains all available workers since for all of them the federated models still need to be constructed, i.e., $\Theta = \{\theta_1, \ldots, \theta_M\}$ and the repository of tree models is composed of $M$ empty sets, one per worker, i.e., $\Gamma = \{\Gamma_1, \ldots, \Gamma_M\}$, where $\Gamma_i = \emptyset$, for $i = 1, \ldots, M$.
2) *Training*: In this step, the model and worker repositories are updated such that devices similar with respect to the model performance are assigned to the same cluster of workers and hence collaboratively build and share the same RF federated model. For this, as described in Algorithm 1, the following steps are executed:
   a) *Local Model Training*: Each worker in $\Theta$ trains its local RF model on a pre-defined part of its data (train set). Another part (test set) is left aside to be used later for the evaluation of the global model. Subsequently, as a result of the training phase, $\#\Theta$ different RF local models are generated, one per worker and each one composed of the same number of $P$ trees. The workflow is described in Algorithm 2.
   b) *Tree repository update*: Each worker in $\Theta$ selects randomly a portion of $Q$ trees from its local RF model and sends them to the central node. The central tree repository $\Gamma$ is updated by replacing the sets of trees donated previously by the workers in $\Theta$ ($Q$ per worker) with the newly selected trees. Note that $Q$ must be at least $P/M$ in case $P > M$ and at least 1 in case $P \leq M$ or formally $Q \geq 1 + P/M$ (cf. Algorithm 3).
   c) *Federated Global Model Construction/Update*: From the set of trees consisting of all $\Gamma_i$ (for $i = 1, \ldots, M$) in $\Gamma$, $P$ trees are randomly sampled and aggregated to construct the global federated model. This global model is added to the model repository $\Phi$, i.e., $\Phi = \Phi \cup \{(\Phi_0, \Theta, s_0)\}$, and $s_0 = \#\Theta/M$ as shown in Algorithm 4. The overall model is used for clustering the workers (see below), for benchmark and for managing cold start cases.
   d) *Evaluation Feature Vector Construction*: The global federated RF model $\Phi_0$ is applied to each worker's private test data (cf. Algorithms 1 and 5). In this way, a $P$-dimensional evaluation feature vector $\xi_k = (\xi_k^1, \ldots, \xi_k^P)$, i.e., one evaluation score per tree, is derived for each worker $\theta_k$. The evaluation vectors are sent back to the central node, where they are stacked in a matrix $\Xi = \bigcup_{k=1,\ldots,M} \xi_k$, needed as an input for the PSO algorithm. The evaluation metric deployed depends on the specific task and can be chosen accordingly. One logical choice is the error rate score of each individual tree of $\Phi_0$ for each worker.

**Algorithm 1** Training - Central node

**Require:** $\Theta, \Phi, \Gamma$
1: $\Gamma \leftarrow TreeRepositoryUpdate(\Theta, \Gamma)$
2: $\Phi_0, s_0 \leftarrow FederatedModelConstruction(\Theta)$
3: $\Xi \leftarrow \{\}$
4: **for all** $\theta \in \Theta$ **do**
5: $\quad \xi_\theta \leftarrow EvaluationFeatureVectorConstruction(\Phi_0)$
6: $\quad \Xi \leftarrow \Xi \cup \xi_\theta$
7: **end for**
8: $\mathcal{C} \leftarrow$ binary PSO ($\Xi$) (cf. [26])
9: **for all** $c \in \mathcal{C}$ **do**
10: $\quad \Theta_c \leftarrow \{\theta_i \in c\}$
11: $\quad \Phi_c, s_c \leftarrow FederatedModelConstruction(\Theta_c)$
12: $\quad \Phi \leftarrow \Phi \cup \{(\Phi_c, \Theta_c, s_c)\}$
13: $\quad \Theta = \Theta - \Theta_c$
14: **end for**
15: $s_0 = 0$
16: **return** $\Phi$

**Algorithm 2** Local Model Training - Worker

**Require:** $P, Q$
1: rf $\leftarrow RandomForest(P)$
2: rf.train($\mathcal{D}^k_{train}$)
3: $T$ = random.sample($\{t \in$ rf$\}$) with $\#T = Q$
4: **return** $T$

**Algorithm 3** Tree Repository Update - Central node

**Require:** $\Theta_k$, $\Gamma$
1: **for** $\theta_k \in \Theta_k$ **do**
2: $\quad\Gamma_k \leftarrow \{\}$
3: $\quad\Gamma_k \leftarrow Local\ Model\ Training - Worker(P, Q)$
4: $\quad\Gamma \leftarrow \Gamma \cup \Gamma_k$
5: **end for**
6: **return** $\Gamma$

e) *Local Node Clustering*: In order to derive personalized models for a set of similar workers, the workers are subsequently split into $K$ non-overlapping clusters. These are obtained by applying the binary PSO algorithm on the evaluation feature vectors calculated in the previous step. The binary PSO clustering has the advantage that the number of clusters does not need to be predefined.

f) *Federated Cluster Models Construction*: For each cluster $k = 1, \ldots, K$, a federated RF model $\Phi_k$ is built following the procedure described in step (c) above ("Federated Model Construction"). Hence, the trees contributed by all private workers in $\Theta_k$, i.e., the trees contained in the respective $\Gamma_i$ ($\theta_i \in \Theta_k$) are pooled together and reshuffled. Subsequently, $P$ trees are randomly sampled to create the federated RF model $\Phi_k$. This model is associated with an initial support score $s_k = \#\Theta_k/M$.

g) *Repository Update*: The repository of federated RF models $\Phi$ is extended by adding the newly obtained federated cluster (customized) models, i.e. $\Phi = \Phi \cup \{(\Phi_k, \Theta_k, s_k)\}$, for $k = 1, \ldots, K$. The repository of workers is reset, i.e., $\Theta = \emptyset$. The support of the global federated RF model $\Phi_0$ is also reset to zero, i.e., $s_0 = 0$.

**Algorithm 4** Federated Model Construction - Central node

**Require:** $\Theta_k$
1: $\phi \leftarrow \{\}$
2: **for** $\theta_k \in \Theta_k$ **do**
3: $\quad\phi \leftarrow \phi \cup \Gamma_i$
4: **end for**
5: $\Phi_k \leftarrow$ random.sample($t \in \phi$) with $\#t = P$
6: $s_k \leftarrow \#\Theta_k/M$
7: **return** $\Phi_k, s_k$

**Algorithm 5** Evaluation Feature Vector Construction - Worker

**Require:** $\Phi_0$
1: **for** $p$ in $\Phi_0$ **do**
2: $\quad y' \leftarrow \Phi_0[p]$.predict($\mathcal{D}^k_{test}$)
3: $\quad\xi \leftarrow$ evaluate$(y, y')$ $\forall y \in \mathcal{D}^k_{test}$
4: **end for**
5: **return** $\xi$

**Algorithm 6** Context-aware Inference - Central node

**Require:** $\Phi$
1: **for all** $\{(\Phi_k, \Theta_k, s_k)\} \in \Phi$ **do**
2: $\quad$**for all** $\theta_i in\ \Theta_k$ **do**
3: $\quad\quad e, y \leftarrow Inference(\Phi_k)$
4: $\quad\quad$**if** $e$ is True **then**
5: $\quad\quad\quad\Theta \leftarrow \Theta \cup \theta_i$
6: $\quad\quad\quad\Theta_k \leftarrow \Theta_k - \theta_i$
7: $\quad\quad\quad s_0 \leftarrow s_0 + 1/M$
8: $\quad\quad\quad s_k \leftarrow s_k - 1/M$
9: $\quad\quad$**end if**
10: $\quad$**end for**
11: **end for**
12: **return** $\Phi$

3) *Context-aware Inference*: Each worker $\theta_i$ ($i = 1, \ldots, M$) receives the parameters of its cluster federated model. At each inference step, each worker calculates the residual between the predicted and observed values for the previous inference step as it is shown in Algorithms 6 and 8. If the worker's residual is above a threshold $\delta$, that is determined by the model's performance on the test set (cf. Local Model Training), this information is communicated to the central node and the following steps are conducted:

a) *Global Model Activation*: The overall federated global model $\Phi_0$ is activated for the worker in question, i.e., $\theta_i$ is added to $\Theta$ and the support of $\Phi_0$ is updated accordingly, i.e., $s_0 = s_0 + 1/M$.

b) *Model Parameter Update*: The parameters of the corresponding (cluster) federated model $(\Phi_k, \Theta_k, s_k)$ are updated, i.e., $\theta_i$ is removed from the list of private workers $\Theta_k$ for this model and the model support is reduced accordingly such that $s_k = s_k - 1/M$. Each time the parameters of a federated model in the repository $\Phi$ are updated the dynamic model maintenance needs to be executed.

4) *Dynamic Model Maintenance*: This final step concerns the identification of federated models in $\Phi$ with relatively low support, possibly due to concept drift (cf. Algorithm 7). Subsequently, a trace of the associated workers is kept in $\Theta$ and if this grows above a certain predefined volume $\Delta$, training for the affected workers is invoked.

a) *Workers' Repository*: For each federated cluster

**Algorithm 7** Dynamic model maintenance - Central node

**Require:** $\Phi_k$
**for** $\{(\Phi_k, \Theta_k, s_k)\} \in \Phi$ **do**
  **if** $s_k < z$ **then**
    $\Theta \leftarrow \Theta \cup \Theta_k$
    $s_0 \leftarrow s_o + \#\Theta_k/M$
    $s_k \leftarrow 0$
  **end if**
**end for**
**if** $s_0 > \Delta$ **then**
  **for** $\{(\Phi_k, \Theta_k, s_k)\} \in \Phi$ **do**
    $\Phi = \Phi - (\Phi_k, \Theta_k, s_k | s_k < z)$
  **end for**
**end if**
$Training - Central\ Node(\Theta, \Phi)$
$Federated\ global\ model\ update - Central\ Node(\Theta)$
**return** $\Phi$

**Algorithm 8** Inference - Worker

**Require:** $\Phi_k$
$y' \leftarrow \Phi_k$.predict$(\mathcal{D}^k_{test})$
$\rho' \leftarrow |y - y'|$
$\delta \leftarrow \mu(\rho') + 3\sigma(\rho')\ \forall y \in \mathcal{D}^k_{test}$
$y'' \leftarrow \Phi_k$.predict$(\mathcal{D}^k_{t-1})$ with $t-1$ being the timestamp of the last measurement
$\rho \leftarrow |y - y''|\ \forall y \in \mathcal{D}^k_{t-1}$
**return** $\rho > \delta, y''$

model $(\Phi_k, \Theta_k, s_k) \in \Phi$ $(k = 1, \ldots, K)$, if its support $s_k$ is smaller than a predefined threshold $z$ its workers are added to the workers' repository, i.e., if $s_k \leq z$ then $\Theta = \Theta \cup \Theta_k$.

b) *Cluster Model De-Activation*: The federated cluster model $\Phi_k$ is de-activated by resetting its support $s_k = 0$. The overall federated global model $\Phi_0$ is activated for the workers in $\Theta_k$ and the support of $\Phi_0$ is updated accordingly, i.e., $s_0 = s_0 + \#\Theta_k/M$.

c) *Federated Models' Repository*: If the support $s_0$ of the overall federated global model $\Phi_0$ is above the predefined threshold $\Delta$, the federated models' repository is updated by pruning away all the models with low support, i.e.,

$$\Phi = \{(\Phi_k, \Theta_k, s_k) | \quad s_k > z\},\ k = 1, \ldots, K$$

Subsequently, proceed with training with the newly arrived data for the workers in $\Theta$.

The above introduced FL method has several advantages for tackling distributed concept drift. First of all, the concept drift detection comes with a very small overhead, as during the regular inference phase not only the actual forecast values are calculated, but additionally the residuals. Based on the residuals, the concept drift is detected locally, while the repositories are updated in the central node.

Furthermore, the concept drift is not assumed to happen at the same time across different workers. Instead, each worker can be part of a shared repository built on information from similar devices or activate the federated global model in case its performance is degrading before being assigned to a new repository. Third, the number of models is not predefined and constant, as the binary PSO allows a flexible number of clusters. With this, any flexible change in behaviour is captured. Finally, as the worker clustering is performed on the model performance only, it is not necessary to share the actual patterns of the workers' data with the central node.

## V. IMPLEMENTATION AND RESULTS

In this section, we describe how to use the FedRepo methodology on the example of the UK household electricity data [29] as described in Section IV-A. We explain and discuss the main steps of the approach elaborated in Section IV-B, namely Model Training, Context-aware Inference and Dynamic Model Maintenance.

### A. Model Training

In order to derive a repository of federated models from the local models of the workers for predicting the electricity consumption of the next 30 minutes, the following steps are executed. For the implementation, we use a subset of 300 households (i.e., workers).

*1) Local Model Training:* First, each worker $\theta_k$ in the repository of workers $\Theta$ with $\#\Theta = 300$ is triggered to perform a training of a local RF model with $P = 100$ trees on its predefined training data set $\mathcal{D}^k_{train}$. We use three months of data, namely January to March 2012 as training data. With these parameter settings, each worker in $\Theta$ contributes $1 + P/M$, which is 2, randomly sampled trees to its respective $\Gamma_i$ in the tree repository.

*2) Global Model Initialisation:* The global model is initialised in the central device such that only one model $\Phi_0$ is active. This global model is derived from the selected local trees of all workers by concatenating them into an overall RF model containing $P$ trees. Note, that in our case the sum of all randomly selected trees, i.e., the amount of trees in all $\Gamma_i$ combined is bigger than $P$, and therefore a random subset of $P$ trees is selected.

The global federated model $\Phi_0$ is on the one hand used for newly installed devices, for which no data is available yet in order to overcome the cold-start problem in real-world environments and on the other hand, for evaluating its performance on the workers' test data.

*3) Evaluation Vector Construction:* In order to divide the workers in $\Theta$ into different groups of workers, the model $\Phi_0$ is sent to all workers in $\Theta$ where their performances are evaluated against a dedicated test data set $\mathcal{D}^k_{test}$ for each worker $\theta_k$. The test data set contains the data from April 2012, thus one month after the training dataset.

The performance of each tree of the global model $\Phi_0$ is evaluated on each worker's test data by calculating the root mean squared error (RMSE) of the original signal and the value forecasted by that tree. The RMSE is useful for regression tasks. In case of a classification model, we can

use an alternative evaluation metric. The evaluation vector $\xi_{\mathbf{k}} = (\xi_k^1, \ldots, \xi_k^P)$, containing the RMSE for each tree on the test data set of worker $\theta_k$, is calculated. This vector is sent to the central device, which collects all vectors $\xi_{\mathbf{k}}, \forall k \in \Theta$ and stacks them into a matrix $\Xi \in \mathbb{R}^{M \times P}$. Before this matrix is used to perform the PSO clustering, the evaluation vectors are z-normalized. Figure 1 shows the distance matrix derived from $\Xi$ in an example implementation using 50 households.

The intuition behind using the evaluation vectors instead of the actual consumption is two-fold: (i) Privacy is protected by design, since no consumption data has to be transferred to the central device; (ii) Clustering the workers on their performance vectors instead consumption patterns allows to identify common behavioural patterns even though the consumption measurements themselves are different.

*4) Local Node Clustering:* The PSO is initialised with $n_{part} = 200$ particles with random positions. We set the maximum number of clusters to be 15 in the case of 300 workers. Swarm parameters are set to $w = 0.72$ and $c_1 = c_2 = 1.49$, as suggested in [30]. No velocity clamp is defined. A patience of 50 iterations is used for the binary PSO, and the re-initialisation of centroids is stopped after 3 iterations without improving the best solution. The silhouette score [31] using cosine distance is used to determine the best clustering solution.

In the case of retraining the federated model repository once the threshold number of workers in $\Theta$ exceeds $\Delta$, the clustering approach described above will only be performed on $M' \leq M$ workers in $\Theta$. At that moment, the previous four months of data are used to train and test the new local models, similar as during the initialisation: three months for training, one month for testing. In this way, the train and test sets are always of the same length in case of retraining.

*5) Federated Cluster Models Construction:* A number of customized (cluster) RF models $\Phi_k$ for $k = 1, \ldots, K$ can be derived, one for each cluster of workers. Notice that each model $\Phi_k$ contains $P$ trees, randomly sampled from the $Q = 1 + \#\Theta_k/P$ trees each worker in $\Theta_k$ contributed. Given that the number of workers is probably not equal across clusters, the weight with which each worker contributes to its federated cluster model varies for the different clusters. 11 clusters are found with different sizes ranging from 6 to 76 workers.

In Figure 2, the RMSE of the local, static cluster and overall global models for each worker on a validation data set $\mathcal{D}_{val}$ (May 2012 - December 2013) is depicted per cluster. The overall global and static cluster models are resulting from the initial training phase. Both the local and the static cluster models exhibit comparable performance. Evidently, the customized cluster models are able to capture adequately the workers' behaviour and can compete with the locally trained models.

## B. Context-aware Inference

In order to identify and mitigate concept drift, for each worker the federated model is continuously evaluated on batch data collected on the local devices. For the electricity consumption data, the performance is measured each day in a rolling fashion for a window size of three days. This value is compared against the worker-specific threshold $\delta$ defined by the $3\sigma$-threshold on the RMSE in the test set. In case the threshold is exceeded, the worker is removed from the current cluster and for that worker, the global federated model is activated. In Figure 3, an example for a degrading model is

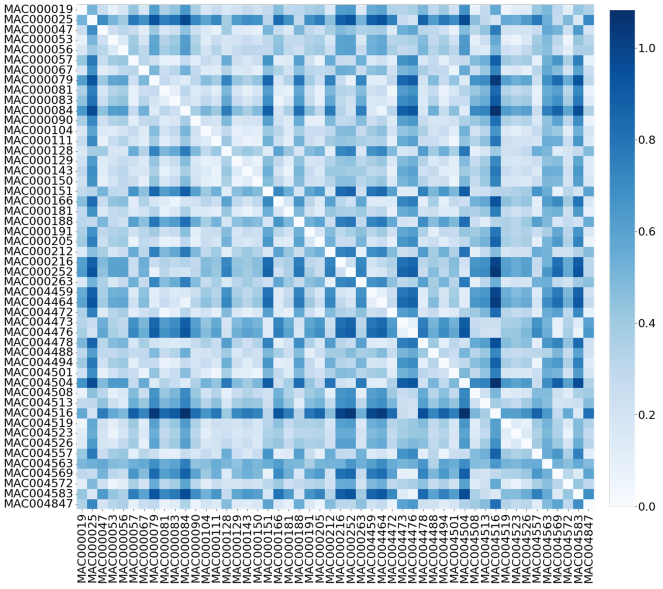

Fig. 1. An example pairwise distance matrix derived from $\Xi$ for 50 households. The color intensity indicates the cosine distance between the z-normalized performance vectors of length $P = 100$ for each pair of workers.

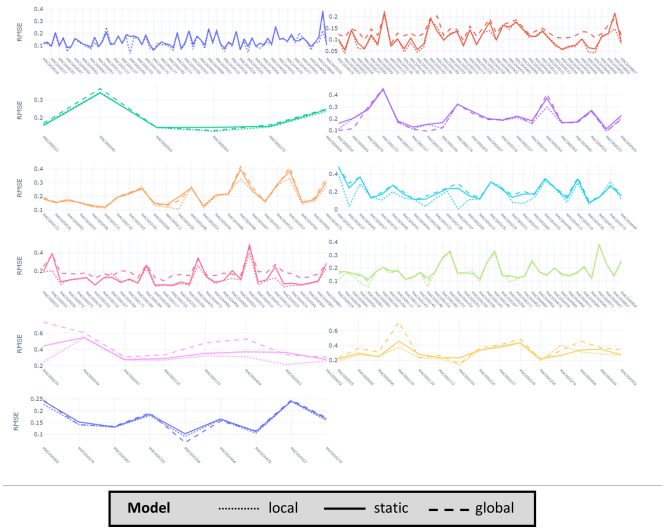


Fig. 2. RMSE on the validation data set for the locally trained model (dotted), the static cluster models (continuous) and the overall global model (dashed). The static cluster models demonstrate better performance than the overall global model almost for all workers, especially well seen in the red cluster. Evidently, the customization of the workers' models has a positive effect on the performance. In addition, the performance of the local and static models are very similar for most workers, indicating that the static cluster models represent the consumption patterns equally well.

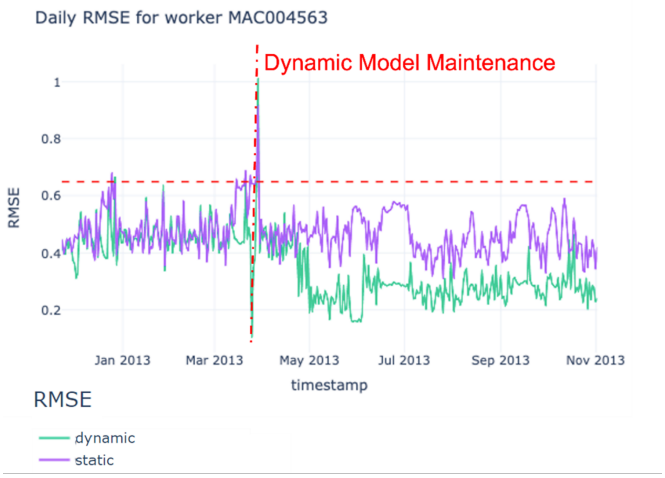


Fig. 3. Dynamic Model Maintenance for worker MAC004563 shows the improvements of the model with maintenance (green) compared to the model without maintenance/drift mitigation (purple). The dashed horizontal line gives the threshold $\delta$.

illustrated for worker MAC004563 by monitoring the RMSE value over time for the model without maintenance, i.e., static (purple) and active, i.e., dynamically updated, (green). We can clearly see how the performance of the static model degrades with time and how the daily RMSE value grows above the threshold $\delta$ in March 2013. The active model though performs better and from April 2013 on, it stays below the threshold and its RMSE is significantly lower than the one produced by the model without maintenance.

### C. Dynamic Model Maintenance

We choose the threshold $z = 0.033$ for repositories with low support to be pruned away and $\Delta = 0.2$ in $\Theta$ to start the retraining. These values were set to correspond to sensible worker amounts for both thresholds (10 and 60 workers, respectively) in the setting of 300 total workers.

Using these values, at the first model maintenance step, the workers in the smaller repositories with 6, 8 and 9 workers and hence a support of 0.02, 0.027 and 0.03, respectively, are pruned away as they are smaller than $z$. For these workers, the global model is activated and they are added to the repository $\Theta$. Note though, with these 23 workers the support of $\Theta$ is 0.077, i.e., smaller than $\Delta$. Hence, the retraining is not yet executed. In order to analyse the effect of the dynamic maintenance, we calculate the difference between the absolute error of both the static and the dynamic model. Figure 4 shows the median difference, with a positive value indicating that the error was smaller for the dynamic model. Of course, this comparison can only be made for those households that enter maintenance at some point. The figure shows that for most households undergoing maintenance, the performance increased. To demonstrate this effect, the distribution of the values in Figure 4 is shown in Figure 5. When a one-sided sign test is performed, the null hypothesis can be rejected at a significance level of 0.05, indicating that the maintenance

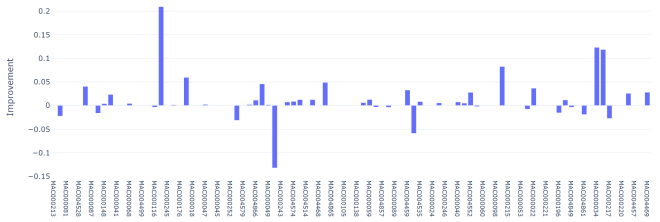


Fig. 4. The effect of dynamic maintenance for those households whose models undergo maintenance during the process. A positive value indicates that the residual was smaller for the dynamic cluster model.

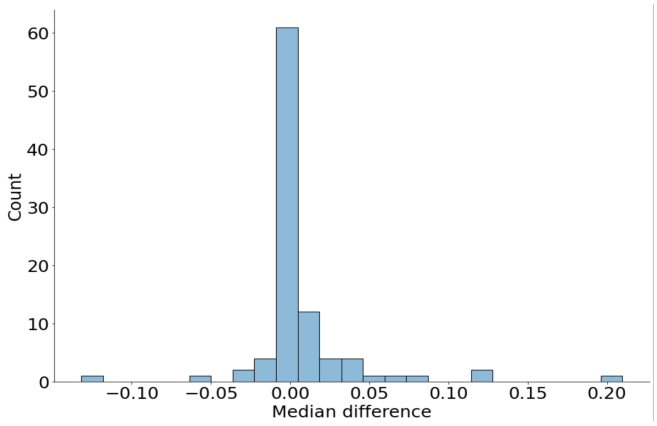


Fig. 5. The distribution of the dynamic maintenance effects seen in Figure 4. A sign test with null hypothesis $H_0 : \mu_{effect} \leq 0$ returns a $p$-value of 0.0028.

effect tends to increase the performance, i.e., the dynamic model outperforms the static model.

In addition, Figure 6 depicts the evolution of the daily median residuals (static vs. dynamic cluster models) over all households involved in maintenance across the validation data set. It is interesting to observe the very clear positive effect on the performance of the dynamic maintenance interventions (vertical lines). Note that the static and dynamic cluster models differ in performance already from the very beginning due to the fact that three workers' clusters are pruned immediately after initial training. Thus for those three clusters (23 workers) the overall global model gets activated, which results in a better performance for the dynamic cluster models already from the very beginning of the validation data set. This is confirmation that replacing poorly performing cluster models with the overall global model while waiting for re-training to happen is effective way to initiate mitigation of concept drift from the moment it is detected. The so initiated mitigation process is completed with the re-training step.

### D. Statistical Evaluation

It is necessary to demonstrate via statistical evaluation that: 1) both (static and dynamic cluster models perform consistently better than the overall global model; 2) the dynamic maintenance strategy does not degrade the overall model performance. For this purpose, the RMSE of all models

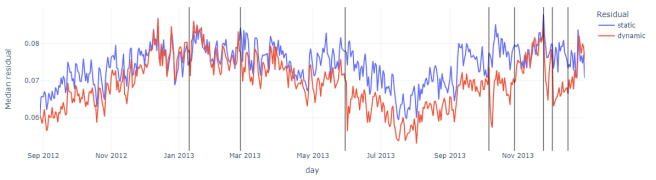


Fig. 6. The daily median residual over all households involved in maintenance across $\mathcal{D}_{val}$. Both the residuals resulting from the static and dynamic cluster models are shown. Vertical lines indicate timestamps when retraining occurred.

(overall global, static and dynamic cluster models) on $\mathcal{D}_{val}$ after the initial training phase was tracked over 100 runs. To manage computation times, a reduced set of 50 households was used for this experiment. Randomness is introduced over the runs by the training of local trees (random bagging), which influences the subsequent steps in the workflow.

Subsequently, the distributions of both cluster (static and dynamic) models vs. the overall global model and also against each other per household were compared performing 4 sign tests with null hypotheses:

1) $H_{01} : \mu_{global} \leq \mu_{static}$
2) $H_{02} : \mu_{global} \leq \mu_{dynamic}$
3) $H_{03} : \mu_{static} \leq \mu_{dynamic}$
4) $H_{04} : \mu_{dynamic} \leq \mu_{static}$

where $\mu_{global}$, $\mu_{static}$ and $\mu_{dynamic}$ are the medians of the RMSE distribution over 100 runs for the global, static and dynamically maintained model, respectively. The alternative hypothesis for each test states the opposite as the null hypothesis, e.g., $H_{a1} : \mu_{global} > \mu_{cluster}$. The tests are performed at a significance level of 0.05.

The sign test was chosen as it is a non-parametric test that does not make assumptions on the shape of the distributions [32]. For test 1, the calculated $p$-value resulted significant for 38 out of 50 households, rejecting the null hypothesis in favour of the alternative hypothesis, i.e., $\mu_{global}$ is significantly higher than $\mu_{static}$. For test 2, the $p$-values were significant for 36 households, i.e., the dynamic cluster models have significantly lower RMSE than the overall global model. For both tests 3 and 4, 8 households had a significant $p$-value, while 19 households did not need maintenance in all 100 runs. In this context, a statistically (sample set of 50 households is probably not large enough) sound comparison between the static and dynamic cluster model performance is not really possible. However, there is no evidence that the dynamic maintenance is degrading the overall performance.

### E. Computational evaluation

Our evaluation strategy is designed to study and evaluate two main characteristics of the proposed FL approach: customization and adaptability. More specifically, our algorithm is able to train a repository of federated forest models, each one customized to a group of similar devices. In addition, this repository is able to adapt dynamically to the data shift by troubleshooting and retraining worse performing models. The experiments were performed in Python by extending the Scikit-Learn framework [33] on a MacBook Pro with 16 GB memory and an Apple M2 chip. For the settings described above, the full workflow for the 300 households and 2 years of data with 30 minutes granularity takes approximately 3 hours, including 8 retraining events. The framework comes with a low communication overhead. Namely, it needs per client the following messaging: 1) triggering the training with a specific set of parameters; 2) communicating the selected trees to the central node and 3) sharing the federated cluster (customized) model from the central node to each client. This is substantially fewer than training a gradient-based model in a federated fashion, where each training round requires communication.

## VI. CONCLUSION

Concept drift detection and mitigation in FL settings is not sufficiently explored problematics in the scientific literature. In this paper, we contribute to this research area by introducing a FL methodology, FedRepo, that is capable to detect concept drift during model inference phase and additionally provide with an efficient mitigation procedure by maintaining a dynamic repository of customized FL models. The federated models are constructed in a privacy-by-design fashion and in addition, each model is built and customized to a cluster of similar clients. The performance of the built federated models is continuously monitored and they are adapted accordingly to the identified shift in the clients data. The proposed methodology is evaluated on an electricity consumption forecasting task. The experimental results have shown that the dynamic model maintenance indeed leads to improved results in case of detected concept drift.

Our future plans are aimed at studying and further evaluation of the proposed FedRepo methodology performance in other applied distributed scenarios. The two main characteristics of the proposed methodology, customization and adaptability, will be further explored and benchmarked to the similar properties proposed by other concept-drift robust federated learning strategies.

## NOMENCLATURE

| | |
|---|---|
| $\Delta$ | Worker repository threshold |
| $\Gamma$ | Repository of tree models |
| $\mathcal{D}_{test}$ | Test set |
| $\mathcal{D}_{train}$ | Train set |
| $\mathcal{D}_{val}$ | Validation set |
| $\Phi$ | Repository of federated models |
| $\Phi_0$ | Global federated model |
| $\Phi_k$ | Federated model of cluster $k$ |
| $\Theta$ | Repository of workers |
| $\Theta_k$ | Workers contained in cluster $k$ |
| $\Xi$ | Collection of evaluation vectors per worker |
| $\xi$ | $P$-dimensional evaluation vector |
| $K$ | Number of deployed federated models |
| $M$ | Total number of workers |

| | |
|---|---|
| $P$ | Number of trees per model |
| $s_k$ | Model support of cluster $k$ |
| $z$ | Model support threshold |

## REFERENCES


[1] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *ArXiv*, vol. abs/1610.05492, 2016.

[2] H. B. McMahan *et al.*, "Federated learning of deep networks using model averaging," *arXiv preprint arXiv:1602.05629*, 2016.

[3] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2346–2363, 2019.

[4] J. a. Gama, I. Žliobaitundefined, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys (CSUR)*, vol. 46, no. 4, 2014.

[5] F. Casado, D. Lema, M. Criado *et al.*, "Concept drift detection and adaptation for federated and continual learning," *Multimedia Tools and Applications*, vol. 81, p. 3397–3419, 2022.

[6] E. Jothimurugesan, K. Hsieh, J. Wang, G. Joshi, and P. B. Gibbons, "Federated learning under distributed concept drift," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, April 2023, selected as Oral Presentation (Top 2% Submissions). [Online]. Available: https://www.microsoft.com/en-us/research/publication/federated-learning-under-distributed-concept-drif/

[7] T. Lesort, V. Lomonaco, A. Stoian, D. Maltoni, D. Filliat, and N. Díaz-Rodríguez, "Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges," *Information Fusion*, vol. 58, pp. 52–68, 2020.

[8] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, 2019.

[9] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *Acm computing surveys (csur)*, vol. 53, no. 2, 2020.

[10] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowledge-Based Systems*, vol. 216, p. 106775, 2021.

[11] J. Wen, Z. Zhang, Y. Lan *et al.*, "A survey on federated learning: challenges and applications," *Int. J. Mach. Learn. & Cyber.*, vol. 14, p. 513–53, 2023.

[12] T. Markovic, M. Leon, D. Buffoni, and S. Punnekkat, "Random forest based on federated learning for intrusion detection," in *Artificial Intelligence Applications and Innovations*, I. Maglogiannis, L. Iliadis, J. Macintyre, and P. Cortez, Eds. Cham: Springer International Publishing, 2022, pp. 132–144.

[13] Y. Liu, Y. Liu, Z. Liu, Y. Liang, C. Meng, J. Zhang, and Y. Zheng, "Federated forest," *IEEE Transactions on Big Data*, vol. 8, no. 03, pp. 843–854, 2022.

[14] W. Z. Songfeng Liu, Jinyan Wang, "Federated personalized random forest for human activity recognition[j]," *Mathematical Biosciences and Engineering*, vol. 19, no. 1, pp. 953–971, 2022.

[15] L. A. C. de Souza, G. Antonio F. Rebello, G. F. Camilo, L. C. B. Guimarães, and O. C. M. B. Duarte, "Dfedforest: Decentralized federated forest," in *2020 IEEE International Conference on Blockchain (Blockchain)*, 2020, pp. 90–97.

[16] J. D. Fernández, S. P. Menci, C. M. Lee, A. Rieger, and G. Fridgen, "Privacy-preserving federated learning for residential short-term load forecasting," *Applied Energy*, vol. 326, p. 119915, 2022.

[17] C. Briggs, Z. Fan, and P. Andras, "Federated learning for short-term residential load forecasting," *IEEE Open Access Journal of Power and Energy*, vol. 9, pp. 573–583, 2022.

[18] G. Canonaco, A. Bergamasco, A. Mongelluzzo, and M. Roveri, "Adaptive federated learning in presence of concept drift," in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–7.

[19] X. Ma, J. Zhu, and M. B. Blaschko, "Tackling personalized federated learning with label concept drift via hierarchical bayesian modeling," in *Workshop on Federated Learning: Recent Advances and New Challenges (in Conjunction with NeurIPS 2022)*, 2022. [Online]. Available: https://openreview.net/forum?id=RBPvr4Ehojh

[20] Y. Jiang, J. Konečnỳ, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," *arXiv preprint arXiv:1909.12488*, 2019.

[21] J. Mills, J. Hu, and G. Min, "Multi-task federated learning for personalised deep neural networks in edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 3, pp. 630–641, 2021.

[22] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[23] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[24] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948 vol.4.

[25] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, 1998, pp. 69–73.

[26] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation*, vol. 5. IEEE, 1997, pp. 4104–4108.

[27] M. G. Omran, A. Salman, and A. P. Engelbrecht, "Dynamic clustering using particle swarm optimization with application in image segmentation," *Pattern Analysis and Applications*, vol. 8, pp. 332–344, 2006.

[28] C. Beckel, L. Sadamori, and S. Santini, "Automatic socio-economic classification of households using electricity consumption data," in *Proceedings of the fourth international conference on Future energy systems*, 2013, pp. 75–86.

[29] L. DataStore, "Smartmeter energy consumption data in london households [online], available at: https://data.london.gov.uk/dataset/smartmeter-energy-use-data-in-london-households," 2018.

[30] M. Omran, A. P. Engelbrecht, and A. Salman, "Particle swarm optimization method for image clustering," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 19, no. 03, pp. 297–321, 2005.

[31] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0377042787901257

[32] W. J. Conover, *Practical nonparametric statistics*. john wiley & sons, 1999, vol. 350.

[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.