ORIGINAL RESEARCH **Open access**

Straggler-Resilient Decentralized Federated Learning for Residential Energy Forecasting in Digital Twin Networks

Patrick O'Connor[1*], Sean Murphy[1]

Abstract

Decentralised federated learning (DFL) revolutionises collaborative energy usage forecasting by leveraging distributed computing across an extensive network of edge nodes, thereby protecting data privacy by removing centralised data collection. Maintaining individual confidentiality in energy prediction is critical, as it shields sensitive personal information from unwarranted scrutiny. Consequently, managing local data securely is essential for robust privacy preservation. In this study, a DFL architecture is introduced for predicting residential energy consumption, with a primary objective of enhancing the collaborative model's predictive accuracy and convergence rate. This architecture facilitates the local training of a long short-term memory (LSTM) model using live domestic energy data within a ring network topology. Crucially, this system tackles the challenge of straggler nodes—those experiencing computational or communication delays—by introducing a heuristic method to identify and mitigate these lagging nodes, thereby minimising their detrimental effects on system efficacy and network efficiency. By doing so, the framework not only elevates collaborative forecasting accuracy but also substantially decreases wait times, accelerating convergence. Empirical evaluations consistently reveal a low mean absolute error ranging from 3 to 3.2 across the participating edge nodes. These experimental outcomes decisively demonstrate the capability of the proposed DFL framework to optimise communication efficiency and boost overall predictive performance simultaneously.

Keywords Decision making, Learning (Artificial Intelligence), Digital twin networks, Long short-term memory (LSTM)

*Correspondence:
Patrick O'Connor
patrick.oconnor@tcd.ie
[1] Department of AI for Sustainable Energy, School of Engineering, Trinity College Dublin, Dublin, Ireland

Introduction

Recent years have witnessed a marked increase in the energy demands of residential properties [1]. A combination of climate change impacts, rapid technological progress, and an increased desire for indoor thermal comfort largely drives this upward trajectory. On a global scale, residential structures account for a major share—roughly 40%—of overall energy use, exceeding the consumption levels of both the industrial and commercial sectors [2]. Such elevated energy usage not only exacerbates environmental crises, including greenhouse gas emissions and global warming, but also places a direct economic strain on households via higher monthly utility costs [3].

Energy expenditure in homes covers a broad spectrum of functions, including lighting, climate control (heating and air conditioning), and the daily operation of household appliances. The patterns of this energy usage are closely tied to the number of occupants and prevailing indoor temperatures [4]. Mitigating energy waste is crucial for broader sustainability goals, given that excessive power consumption hinders the effective rollout of energy conservation initiatives [5, 6]. Consequently, there is an escalating need for predictive frameworks capable of forecasting residential energy demands to

This is an open access article distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License (CC BY-NC-SA 4.0), which permits non-commercial use, distribution, reproduction, and adaptation in any medium or format, provided that appropriate credit is given to the original author(s) and source, a link to the license is provided, and any changes made are indicated. Adapted material must be distributed under the same license.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

tackle these pressing issues. Implementing such forecasting systems can enhance management practices within the energy sector and play a pivotal role in reducing carbon footprints [7].

To efficiently monitor and record power usage, smart metres are frequently installed across various residential properties and apartments [8]. These devices provide critical data for forecasting energy needs, which in turn empowers building occupants to streamline consumption, boost overall efficiency, and curtail power waste [9]. However, the data collected by smart home technologies contains highly private details regarding residents' financial status, daily habits, and historical billing, which consequently raises severe privacy concerns.

Decentralised energy administration frameworks, particularly federated learning (FL), offer a promising solution by enabling cooperative model development without requiring the transfer of raw data from individual nodes. Within this progressive architecture, participating nodes transmit only their computed model weights to a central aggregating server instead of exposing underlying datasets. Each FL node trains its model locally, relying exclusively on its proprietary data repository. Subsequently, the node forwards only the parameters derived from its trained model to the central server. This deliberate methodology safeguards user privacy by ensuring that all sensitive information remains strictly confined to the local edge nodes. Despite these benefits, employing decentralised training architectures presents certain vulnerabilities, most notably susceptibility to semantic backdoor attacks. Such malicious intrusions could potentially gain unauthorised access to sensitive information. Moreover, the traditional FL architecture's dependence on a central server entity establishes a single point of failure, thereby elevating the entire system's risk of operational disruption.

Numerous recent investigations have explored the application of FL for forecasting residential energy use, yielding valuable perspectives on innovative techniques and strategies. Notably, the approach introduced by Savi and Olivadese [10] incorporated fuzzy logic to estimate power consumption within residential buildings. In a related effort, Nightingale *et al*. [11] focused on energy forecasting by leveraging smart metering customer behaviour trial data in conjunction with FL techniques. Additionally, Mendes *et al*. [12] applied a horizontal FL methodology to predict energy demand. Building on this research trajectory, Li *et al*. [13] utilised an FL-driven method to train a comprehensive predictive model using domestic building data to elevate forecast precision. Exploring an alternative approach, Tang *et al*. [14] introduced a knowledge-sharing framework designed to overcome the hurdle of limited energy data samples. By employing a central server, this framework facilitated the exchange of models among various nodes, culminating in the development of a robust overarching model for energy estimation. The knowledge gleaned from this global model was then transferred to secondary building datasets to refine forecasting accuracy further.

Furthermore, Tun *et al*. [15] presented an FL-based framework for anticipating energy demand. Their strategy relied on training separate models across various edge nodes and subsequently aggregating their weights at a central server. In contrast, earlier scholarly works, such as Elbeltagi and Wefki [16], Olu-Ajayi *et al.* [17], deployed traditional machine learning techniques—including artificial neural networks (ANN), random forests, and support vector machines—to forecast domestic energy consumption; similarly, Khan *et al*. [18, 19] examined household energy predictions using ensemble voting mechanisms and long short-term memory (LSTM) models. Collectively, these contributions highlight the growing enthusiasm and methodological diversity surrounding FL-enabled energy prediction. Analogous works include the findings in Elbeltagi and Wefki [16], Olu-Ajayi *et al.* [17], Khan *et al*. [18, 19], which executed their energy forecasting processes in a distinctly centralised manner.

None of the aforementioned methods, encompassing those detailed in Savi and Olivadese [10], Nightingale *et al*. [11], Mendes *et al*. [12], Li *et al*. [13], Tang *et al*. [14], and Tun *et al*. [15], provide inherent privacy guarantees. These systems rely on the centralised sharing of data originating from distinct residential buildings, which significantly raises privacy risks. The highlighted literature underscores particular obstacles that require attention when deploying centralised FL strategies. These challenges include the potential vulnerability of a single point of failure, reduced communication efficiency, limited scalability, and increased demands for computing resources. Such limitations emphasise the critical importance of investigating alternative methodologies, as exemplified by the decentralised FL framework proposed in this study.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

To resolve these concerns, we present an innovative methodology grounded in decentralised FL. Under the proposed decentralised federated learning (DFL) paradigm, every edge node swiftly exchanges its updated model with an adjacent edge node, thereby mitigating the potentially widespread damage caused by resilient attacks. This research introduces a novel DFL architecture specifically engineered for predicting energy consumption. This framework leverages a decentralised cooperative training method, completely bypassing the requirement for a central server. In this conceptual design, the edge nodes communicate directly with one another. Under this decentralised architecture, nodes establish connections in a ring topology, wherein each node maintains direct links to exactly two neighbouring peers: the preceding node and the subsequent node. Every residential building functions as an autonomous entity, developing a dedicated learning model based on its local data, including climate conditions and historical energy usage patterns. Following this, through a collaborative mechanism, every active node shares its locally trained model weights with its direct neighbours. This seamless exchange of model parameters takes place during each designated communication cycle. To guarantee efficient and secure data transmission among the various nodes, the communication process is orchestrated via the transmission control protocol (TCP). Concurrently, the vital task of weight aggregation is executed using the FedAvg method. FedAvg facilitates the harmonious integration of a node's local model weights with those supplied by its neighbours, effectively consolidating the distributed intelligence of the entire network. Furthermore, to maintain precise alignment across the ring-configured edge nodes, we implement the bulk synchronous parallel (BSP) synchronisation technique. This synchronisation method guarantees that inter-node communications and model refinements are executed in a coordinated manner, further boosting overall system efficiency and dependability.

Additionally, this research study introduces an original straggler mitigation technique formulated to counteract the adverse effects caused by under-performing edge nodes. For the core learning algorithm, we selected the LSTM model, widely recognised for its proficiency in processing sequential data. The LSTM model outperforms traditional frameworks in energy forecasting tasks because of its ability to capture the sequential traits of energy metrics, thereby allowing effective modelling of long-term temporal dependencies [18]. LSTMs offer remarkable flexibility, seamlessly processing data sequences of varying lengths. This characteristic enables the effortless assimilation of the uneven and irregular time-series data typical of energy records. By integrating sophisticated gating mechanisms, LSTMs effectively overcome the vanishing gradient problem, empowering the network to learn both short-term and long-term patterns within energy datasets.

Furthermore, LSTMs' ability to autonomously extract and learn pertinent features from raw sequential inputs removes the need for manual feature engineering, thereby improving the efficiency and precision of the collaborative prediction models. Ultimately, this decentralised framework integrates vital components to deliver a unique solution that encompasses privacy preservation, decentralised collaboration, and the effective resolution of synchronisation hurdles. This integration produces a comprehensive and adaptable platform designed for the precise prediction of residential energy usage, offering a substantial benefit for professionals in the energy sector. The primary contributions are detailed below:

· Construct a decentralised FL architecture that collaboratively trains a machine-learning model to predict energy consumption.

· Propose an innovative neighbour-aware straggler mitigation technique within a decentralised FL environment to reduce the negative impact of delayed nodes.

· Utilise TCP to facilitate communication across edge nodes, alongside the implementation of the BSP method to synchronise the edge network.

· Evaluate the performance of the proposed framework using a highly granular energy consumption dataset collected from 9 residential buildings.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

Background of FL

Federated learning (FL) is a developing approach that trains machine learning models cooperatively to establish a global model across a distributed network [20]. Within the FL framework, a set of K nodes, each possessing a distinct dataset D, carry out local training T during every communication round R, subsequently transmitting their calculated weights wk to a centralized server. The local loss function for a given node K is expressed as:

$$f_k(w_k) = 1/m_k \sum_{i=1}^{m_k} loss\left(x_j, y_j : w_k\right) \quad (1)$$

Where loss(xj, yj: wk) denotes the specific loss function of node k evaluated for individual data points xj, yj relative to the local parameter wk, while mk signifies the aggregate count of data samples.

The objective of this cooperative training environment is to optimize a global model parameter Wg that minimizes the overarching global loss F(w) across the entire training dataset. The mathematical representation of the global loss function is:

$$W_g = \arg\left(\min\left(\frac{1}{k}\sum_{k=1}^{K} f_k(w_k)\right)\right) \quad (2)$$

Where fk(wk) represents the individual loss value of a particular node, and K indicates the total number of participating nodes.

During every communication interval, individual local nodes execute training operations to calculate their respective local weights wl, which are then forwarded to the server. The central server subsequently compiles the weights from all active nodes by applying the FedAvg algorithm. The mathematical formulation for weight aggregation via FedAvg is given by **Eq. 3:**

$$W_g = 1/k \sum_{i=1}^{K} wl_i \quad (3)$$

Where wli constitutes the weight vector of an individual local node, and Wg represents the combined global weights returned to the nodes by the server.

In centralized federated learning (CFL), model aggregation relies entirely on a core central server [21]. Nevertheless, CFL architectures are vulnerable to a single point of failure; if the central server experiences an outage, the entire cooperative training process halts [22]. Participating nodes in this joint training scheme constantly transmit and receive their parameter weights from the server. Consequently, when scaling to a larger network of nodes handling a vast quantity of model parameters, substantial network bandwidth becomes necessary [23]. In contrast, decentralized federated learning (DFL) operates without a central server entity, enabling edge nodes to interact directly with one another [24]. DFL offers superior privacy preservation compared to CFL, as model weights or gradients are never routed through a centralized point. This reduces the risk of malicious data exposure while simultaneously enhancing system scalability [25].

In the domain of energy forecasting, securing local data privacy is crucial not merely as a technical specification, but as an essential safeguard for individual autonomy and safety. Power usage profiles reveal detailed insights into personal schedules and lifestyle habits, ranging from ordinary daily timelines to critical indicators like vacant household intervals. If the confidentiality of this data is breached, the exposed information could be exploited for aggressive marketing, calculated property thefts, or sophisticated forms of identity fraud and surveillance. Unregulated sharing of energy

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

consumption records compromises personal security and introduces wider social risks, undermining community trust in utility corporations and smart grid infrastructures. Furthermore, it complicates compliance with rigorous legal mandates, such as Europe's General Data Protection Regulation (GDPR) [26], which dictates stringent privacy controls and data governance standards. As a result, protecting privacy within energy forecasting frameworks requires more than basic data anonymisation; it demands the deployment of a fully decentralized, privacy-focused architecture.

Related Work

A limited number of investigations exist in the current literature regarding the prediction of energy consumption in domestic structures.

FL-based methods for energy forecasting

An investigation [27] predicted the electrical load of domestic buildings by utilizing smart meter metrics. This research applied two distinct FL algorithms, FedSGD and FedAvg, to execute cooperative learning across the node network. The empirical outcomes indicated that FedAvg achieved superior performance compared to FedSGD regarding predictive accuracy. A separate study [28] introduced an FL-centric approach to project net energy within net metering infrastructures. To safeguard user privacy, the researchers developed a secure aggregation technique utilizing cryptographic encryption of the node models. Similarly, Briggs *et al*. [29] introduced an FL framework designed for short-term residential load forecasting. This near-term energy prediction assists property owners in boosting efficiency and refining power utilization patterns. In their design, each independent node utilized an LSTM network to conduct local training on proprietary smart meter datasets. In another inquiry, Gholizadeh and Musilek [30] likewise deployed the LSTM architecture for localized training at each node to anticipate energy consumption. This approach implements the FedAvg algorithm on a centralized server to consolidate the locally trained models.

Furthermore, Zhang *et al*. [31] implemented a hybrid CNN-Attention-LSTM architecture to evaluate energy load projections across a multi-node network. Their methodology compared four FL optimization techniques: FedAvg, FedYogi, FedAdagrad, and FedAdam, while also assessing system resilience against false data injection exploits. Their conclusions indicated that FedAdagrad remains robust under false data injection conditions. An additional FL framework was presented by Petrangeli *et al*. [32] for projecting domestic energy requirements. Meanwhile, Wang *et al*. [33] introduced a customized FL strategy to predict the power usage of specific consumers. Along a similar line of inquiry, another publication [34] integrated FL concepts into energy load forecasting. In that research, a global model was trained using datasets from various residential properties to predict energy demand, with the resulting global insights subsequently transferred to execute predictions on external building data. Additionally, Gao *et al*. [35] proposed a fully decentralized FL structure for household energy forecasting, validating their framework using the open-source Pecan Street dataset.

FL-based methods for straggler mitigation

Within FL environments comprising numerous edge nodes, variations in computational capacity and network bandwidth inevitably occur across the hardware. Consequently, faster edge nodes experience idle waiting periods for lagging peers if specific units require extended time for local training cycles. To minimize the disruptive impact of these stragglers, several countermeasures have been introduced in current literature. A study [36] formulated a strategy to optimize network efficiency and curtail the influence of lagging nodes. This approach dynamically modifies the local batch configurations and epoch counts of slower nodes to decrease server latency. Their empirical data indicated that this technique accelerates convergence while reducing server idle time. Research conducted by Tziotis *et al.* [37] targeted the joint challenges of data and system heterogeneity by optimizing node-specific parameters to formulate customized models for individual participants. They introduced an adaptive participant selection framework to mitigate straggler interference, wherein nodes are picked based on their statistical relevance and processing throughput.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

The investigators in Reisizadeh *et al*. [38] likewise developed a straggler-resistant mechanism leveraging the statistical behavior of participating nodes. Their framework incorporates adaptive node selection to alleviate delays and accelerate the overall training trajectory, initiating optimization cycles with high-performance nodes before progressively integrating slower units. In a separate publication, Wolfrath *et al*. [39] designed a server-side participant selection mechanism to dynamically manage nodes within heterogeneous environments. This method capitalizes on statistical variations and exhibits robust resilience against random node disconnections. In Sultana *et al*. [40], researchers developed an elastic FL technique aimed at diminishing straggler impacts through a targeted node selection policy. By filtering nodes at the server level, they simultaneously managed computational delays and statistical data discrepancies. An analysis in Elzohairy *et al*. [41] addressed FL network constraints by dynamically categorizing participating devices into three distinct performance echelons—rookies, active participants, and stragglers—based on behavioral profiles. These operational profiles are derived from observed training durations, thermal cooling intervals, and omitted training rounds, complemented by a staleness-aware model aggregation technique.

Coordination strategies within FL play a critical role in dictating network efficiency and model convergence rates. A paper by Xie *et al*. [42] introduced a solution utilizing asynchronous coordination mechanisms. This architecture operates via dual execution threads: a scheduler and an updater. The updater thread retrieves the global model from the server to merge it with local weights. In contrast, the scheduler thread routinely triggers local training steps using the most up-to-date global parameters.

Furthermore, Tao and Zhou [43] presented an n-cycle redundant tree architecture designed for fully decentralized FL environments. This methodology seeks to maximize system redundancy and alleviate straggler bottlenecks by exploiting hierarchical network layouts. Under this scheme, a designated fusion node serves as the structural root, while remaining nodes function as intermediate relays or terminal leaves, all orchestrated via a distributed message-passing interface. The disruptions caused by lagging nodes and hostile intrusions are countered in Park *et al*. [44] through a staleness-conscious grouping strategy. This framework integrates loss-weighted averaging alongside entropy-driven filtering techniques. This combination of model categorization and weighted balancing provides strong systemic defense against stragglers.

A separate investigation conducted by Liu *et al*. [45] utilized a partial model aggregation policy to offset delays from slower devices. Their architecture updates global weights as soon as a designated threshold of K nodes check in with the server, dynamically optimizing this K value over time. Furthermore, their approach integrates outdated updates while evaluating both the benefits and drawbacks of stale parameters. Additionally, Dhakal *et al*. [46] devised a coded computing environment to mitigate straggler-induced bottlenecks. At the initialization of cooperative training, every node creates parity training sets and distributes them to a core server. This central hub then performs redundant gradient estimates on the parity data to compensate for delayed node updates caused by local training lag or network latency. Another study [47] overcomes straggler challenges by embedding an alternative operational phase into FL, termed split learning. Delayed nodes execute cooperative training steps with their immediate geometric neighbors using split learning methodologies.

Current research, such as the works highlighted in Fekri *et al*. [27], Badr *et al*. [28], Briggs *et al*. [29], Gholizadeh and Musilek [30], and Zhang *et al.* [31], has successfully produced diverse CFL architectures optimized for specialized use cases. However, CFL systems inherently struggle with scalability and demand substantial computational infrastructure. For instance, Park *et al*. [36] combated lagging devices in centralized FL by reconfiguring the batch dimensions and local training epochs of the affected nodes. Similarly, Tziotis *et al*. [37] introduced a participant selection protocol for centralized FL. Parallel approaches, including Reisizadeh *et al*. [38], likewise exploit node selection routines that evaluate data statistics alongside device hardware limits. While numerous frameworks exist to counter the effects of lagging nodes in FL, to the best of our knowledge, no existing methodology incorporates an adaptive neighbor-centric strategy to mitigate stragglers in the manner presented by this framework.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

Decentralized FL For Energy Consumption Prediction

The choice of network topology is critical in decentralized federated learning (DFL), with no single configuration universally outperforming others. Nonetheless, a ring topology offers distinct advantages regarding scalability, operational efficiency, and fault tolerance [48]. Consequently, this research constructs a DFL architecture utilizing a ring topology specifically for energy consumption forecasting. The dataset comprises consumption records from 9 residential structures, corresponding to nine edge nodes within the network. These nodes train local models on their proprietary data and then transmit their local parameters to the adjacent node, thereby facilitating collaborative learning without exposing raw data. The overall architecture of the envisioned framework is illustrated in **Figure 1**, while the precise operational steps are outlined in **Algorithm 1.**

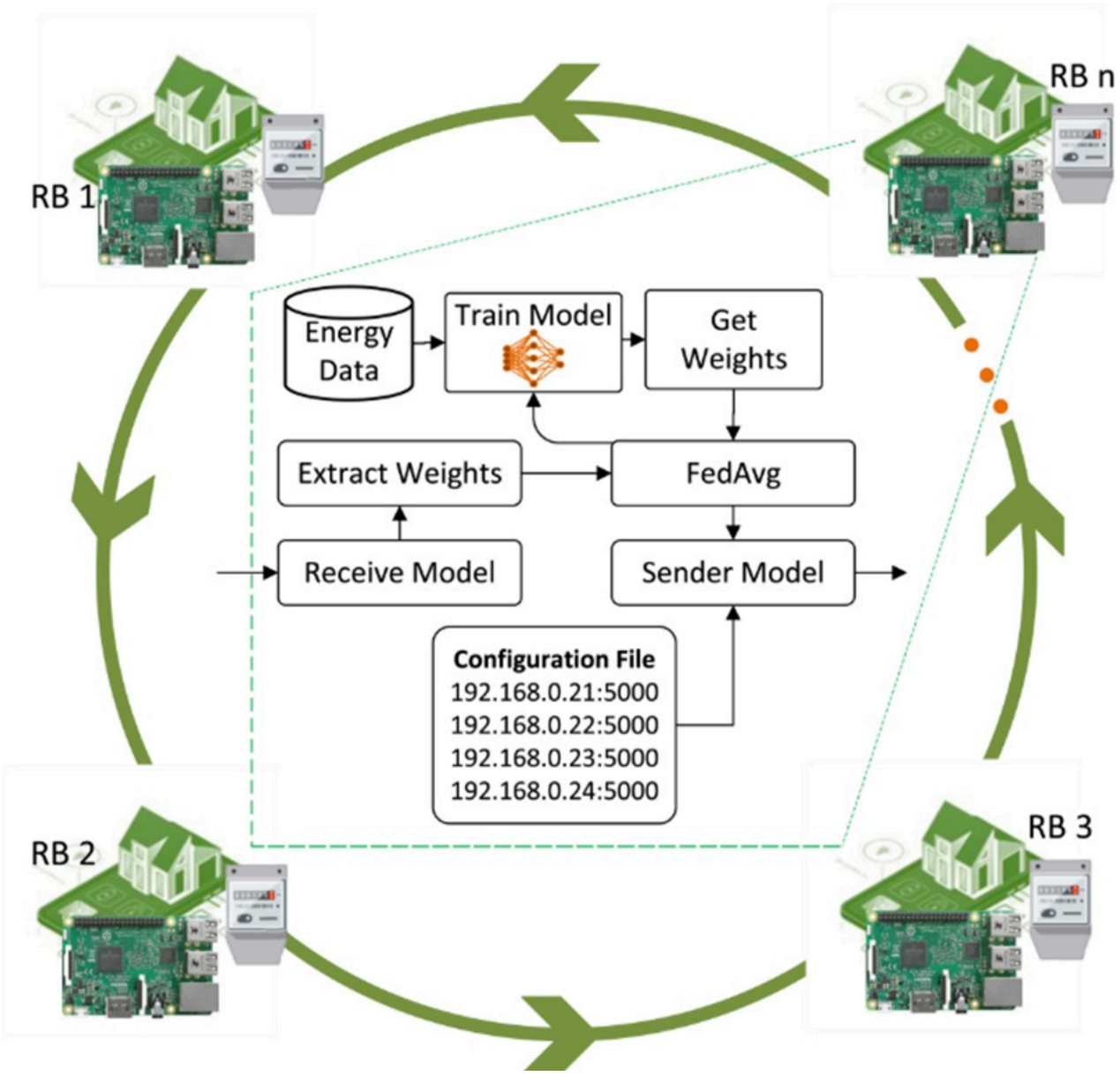


Figure 1. Architecture diagram of proposed decentralised federated learning framework for energy consumption prediction.

The system design for predicting energy consumption through a collaborative training paradigm is detailed in **Figure 1**. Each residential unit is equipped with a Raspberry Pi hardware device dedicated to running the local training cycles of the

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

deep learning model. Localized energy consumption metrics are fed directly into this deep learning model to forecast future demand. After training completes, the model parameters are extracted, and each edge component routes its model to the directly succeeding neighbor node. During this transmission phase, the sending module references a configuration file to acquire the specific IP address and communication port of the target receiving node. This receiving entity extracts the incoming weights, which arrive formatted as JSON data. Subsequently, the node merges the newly acquired weights with its own localized model parameters by applying the FedAvg aggregation mechanism. Once consolidated, the updated model is passed forward to the next consecutive neighbor.

Data: $data = (x_1, x_2, x_3, ...x_n), Epoch, GlobalRound$
Result: GlobalModel
initialization;
$con \leftarrow$ ConfigurationLoading()
$isFirstNode \leftarrow con['nodetype']$
CurrRound=0
GlobalRound=20
while *CurrRound<GlobalRound* **do**
  $Rec_Model, recconformationflag \leftarrow ReceiverListening()$
  if *isfirstnode==True and CurrRound==0* **then**
    Model $\leftarrow ModelTraining(data, epoch)$
    weights $\leftarrow getweights(Model)$
    Model.json $\leftarrow Json.dumbs(weights)$
    modelreceivingconformation$\leftarrow sendmodel(Model.json)$
    isfirstnode=False
    CurrRound=CurrRound+1
  if *isfirstnode==False* **then**
    Model $\leftarrow ModelTraining(data, epoch)$
    localweights $\leftarrow getweights(Model)$
    if *recconformationflag==1* **then**
      recweights $\leftarrow getweightsfromrecivedmodel(Rec_Mode)$
      weight $\leftarrow weightaggregation(localweights, recweights)$
      model.json $\leftarrow Json.dumbs(weight)$
      modelreceivingconformation$\leftarrow sendmodel(model.json)$
      model $\leftarrow model.setweight(weight)$
      MAPE, MAE $\leftarrow model.evaluate(data)$
      CurrRound=CurrRound+1

Algorithm 1. Decentralised federated learning framework

The procedural steps of the introduced collaborative learning paradigm for forecasting energy demands are delineated in **Algorithm 1**. As indicated, each network node features an embedded receiver function tasked with monitoring the

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

channel for incoming models and associated configuration parameters. During the inaugural round, the initial node within the ring network kicks off local training and dispatches its parameter weights to its direct neighbor. Upon successful delivery, the recipient node replies with an acknowledgment message to the transmitting node. The receiving node then consolidates its own local weights with the incoming parameters, archives the updated model weights in JSON format, and routes the joint model to its respective successor. Following this, the unified weights serve as the baseline for the subsequent local training iteration. This cyclical procedure repeats for a total of 20 communication rounds.

Ring topology

In this work, a DFL network is established in which edge nodes are interconnected using a ring configuration [49]. Within this ring arrangement, every edge node maintains direct connections with its immediate predecessor and successor. This layout allows each node to accept an incoming model from the prior device and forward updates to the subsequent one. At the onset of joint training in the initial communication cycle, the primary edge node begins local computation. After concluding its training phase, it dispatches the resulting local model to the following edge node. The recipient node then executes its own local training before computing the mathematical average of its local parameters and the received weights. Following this aggregation step, the consolidated model is forwarded to the next sequential node in the circuit, repeating this workflow until the active global iteration reaches the target number of total global rounds. Operating under a ring topology delivers distinct benefits, including minimized latency, enhanced scalability, and balanced data distribution. The layout of this network configuration is depicted in **Figure 2.**

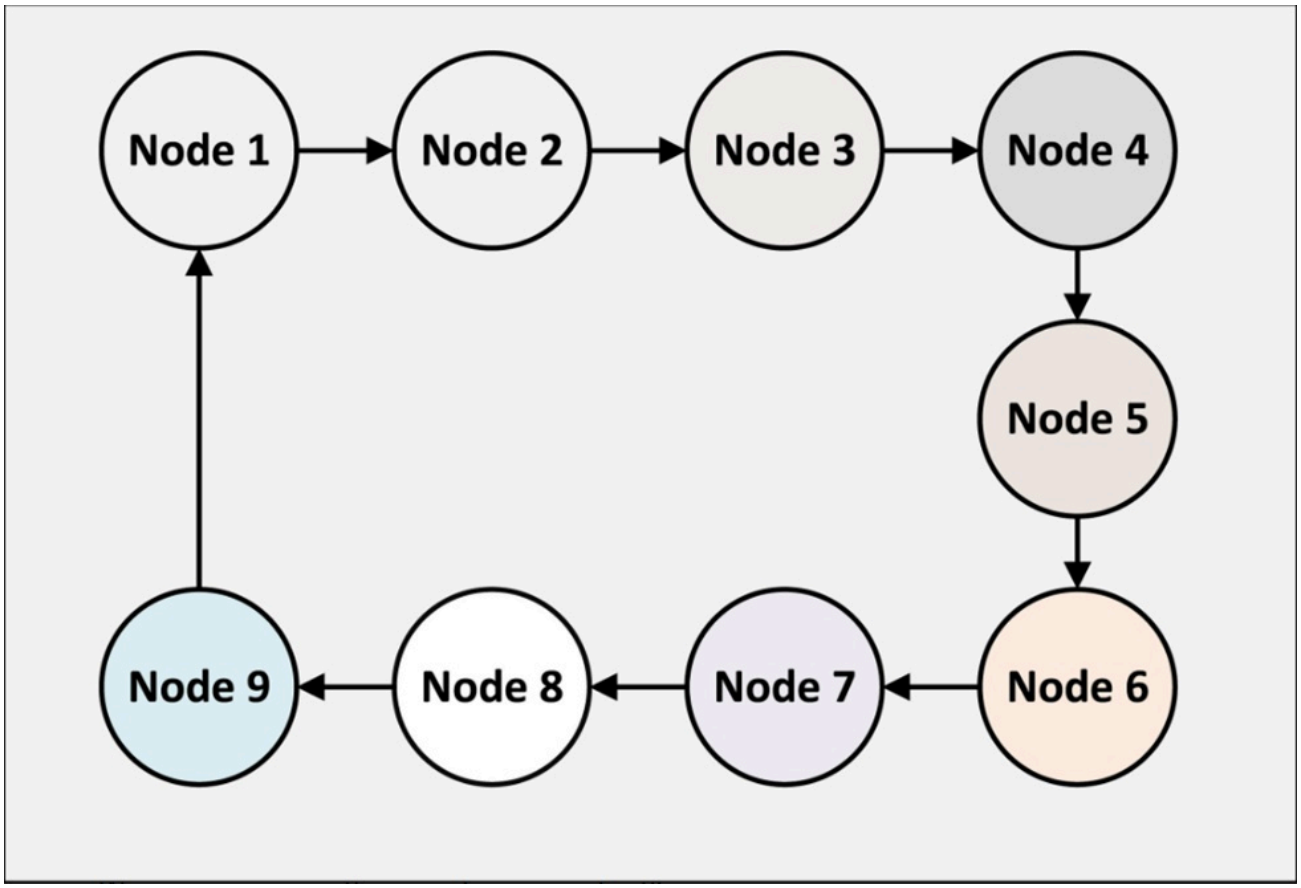


Figure 2. Ring topology for decentralised federated learning network.

As illustrated in **Figure 2**, a network of nine nodes forms the structural ring, with each node corresponding to an individual residential property. Communication across these nine elements is managed via the TCP protocol, with each device routing model configurations to its direct successor. This particular topology was intentionally chosen due to its inherent

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

vulnerability to straggler nodes. Lagging nodes have a more disruptive impact within a ring network than in mesh or star alternatives. For instance, in a mesh layout, if a node lags, neighboring clients can still acquire updates from alternative sources. Similarly, in a star framework, a central supervisor can proceed with aggregation by leveraging updates from active nodes while bypassing the straggler. In contrast, a delayed node in a ring architecture can bottleneck the entire sequential training pipeline. To address this risk, this study introduces a dedicated strategy to counter and mitigate straggler constraints specifically within a ring configuration.

Model transmission protocol

Inter-node communication within this framework is managed using the TCP standard [50], implemented through network socket programming interfaces. Each edge device in the network is configured with a distinct IP address and port number and operates using dedicated sending and receiving components. The receiving component captures incoming model files from the preceding node, whereas the sending module routes updates onward to the following node. The receiving port continuously monitors for inbound data. Once localized training wraps up, the sending unit dispatches the compressed JSON model file to the subsequent node. Upon successfully capturing the full model, the receiving node returns a delivery confirmation message to the sender. This specific protocol was chosen because it guarantees lossless and accurate delivery of model updates across nodes. **Algorithm 3** illustrates how the sending module processes the model parameters along with destination IP and port values to execute the transfer. Conversely, **Algorithm 2** details the inner workings of the receiving module, which updates the current communication round counter immediately upon successful model acquisition.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

Data: $GlobalRound$
Result: ModelReceived
initialization;
$con \leftarrow socketlistening(Ip, Port)$
$CurrRound = 0$
while *CurrRound<GlobalRound* **do**
  $receiveflag \leftarrow conn.receive(flag)$
  if *receiveflag==1* **then**
    $model \leftarrow conn.receivemodel()$
    $time \leftarrow checkmodelreceivedtime()$
    $confirmation \leftarrow conn.send('modelisreceived')$
    CurrRound=CurrRound+1
  if *receiveflag==0* **then**
    $time \leftarrow currenttime()$
    $file \leftarrow storetime(time)$
  if *receiveflag==2* **then**
    $confirmationformodelsending \leftarrow conn.receive()$
  if *receiveflag==3* **then**
    $confirmationforhalt \leftarrow conn.receive()$

Algorithm 2. Operational overview of receiver module in decentralised federated learning

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

Data: $model, Ip, Port$
Result: $ReceiveConfirmation$
initialization;
$con \leftarrow ConfigurationLoading()$
$ip \leftarrow con(ip)$
$port \leftarrow con(port)$
if $modelsend$==$True$ **then**
    flag==1
    $sock \leftarrow preparedsocket(ip, port)$
    $sock \leftarrow sock.send(flag)$
    $sock \leftarrow sock.send(model)$
    $modelisreceived \leftarrow sock.receive()$
    flag=0
    $timeflagrecieved \leftarrow send_confir_to_n_of_n_node(flag)$
if $straggler$==$True$ **then**
    flag1==2
    flag2==3
    $modelisreceived \leftarrow sock.send_confir_to_p_of_p_node(flag1)$
    $modelisreceived \leftarrow sock.send_confir_to_prev_node(flag2)$

Algorithm 3. Operational overview of sender module in decentralised federated learning

Aggregation method

Every edge component in the DFL ring configuration executes independent local training before combining the incoming weights with its own. The functional workflow for this consolidation process is defined in **Algorithm 4.** The parameters required by **Algorithm 4** include the local model weights, the incoming weights from the neighbor, the node count n, and the designated model layers. Setting n = 2 with a layer count of 4, the aggregation routine integrates the local parameters with the structural weights inherited from the antecedent node. This methodology performs the calculation by executing a precise, layer-by-layer consolidation of the weight structures from both the localized and incoming network models.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

Data: $localweight$,$receivedweight$, $layers$, n
Result: aggregatedmodel
initialization;
$aggregated \leftarrow []$
mergeweight $\leftarrow mergeweights(localweight, receivedweight)$
for *i in layers* **do**
    lay=[]
    for *j in n* **do**
        $lay \leftarrow lay.append(mergeweight[j][i])$
    $laywiseweigh \leftarrow \frac{1}{n}\Sigma_{i=1}^{n} lay_i$
    $aggregatedmodel \leftarrow append.aggregated(laywiseweigh)$

Algorithm 4. Layer-wise model aggregation in DFL

Synchronisation method

During collaborative DFL training iterations, numerous edge nodes conduct independent local learning and dispatch their resulting parameters to adjacent nodes. Managing this process requires a structured control protocol to coordinate training phases across all nodes. This framework applies the Bulk Synchronous Parallel (BSP) methodology to keep the various edge components synchronized. The operational sequence of this BSP implementation is mapped out in **Figure 3**.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

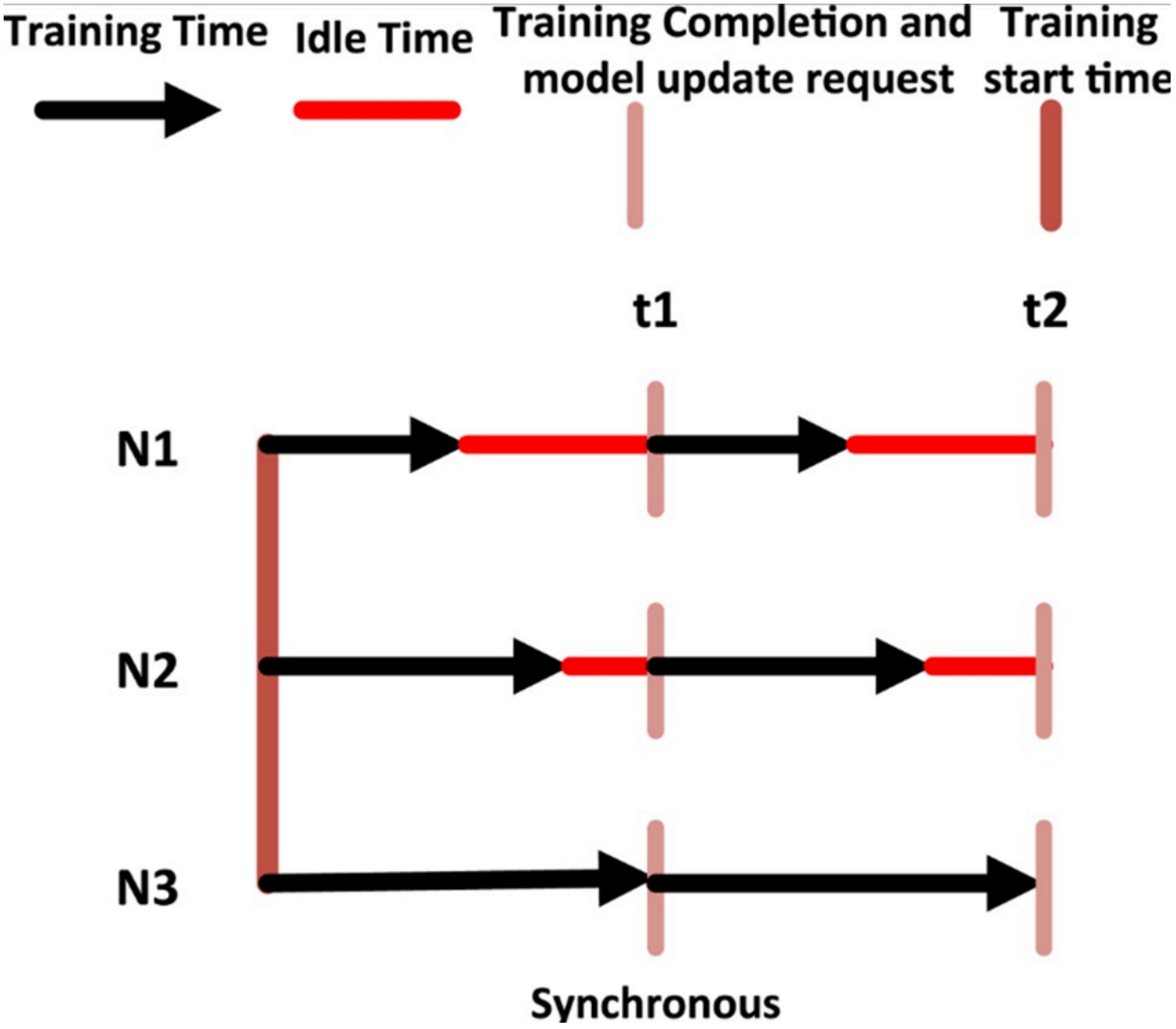


Figure 3. Working flow of BSP synchronisation method.

The BSP execution model mandates that an edge node completes its local training cycles and then pauses until it receives the model update from the preceding node. Upon successful delivery from the prior device, the node aggregates the parameters. Nevertheless, because network bandwidth varies across devices, communication channel issues can introduce processing latency [51, 52]. For instance, in a decentralized FL framework connecting M nodes, let edge node Mi be the recipient of a model from Mi−1. If network delays manifest, Mi must remain idle while waiting for Mi−1. Because the edge nodes are linked in a sequential ring, an extended delay at any single node cascades down the pipeline, forcing subsequent nodes into an idle state. In this sequential arrangement, each device interacts solely with its immediate upstream and downstream neighbors; thus, if Mi is stalled waiting for Mi−1, then Mi+1 is inevitably stalled waiting for Mi.

The functional timeline of the BSP coordination method is depicted in **Figure 3**. Under a classic BSP protocol, if high-performance nodes complete their local computations quickly while other units lag, the faster machines must remain inactive until the slower participants finish. This idle period for high-velocity nodes bottlenecks the overall speed of collaborative optimization. These lagging elements are typically termed straggler nodes. The diagram outlines three specific nodes: N1, N2, and N3. The indicators t1 and t2 denote the first and second communication intervals, highlighting the specific timestamps where weight aggregation occurs and the subsequent round of training begins. Black arrows

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

correspond to active processing durations, whereas red arrows mark the idle periods of the respective nodes. As shown, nodes N1 and N2 complete their tasks rapidly and enter a waiting state well before node N3, meaning they experience extended idle windows compared to N3 due to the latter's prolonged calculation time. Consequently, N3 acts as a straggler node within this workflow. Under this BSP setup, optimized nodes incur significant idle overhead while waiting for lagging devices.

Straggler mitigation method

Straggler bottlenecks emerge in DFL environments when specific edge nodes operate with lower processing speeds or constrained bandwidth relative to the rest of the network, requiring more time to execute assigned workloads. Within a DFL setup, each node depends on receiving an update from its upstream neighbor, meaning a lagging device delays overall model convergence. Minimizing the impact of these stragglers is essential for accelerating the training timeline. The foundational layout of the introduced mitigation framework is presented in **Figure 4.**

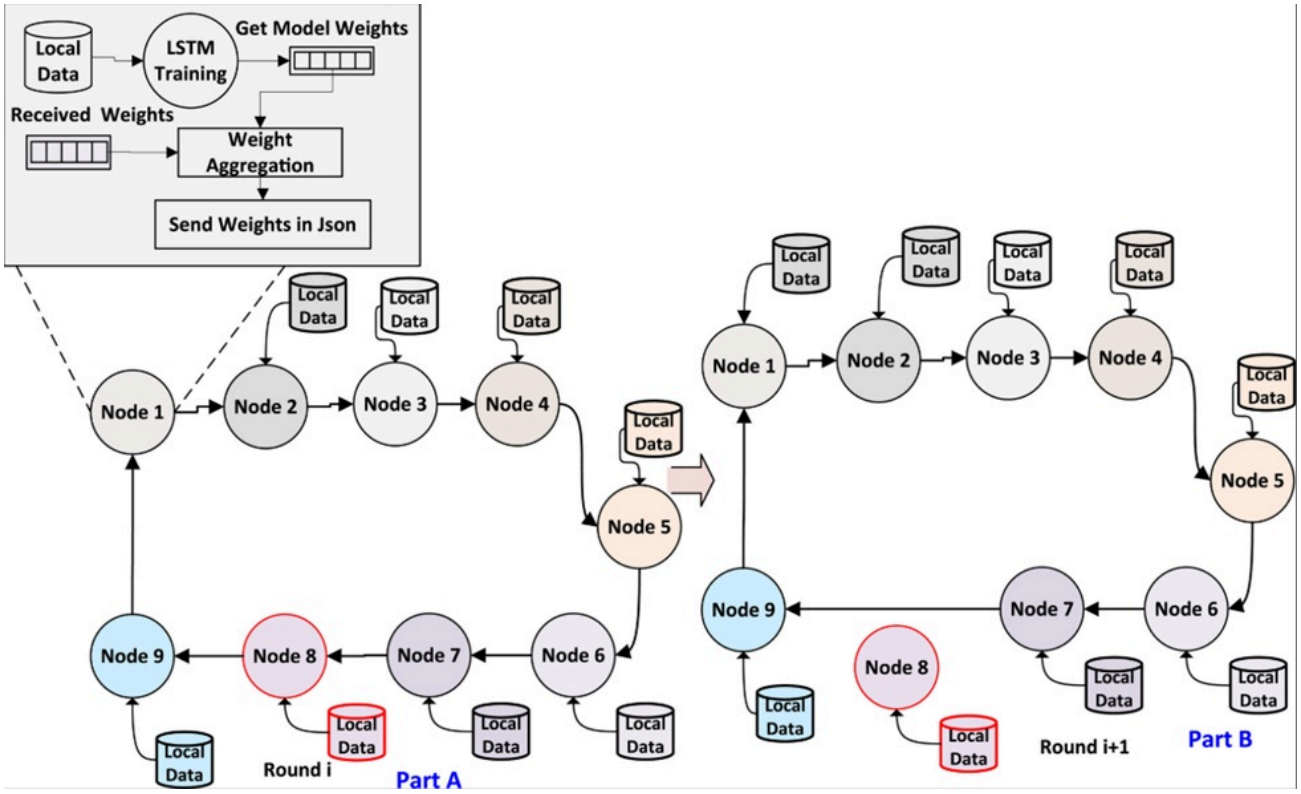


Figure 4. Architecture of a proposed straggler node mitigation method for energy consumption prediction.

The system architecture designed for straggler mitigation is detailed in **Figure 4**. The diagram displays nine interconnected edge devices cooperating to train an LSTM model, offering a comprehensive look at the operational routine used to detect lagging nodes. As shown in part A of **Figure 4,** node 8 is flagged as a straggler during communication round i by applying the technique. Moving to part B of the diagram, during the subsequent (i + 1)th iteration, node 9 bypasses the bottleneck by acquiring the model directly from node 7. The underlying mechanism for resolving straggler delays is further mapped out in **Figure 5.**

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

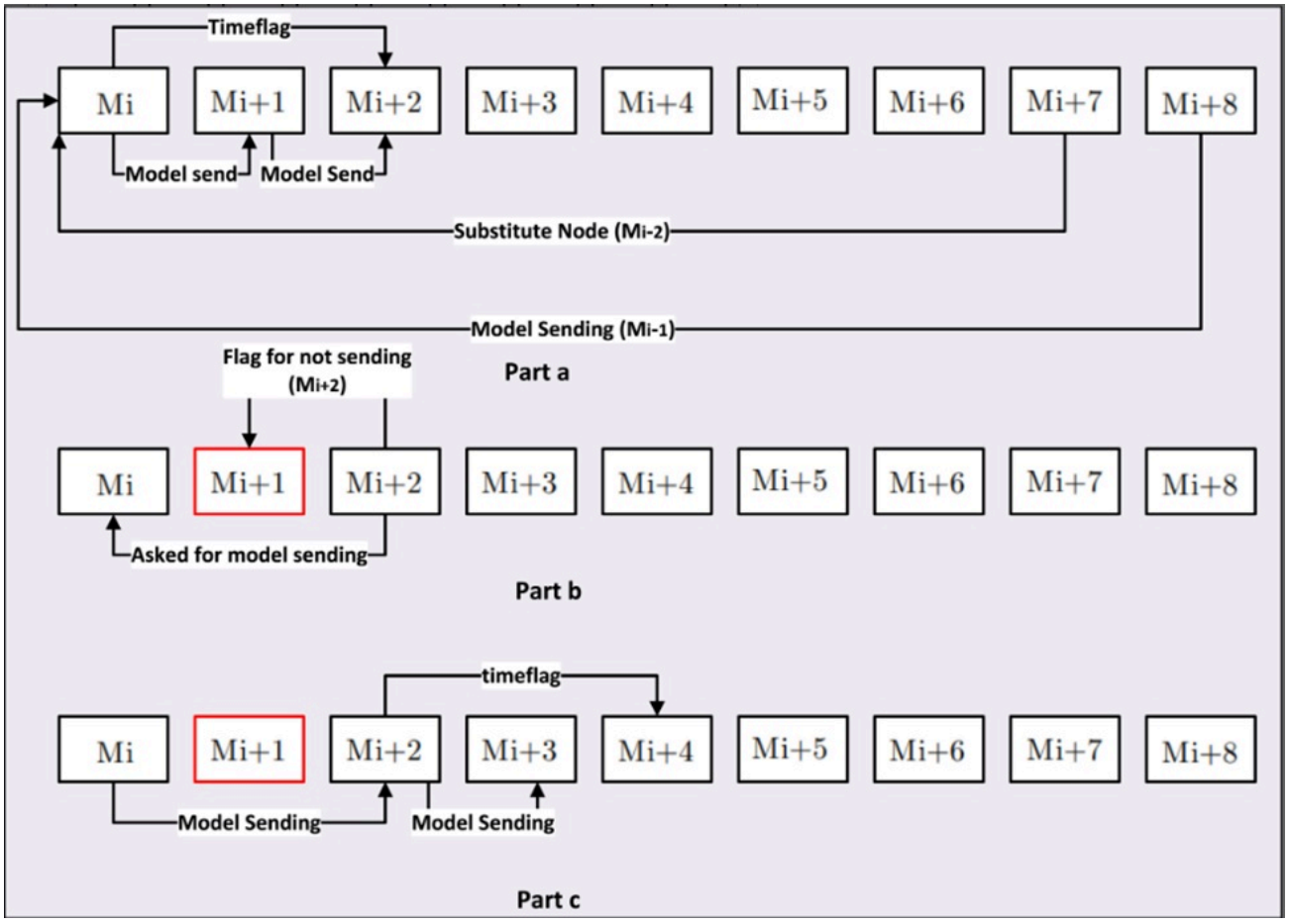


Figure 5. Schematic process for mitigating a straggler node.

Process of straggler node mitigation

This research introduces a dedicated mechanism to address straggler nodes within a decentralized federated learning environment. Under this framework, each participating edge device establishes connections with four distinct neighbors. Suppose the system contains M edge nodes, where a given node Mi is linked to two upstream predecessors, Mi−1 and Mi−2, as well as two downstream successors, Mi+1 and Mi+2. Node Mi dispatches its model updates to Mi+1, which returns a receipt acknowledgment to Mi once transmission concludes. Upon receiving this acknowledgment, Mi emits a specific confirmation bit designed to track the combined training and transmission duration of Mi+1. Device Mi+2 captures this confirmation bit and uses it to initiate a countdown tracking the expected turnaround time for node Mi+1. Once Mi+1 completes its localized training and weight consolidation, it routes the resulting model to Mi+2. Upon receiving the payload, Mi+2 evaluates the elapsed time; if this duration exceeds the expected baseline standard, Mi+2 prompts node Mi directly to supply the model parameters for the following cycle. Consequently, the lagging node Mi+1 is set to an idle state for the upcoming communication round, during which Mi+2 pulls the required model update directly from Mi. Every edge component throughout the network monitors these waiting periods and matches them against standard benchmarks. Whenever a delay passes the defined threshold, the receiving device signals its grandparent node (the previous-to-previous node) to transmit the model. This workflow is structurally diagrammed in **Figure 5** and formally detailed in **Algorithm 5.**

The benchmark for the expected waiting duration is determined via **Eq. 4.**

$$\text{Expected Waiting Time} = TT + LTT + \epsilon \quad (4)$$

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

Where TT represents the aggregate transmission duration, LTT signifies the localized training time, and ϵ represents the tolerance buffer assigned by the node. When a device acquires a model from its immediate predecessor, it measures the elapsed duration against this calculated baseline. If the observed waiting interval surpasses the benchmark expectation, the preceding node is classified as a straggler.

Data: $CurrentTime$
Result: $straggler, nonStraggler$
initialization;
$prevstoretime \leftarrow readfile(timestor)$
$timeinsecond \leftarrow calculatewaitingtime(curr, prevstoretime)$
if $timeinsecond > expectedtime$ **then**
    flag1==2
    flag2==3
    $flag_p_of_node \leftarrow send_confirmation_to_p_of_p_node(flag1)$
    $flag_p_node \leftarrow send_confirmation_to_prev_node(flag2)$

Algorithm 5. Straggler Identification Mechanism

Dataset, learning model and evaluation metrics

Dataset

This investigation carries out empirical evaluations utilizing an energy utilization dataset provided by a research institution. The dataset encompasses data regarding residential structures and meteorological conditions in Seoul, consisting of six independent features alongside the target energy consumption feature. This data represents 9 separate residential buildings, with the individual records from each structure allocated across 9 distinct network nodes. Let Difeature and Djfeature represent the feature spaces, while DiID and DJID denote the corresponding identification spaces. Within this dataset, Difeature = Djfeature and DiID ≠ DJID. The distribution of data across the residential structures is non-uniform, meaning each separate node exhibits a unique data distribution profile. A comprehensive summary of the dataset is outlined in **Table 1,** and the corresponding data configurations are visualized in **Figure 6.**

Table 1. Energy dataset details.

S. No.	Feature name	Description
1	Humidity	Indicates the humidity level as a percentage on a given day.
2	Temperature	Represents the temperature value recorded on a specific day.
3	Day of week	Specifies the day of the week, with possible values ranging from 1 to 7.
4	Month	Represents the month of the year, with values ranging from 1 to 12.
5	Weekend	Denotes whether the day falls on a weekend. Values can be 0 or 1, where 0 indicates weekend and 1 indicates a weekday.
6	Day of the month	Indicates the specific day of the month, ranging from 1 to 31.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

7	Energy	Represents the energy consumption of a residential building on that day, measured in kilowatts per hour (kWh).

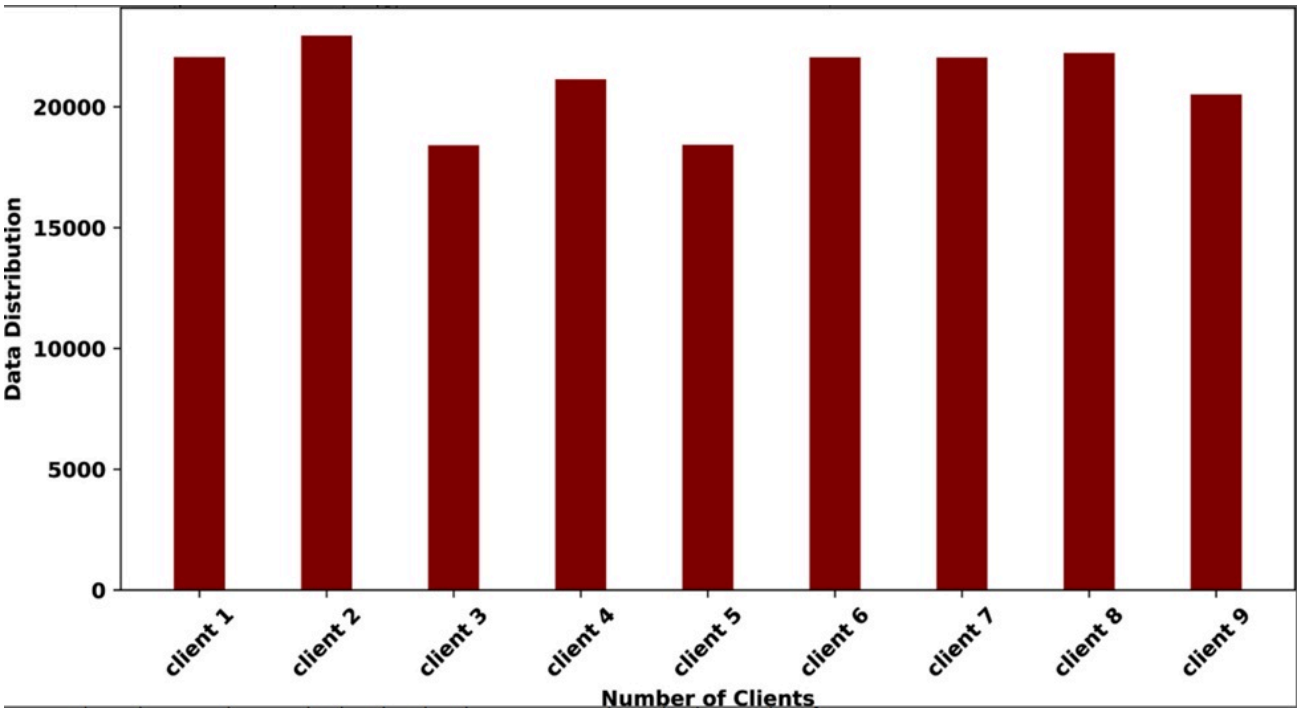


Figure 6. Data distribution for each residential building.

The data distribution profiles for the network nodes are illustrated in **Figure 6**. Within this graphical representation, the horizontal axis indicates the client designation, whereas the vertical axis reflects the corresponding sample distribution for each participant. The visualization highlights that client 2 possesses a larger volume of data samples relative to its peers. Conversely, client 3 contains a smaller quantity of data records compared to the remaining participants.

Learning model

Each edge device trains an LSTM model [53] using the energy dataset to forecast power consumption. The LSTM architecture represents a specialized category of recurrent neural networks capable of capturing extended spatial and temporal dependencies within sequential time-series energy records. It demonstrates superior computational efficacy compared to standard RNN architectures. An LSTM block is primarily composed of two core structural elements: control gates and memory cells. The gating mechanisms regulate information flow, while the memory cells maintain and store hidden states over time. Specifically, the forget gate determines which historical insights should be discarded and which should be preserved. When novel data enters the network, an input gate quantifies the significance of these incoming updates.

The architectural configurations and experimental parameters of the LSTM framework are detailed in **Table 2.** This investigation adopts the LSTM model for projecting energy demand due to its inherent compatibility with sequential time-series analysis. Furthermore, it successfully models both transient short-term fluctuations and persistent long-term trends. The sequential training logic of the LSTM model is formalized in **Algorithm 6.**

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

Table 2. Model architecture details.

S. No.	Parameter name	Detail
1	Optimiser	Adam
2	Loss function	Mean squared error
3	LSTM layers	3
4	Local epoch	5
5	Global communication rounds	20

Data: Energy Dataset $data = (s_1, s_2, s_3, \ldots s_n)$
Result: Energy Consumption
initialization;
Edata $\leftarrow Read_Csv()$;
$X, Y \leftarrow FeatureSplit(Edata)$
NormalizeX $\leftarrow Data_Standardization(X)$;
$testdata, ytest, trainset, ytrain \leftarrow TestTrainsplit(NormalizeX, Y)$
$LSTM \leftarrow LSTM_Parameters_initialization()$
for *each* ***Rounds*** **do**
  $LSTM \leftarrow LSTM_Training(Trainset, ytrain)$;
  $\hat{y} \leftarrow LSTM_Model_Prediction(testdata)$;
  Evaluate;
  $MAE \leftarrow \sum_{(i=1)}^{m} |(ytest - \hat{y})|$
  $MAPE \leftarrow 1/\mathrm{m} \sum_{(i=1)}^{m} |(ytest - \hat{y})/y|$
  $MSE \leftarrow \sum_{(i=1)}^{m} (ytest - \hat{y})^2)/m)$
  $RMSE \leftarrow \text{square root}(\sum_{(i=1)}^{m} (ytest - \hat{y})^2)/m)$

Algorithm 6. LSTM model training

Evaluation metrics

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

Quantitative evaluation metrics are essential for analyzing the predictive accuracy of machine learning frameworks. This research employs mean squared error, mean absolute error (MAE), root mean squared error (RMSE), and mean absolute percentage error (MAPE) to evaluate the forecasting performance of the trained LSTM model rigorously.

Mean absolute error

Mean absolute error (MAE) represents the mathematical average of the absolute variances between the true empirical observations and the values generated by the forecasting model. It is determined via the following expression:

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|y_i - \hat{y}_i| \quad (5)$$

Mean absolute percentage error

Mean absolute percentage error (MAPE) signifies the mean percentage of absolute deviations separating the actual ground truth observations from the predicted outcomes. It is calculated as follows:

$$MAPE = \frac{1}{m}\sum_{i=1}^{m}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (6)$$

Mean squared error

Mean squared error (MSE) measures the average of the squared deviations between the actual data points and the corresponding model predictions. It is computed according to the provided formula:

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \quad (7)$$

Root mean square error

Root mean squared error (RMSE) represents the square root of the MSE, where the underlying MSE computes the average of the squared variances between true measurements and estimated parameters. The mathematical formulation for RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2} \quad (8)$$

Where y represents the true ground truth value, and ŷ denotes the forecasted value.

Experimental Results

The decentralized federated learning (DFL) framework is developed using the core Python programming environment, with specific hardware and software technical configurations. Within this deployed DFL architecture, multiple edge devices are linked sequentially, with each node managing localized records corresponding to a specific residential building. Each participant trains an LSTM network using its localized variables (comprising meteorological conditions and building parameters) to forecast the electricity usage of that structure. Under this DFL paradigm, these distributed nodes cooperatively optimize a global model structure to inherit underlying data features from peer devices without exposing local raw datasets. The edge entities are arranged in a ring topology, where each node conducts localized optimization for five training epochs before routing its updated parameters to the adjacent subsequent device. This operational cycle is repeated across 19 communication rounds. Empirical observations revealed that communication channel constraints

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

force downstream nodes to wait for upstream updates. This investigation incorporates a straggler mitigation mechanism to minimize these idle intervals.

Experimental setup

The empirical evaluations in this research were executed utilizing Raspberry Pi 4 single-board computers. These Raspberry Pi systems are deployed within a smart home infrastructure to gather ambient environmental metrics and electricity consumption records from integrated smart meters. Atmospheric temperature and relative humidity data are captured via DHT sensor modules. Both environmental metrics and energy utilization records are sampled continuously at 1-hour intervals. The hardware configuration profiles for the Raspberry Pi units are detailed in **Table 3**. Following data acquisition, a collaborative training paradigm is initiated for the LSTM network to forecast energy requirements. The complete implementation specifications are structured in **Table 4.**

Table 3. Specification of Raspberry PI device.

S. No.	Component	Detail
1	CPU	Quad-core 1.5 GHz 64-bit CPU
2	Memory	4 GB
3	Hard disk	32 GB micro SD card
4	OS	Raspberry Pi 64-bit

Table 4. Implementation details.

S. No.	Components	Detail
1	Hardware	Raspberry Pi (4)
2	Edge devices	9
3	Communication protocol	TCP
4	Data file storage	MS excel
5	Programming language	Python
6	Python required libraries	Pandas, scikit, tensorflow, seaborn, matplotlib
7	IDE	Jupyter notebook and VS code

The technical specifications of the Raspberry Pi hardware node are detailed in **Table 3**. This study adopts the Raspberry Pi 4 model equipped with 4 GB of primary random-access memory, supplemented by a 32 GB SD card for secondary storage capacity. The physical and logical testing infrastructure for the engineered deployment is visualized in **Figure 7.**

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

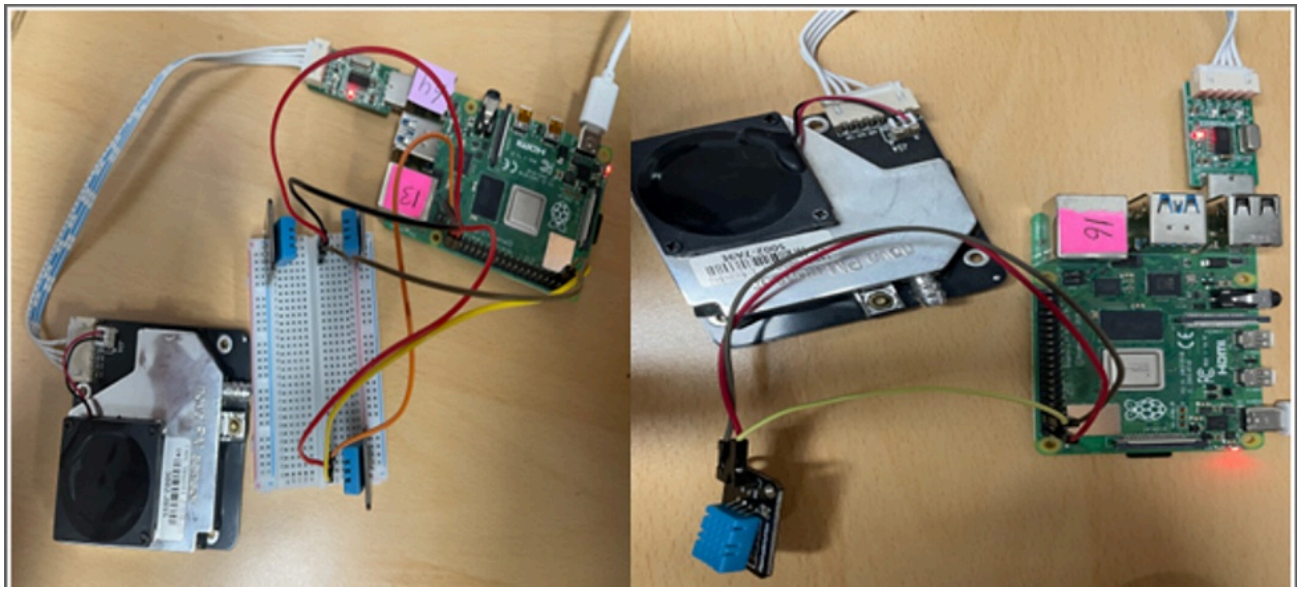


Figure 7. Experimental configuration of a Raspberry Pi device.

The concrete implementation parameters of the empirical environment are summarized in **Table 4**, confirming the utilization of Raspberry Pi hardware nodes for execution. Throughout the cooperative learning operations, the TCP network protocol manages inter-device data exchanges. The core DFL architecture is entirely constructed within the Python programming ecosystem.

Digital twin network for decentralised federated learning

This investigation carries out empirical tests utilizing a digital twin network representation of the underlying physical hardware. The primary objective of this digital twin system is to oversee the operational health and status of edge units deployed across the smart home infrastructure. While the actual physical modules operate within residential environments, a virtual replication of the Raspberry Pi devices is maintained on a centralized coordinator platform, which tracks and manages the remote nodes. To bridge communications between physical hardware and the central coordinator, a dedicated digital twin assistant application was engineered. This application runs locally on each participant node and relies on the HTTP protocol to transmit updates to the coordinator machine. Through this central interface, operators can oversee edge device states—including tracking active or inactive statuses—and review real-time cooperative training diagnostics.

Working details of the digital twin network

This framework adopts the digital twin network model introduced by Rizwan *et al*. [54], which delivers foundational services including edge client registration, model initialization, and the configuration of machine learning hyperparameters and DFL orchestration parameters. The collaborative interactions governing this workflow are diagrammed in **Figure 8.**

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

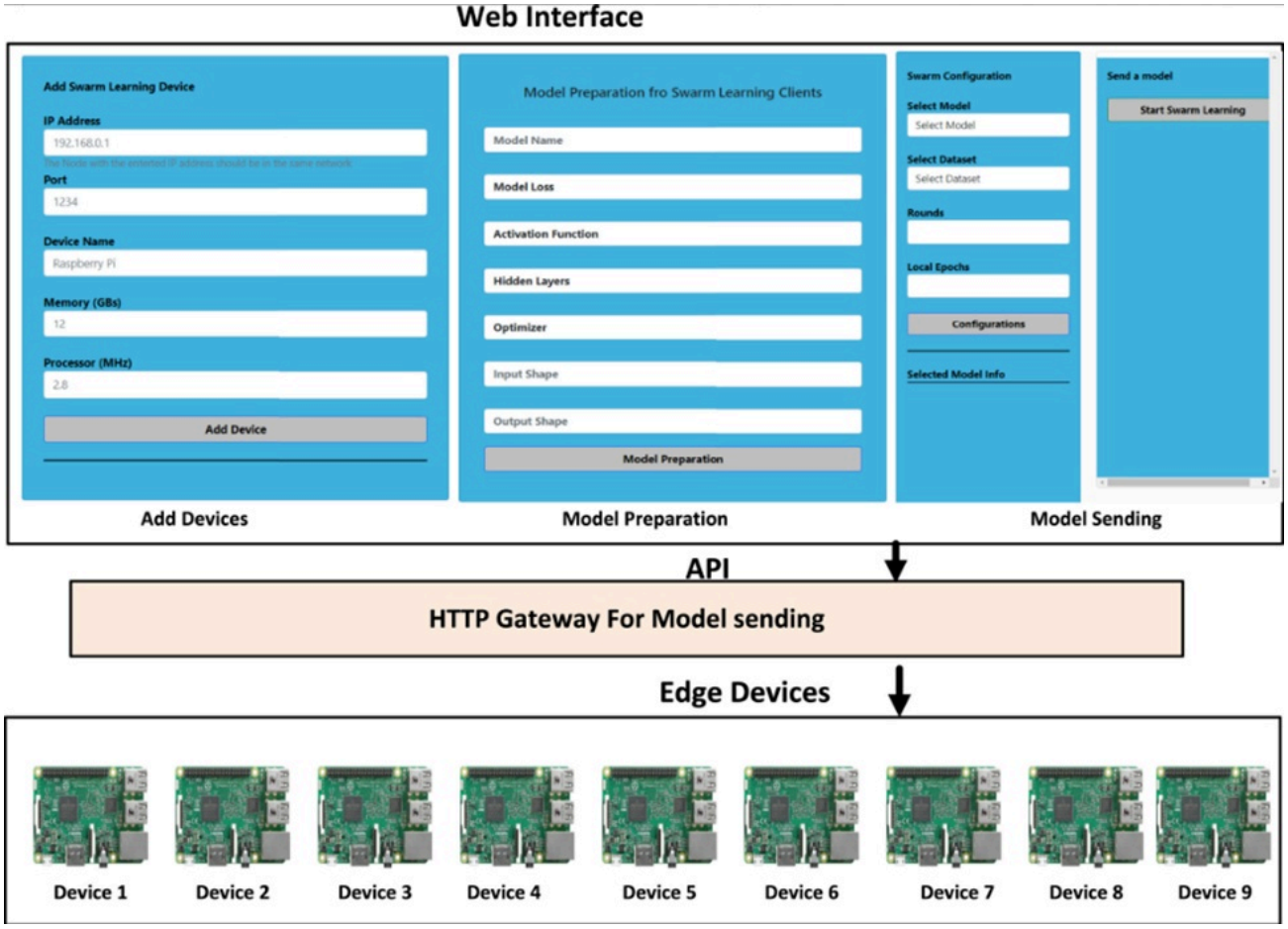


Figure 8. Collaboration diagram of a web interface of a digital twin.

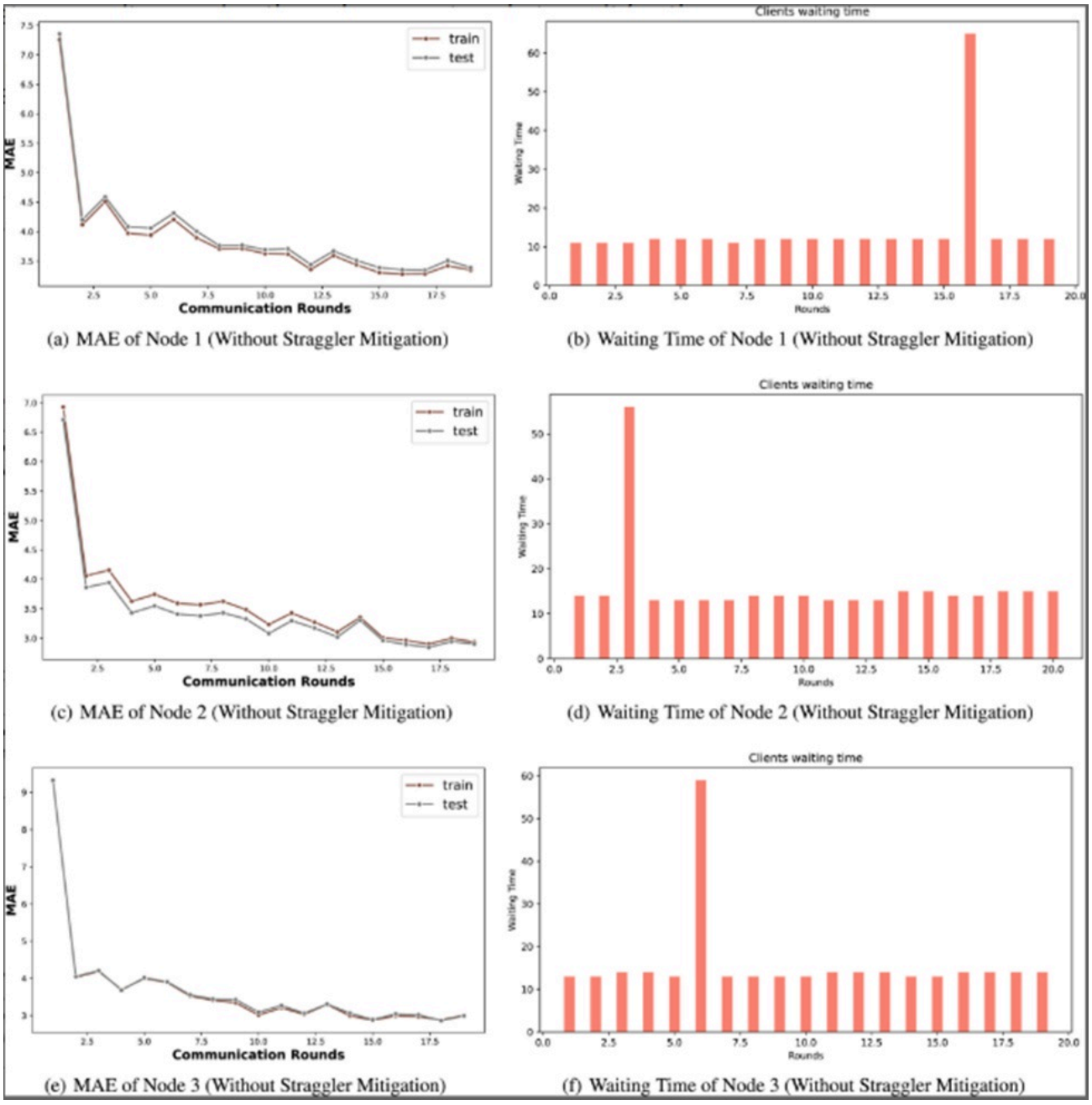
The interactive collaboration sequence for the digital twin web dashboard is presented in **Figure 8**, highlighting the administrative tasks involved in enrolling new devices into the DFL architecture and provisioning the base global model. Upon adding a node to the network, its IP address, port configuration, and hardware capabilities are logged into a central database. Through the model provisioning dashboard, operators define structural specifications such as the total count of hidden layers, specific activation functions for hidden layers, and the output layer configuration. Additionally, operational parameters like localized epochs and global communication limits are established. Once configured, these operational schemas are distributed to all enrolled participant devices.

Experimental results of decentralised federated learning framework

The mean absolute error profiles and corresponding idle waiting intervals for all participating nodes are displayed in **Figure 9.** Throughout the cooperative optimization timeline, several nodes experience transient variations in their MAE metrics, exhibiting minor upward and downward trends. Nonetheless, **Figure 9** confirms a consistent, stable convergence pattern across all network participants over consecutive communication steps, culminating in an approximate MAE value of 3% across 20 communication rounds. At the onset of optimization, the error metric opened at 7 during the initial phase, steadily decaying with each successive communication interval. These empirical outcomes suggest that merging an incoming external model structure with a node's localized weights does not degrade predictive accuracy on local test subsets; rather, it actively supports individual node convergence. Within this sequential topology, hardware and statistical discrepancies cause sporadic transmission delays among certain devices, forcing downstream nodes to halt processing while awaiting updates from upstream predecessors. The sequence from sub- **Figure 9b** through **Figure 9p** illustrates

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

that these waiting intervals regularly exceed the acceptable performance threshold at multiple points, ultimately slowing the joint learning process.



(a) MAE of Node 1 (Without Straggler Mitigation)

(b) Waiting Time of Node 1 (Without Straggler Mitigation)

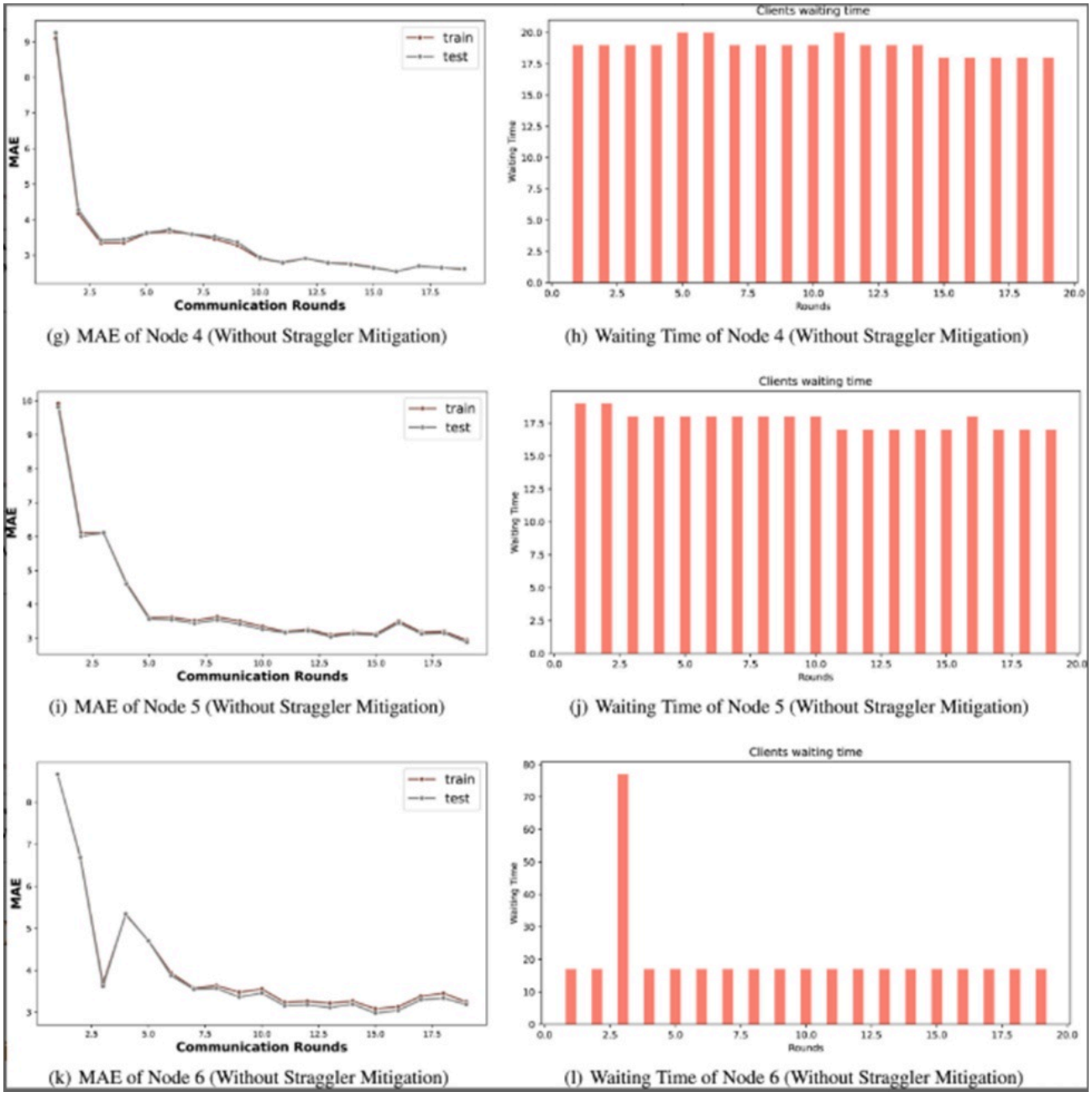
(c) MAE of Node 2 (Without Straggler Mitigation)

(d) Waiting Time of Node 2 (Without Straggler Mitigation)

(e) MAE of Node 3 (Without Straggler Mitigation)

(f) Waiting Time of Node 3 (Without Straggler Mitigation)

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press



(g) MAE of Node 4 (Without Straggler Mitigation)

(h) Waiting Time of Node 4 (Without Straggler Mitigation)

(i) MAE of Node 5 (Without Straggler Mitigation)

(j) Waiting Time of Node 5 (Without Straggler Mitigation)

(k) MAE of Node 6 (Without Straggler Mitigation)

(l) Waiting Time of Node 6 (Without Straggler Mitigation)

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

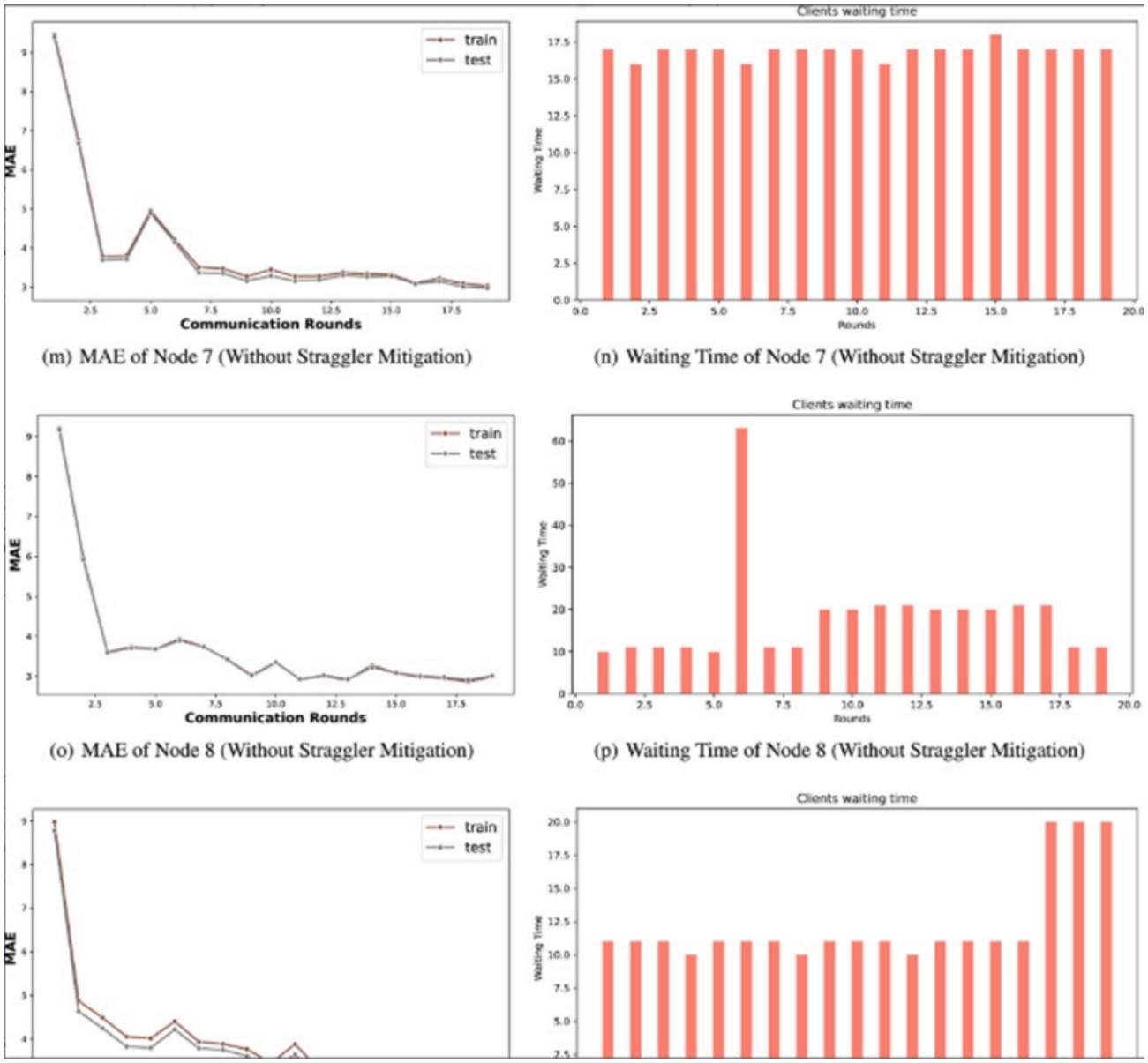


Figure 9. Mean absolute error and waiting time of participating nodes for energy consumption prediction.

The quantitative metrics generated by the decentralized learning architecture for forecasting energy demand are detailed in **Table 5**. With an LSTM network deployed locally at each participant node, the specific accuracy metrics of each configuration are evaluated. The documented performance values represent outcomes derived from MAE, MAPE, RMSE, and MAE validation criteria recorded during the concluding communication round. The breakdown of individual error metrics for every node at the final communication round is displayed. The mean squared error for node 4 is lower than that of any other network participant. Conversely, node 6 exhibits the highest mean squared error among all edge nodes.

Regarding the MAPE and RMSE metrics, node 4 sustains the most optimal error reduction profile among all clients. The operational performance of each node is assessed across both training and testing datasets. This uniform alignment between training and testing errors across the node collective indicates that the predictive models are stable and free from overfitting.

Table 5. Experimental results of decentralised federated learning framework for energy consumption prediction.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

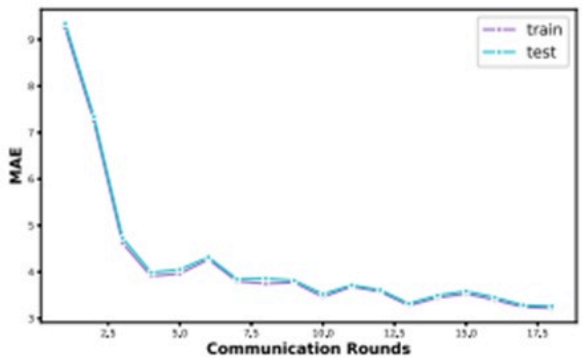
Nodes	Test_rmse	Train_rmse	Test_mape	Train_mape	Test_mae	Train_mae	Test_mse	Train_mse
Node 1	4.7746	4.7737	0.4099	0.4089	3.3897	3.3453	22.7970	22.7885
Node 2	4.3507	4.5438	0.3965	0.3965	2.9018	2.9345	18.9289	20.6465
Node 3	4.9793	5.0058	0.3107	0.3184	2.9930	3.0097	24.7940	25.0586
Node 4	3.6327	3.7922	0.3413	0.3641	2.6242	2.6030	13.1967	14.3813
Node 5	4.0311	4.2401	0.2929	0.2999	2.8832	2.9425	16.2498	17.9786
Node 6	5.7097	5.6863	0.4024	0.4307	3.1897	3.2495	32.6007	32.3349
Node 7	4.4533	4.4810	0.2786	0.2946	2.9794	3.0376	19.8325	20.0802
Node 8	4.4740	4.5797	0.3145	0.3144	3.0117	2.9928	20.0171	20.9742
Node 9	4.5733	4.7332	0.3284	0.3568	3.1611	3.2119	20.9154	22.4032

Experimental results of decentralised federated learning framework with straggler mitigation

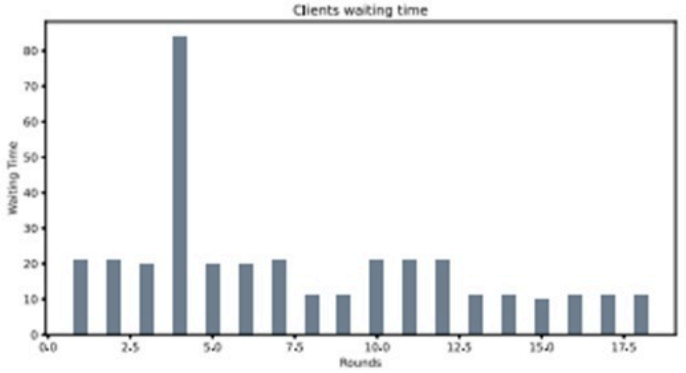
A straggler is defined as an underperforming node that requires additional time to transmit its model to the subsequent device. The existence of such a node impedes the overall collaborative learning procedure. As illustrated in **Figure 9**, certain nodes experience extended idle periods during specific communication rounds, consequently forcing the subsequent node into a waiting state. To address this, the current study introduces a mechanism designed to alleviate the impact of stragglers. This proposed technique neutralizes the straggler node for the subsequent communication cycle. During the following round, the device typically reliant on the straggler will instead acquire the model directly from the node preceding the straggler.

The empirical outcomes of the energy consumption forecasting DFL framework, incorporating this straggler mitigation strategy, are presented in **Figure 10.** The data indicates that nodes 1, 3, 4, 5, 6, 7, and 8 experience waiting intervals exceeding the standard average during communication rounds 4, 6, 9, 12, 14, and 16, respectively. The specific nodes connected to these delayed participants are designated as stragglers; thus, in the subsequent iteration, the affected nodes request model transmission from the preceding node in the sequence. In **Figure 10**, subfigure 10b shows that the waiting duration for node 1 exceeds the threshold during round 4, leading it to receive the model directly from node 8 in the subsequent round. Subfigure 10a further reveals that this alternative model acquisition from node 8 does not alter node 1's convergence trajectory, nor does it degrade its predictive performance. Throughout this experimental phase, seven distinct straggler instances were identified across various rounds. However, by neutralizing each straggler for the following communication cycle, the proposed mitigation technique successfully fostered collaborative model convergence while simultaneously decreasing the total required learning time. This outcome is rooted in the sequential nature of the ring topology, where every node receives model parameters sequentially from its predecessor. Consequently, each participant becomes adapted to the preceding node's data patterns. When a model is acquired from an earlier node in the sequence, it does not drastically disrupt the receiving node's convergence pattern. Based on **Figure 10**, it can be concluded that applying the straggler mitigation technique yields a Mean Absolute Error (MAE) of approximately 3 across all nodes for energy consumption forecasting, with the convergence trajectories remaining tightly aligned throughout the training phase.

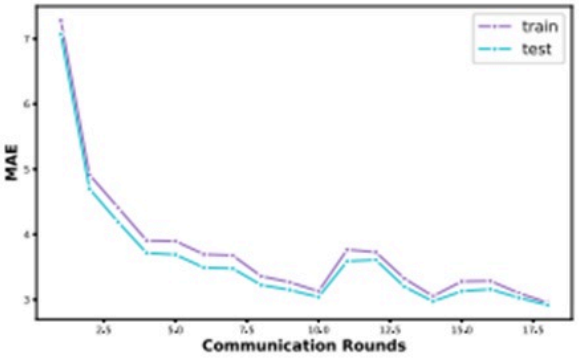
© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press



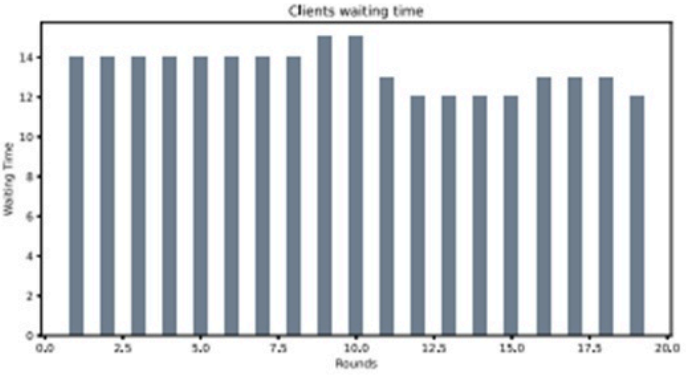
(a) MAE of Node 1 With Straggler Node Mitigation



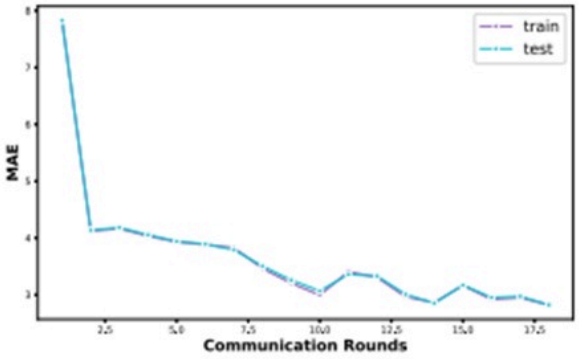
(b) Waiting Time of Node 1 With Straggler Node Mitigation



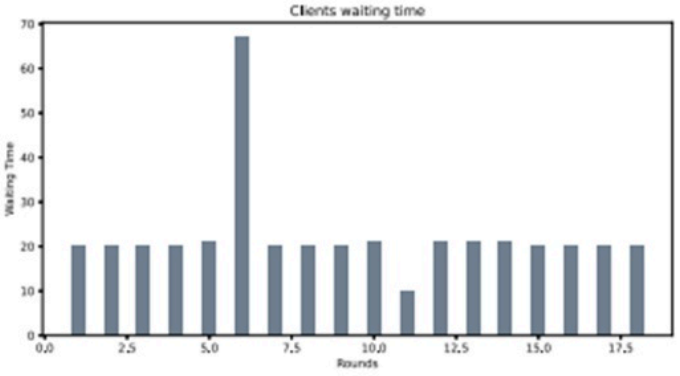
(c) MAE of Node 2 With Straggler Node Mitigation



(d) Waiting Time of Node 2 With Straggler Node Mitigation

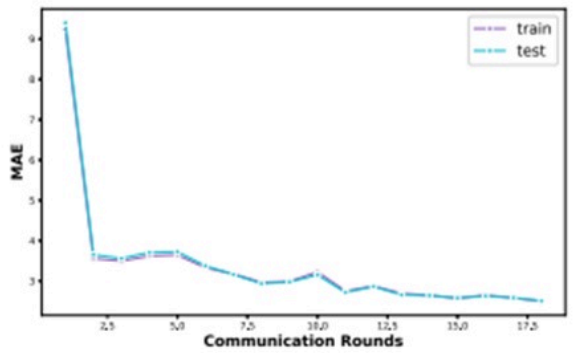


(e) MAE of Node 3 With Straggler Node Mitigation

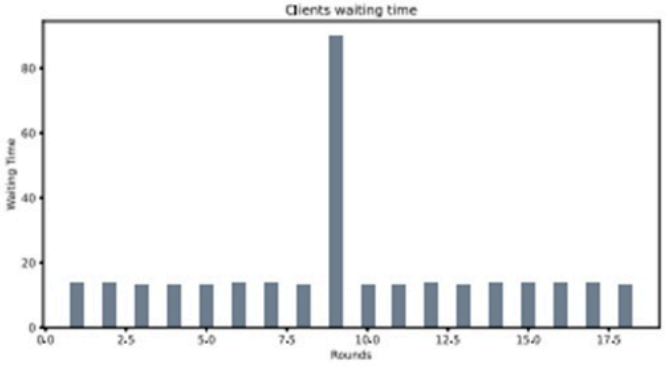


(f) Waiting Time of Node 3 With Straggler Node Mitigation

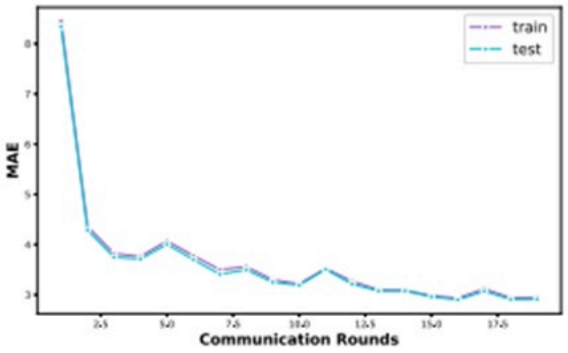
© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press



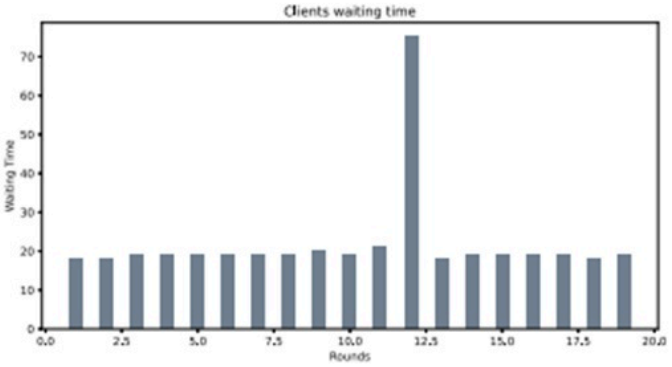
(g) MAE of Node 4 With Straggler Node Mitigation



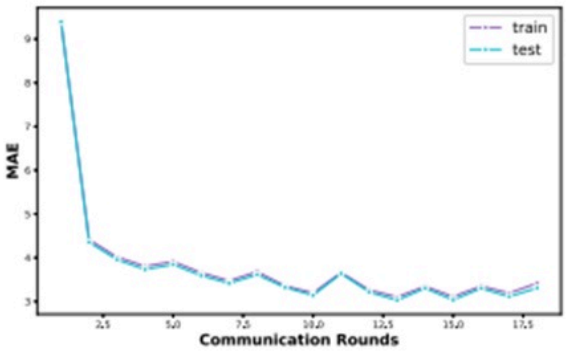
(h) Waiting Time of Node 4 With Straggler Node Mitigation



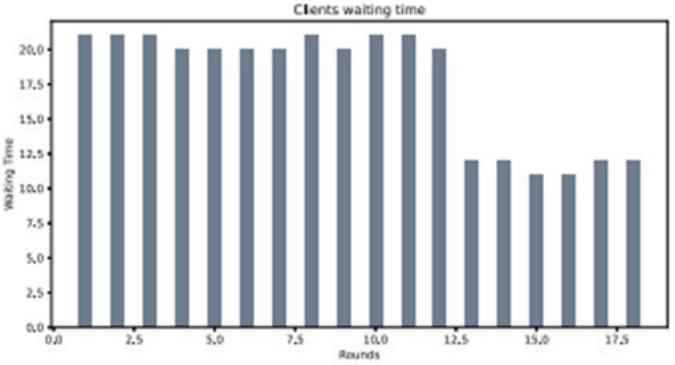
(i) MAE of Node 5 With Straggler Node Mitigation



(j) Waiting Time of Node 5 With Straggler Node Mitigation

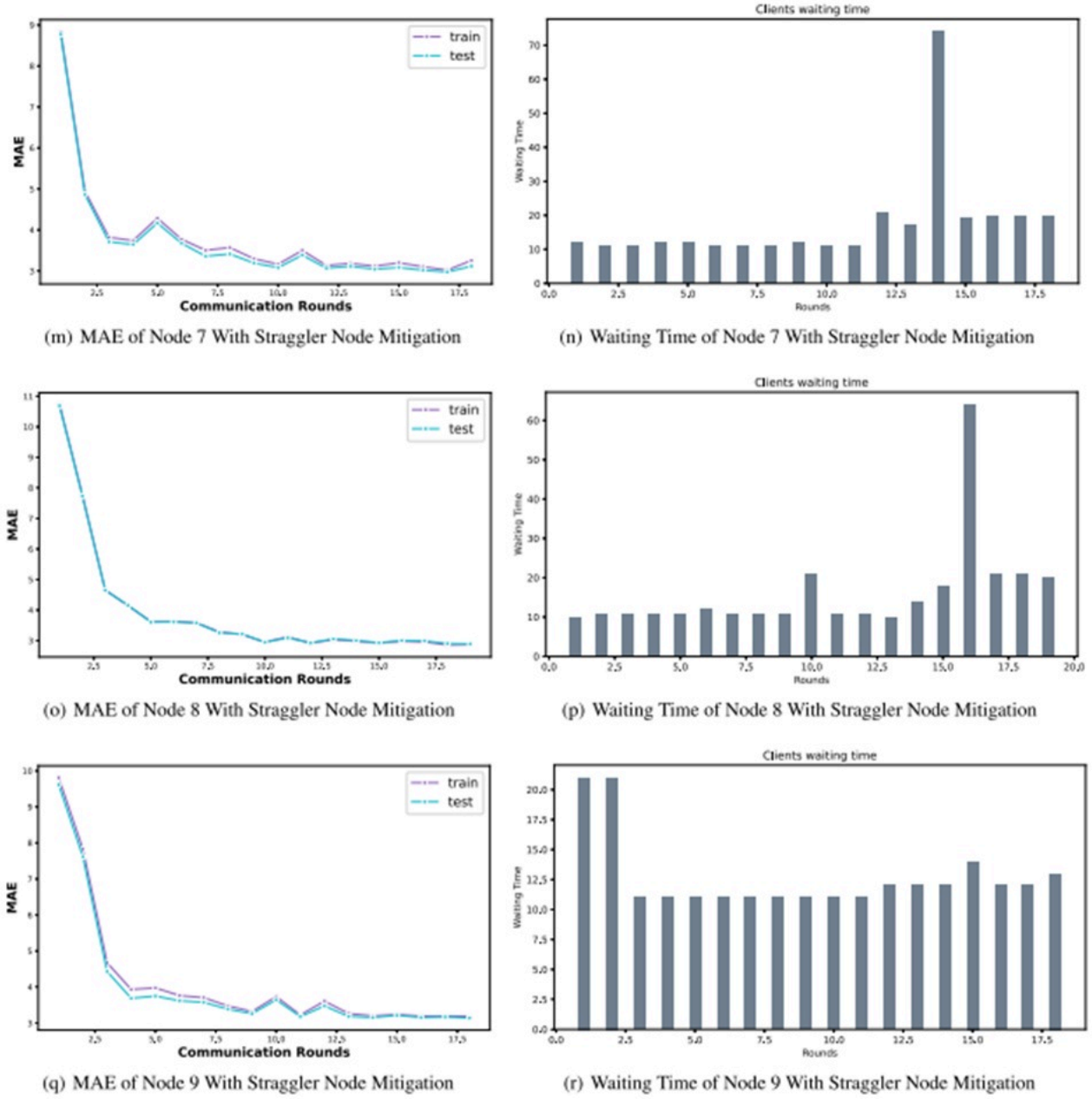


(k) MAE of Node 6 With Straggler Node Mitigation



(l) Waiting Time of Node 6 With Straggler Node Mitigation

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press



(m) MAE of Node 7 With Straggler Node Mitigation

(n) Waiting Time of Node 7 With Straggler Node Mitigation

(o) MAE of Node 8 With Straggler Node Mitigation

(p) Waiting Time of Node 8 With Straggler Node Mitigation

(q) MAE of Node 9 With Straggler Node Mitigation

(r) Waiting Time of Node 9 With Straggler Node Mitigation

Figure 10. Mean absolute error and waiting time of all nodes for energy consumption prediction with straggler node mitigation.

The analytical results of the enhanced DFL architecture utilizing the straggler mitigation protocol are detailed in **Table 6**. The primary objective of this integration was to minimize the detrimental effects of delayed nodes without sacrificing forecasting accuracy. Consistent with the prior evaluations detailed in **Table 5**, these results were generated utilizing the LSTM architecture, but with the active suppression of straggler nodes. The empirical findings reveal that the energy forecasting accuracy across all participating nodes remains comparable to the baseline results in **Table 5**, specifically concerning MAPE, MSE, MAE, and RMSE metrics. Node 4 demonstrates the lowest incidence of prediction errors, whereas node 6 registers the highest error rate relative to its peers. The fundamental goal of this research was to suppress straggler interference without compromising the predictive model's overall efficacy. These findings substantiate the proposed model's capacity to sustain reliable prediction accuracy while successfully optimizing communication efficiency.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

Table 6. Experimental results of decentralised federated learning framework for energy consumption prediction with straggler mitigation.

Nodes	Test_rmse	Train_rmse	Test_mape	Train_mape	Test_mae	Train_mae	Test_mse	Train_mse
Node 1	4.6930	4.6658	0.3879	0.3873	3.2622	3.2186	22.0248	21.7701
Node 2	4.2877	4.4000	0.4112	0.4111	2.9156	2.9536	18.3847	19.3602
Node 3	4.1392	4.2004	0.2972	0.3022	2.8230	2.8205	17.1333	17.6438
Node 4	3.5550	3.7108	0.3171	0.3417	2.5070	2.4941	12.6383	13.7704
Node 5	4.1317	4.2921	0.2904	0.29465	2.9121	2.9467	17.0716	18.4224
Node 6	6.6261	6.8924	0.4074	0.4436	3.3086	3.4160	43.9057	47.5063
Node 7	5.3887	6.2542	0.2788	0.2981	3.1158	3.2539	29.0381	39.1161
Node 8	4.5291	4.7428	0.2872	0.2896	2.8913	2.8741	20.5135	22.4943
Node 9	4.5980	4.8087	0.3198	0.3461	3.1404	3.1893	21.1420	23.1245

Comparison of existing methods

Within the current body of literature, only a limited number of investigations have utilized this specific dataset for energy consumption forecasting. The findings from these comparable studies are detailed in **Table 7.**

Table 7. Comparative analysis with existing methods.

Contemporary studies	Reported MAE
Non-FL methods	
Spatial-temporal clustering [18]	572.693
Ensemble model [19]	373.580
FL methods	
Centralised FL-based method [54]	3.3 ± 0.1
Proposed method	2.5070

A comparative evaluation of various technical approaches applied to the identical energy consumption dataset is provided in **Table 7**: prior investigations, notably Khan *et al*. [18, 19], utilized traditional centralized machine learning architectures, while Rizwan *et al*. [54] proposed a Clustered Federated Learning (CFL) framework. The methodologies documented in Khan *et al.* [18, 19] face inherent vulnerabilities regarding data privacy. Conversely, the framework presented by Rizwan *et al*. [54] suffers from operational bottlenecks and the risk of a single point of failure.

To resolve these inherent challenges, the present research introduces an innovative DFL architecture based on a ring topology, augmented with an adaptive straggler mitigation protocol, effectively circumventing the constraints observed in prior studies. The primary advancement of our proposed methodology is its capacity to substantially lower prediction errors to 2.570, down from a comparative baseline of 3.3. This directly resolves critical issues including delayed model convergence caused by straggling devices, data privacy vulnerabilities, and centralized failure points. Additionally, this

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

methodology fosters superior personalized modeling, mitigates the instability caused by sudden model parameter shifts, and optimizes holistic network communication efficiency.

The empirical data presented in **Table 7** validates the superiority of our introduced framework and underscores its overall efficacy. Furthermore, **Tables 5 and 6** provide additional substantiation of the DFL framework's advantages in forecasting energy consumption, specifically illustrating performance outcomes both prior to and following the integration of the straggler mitigation mechanism.

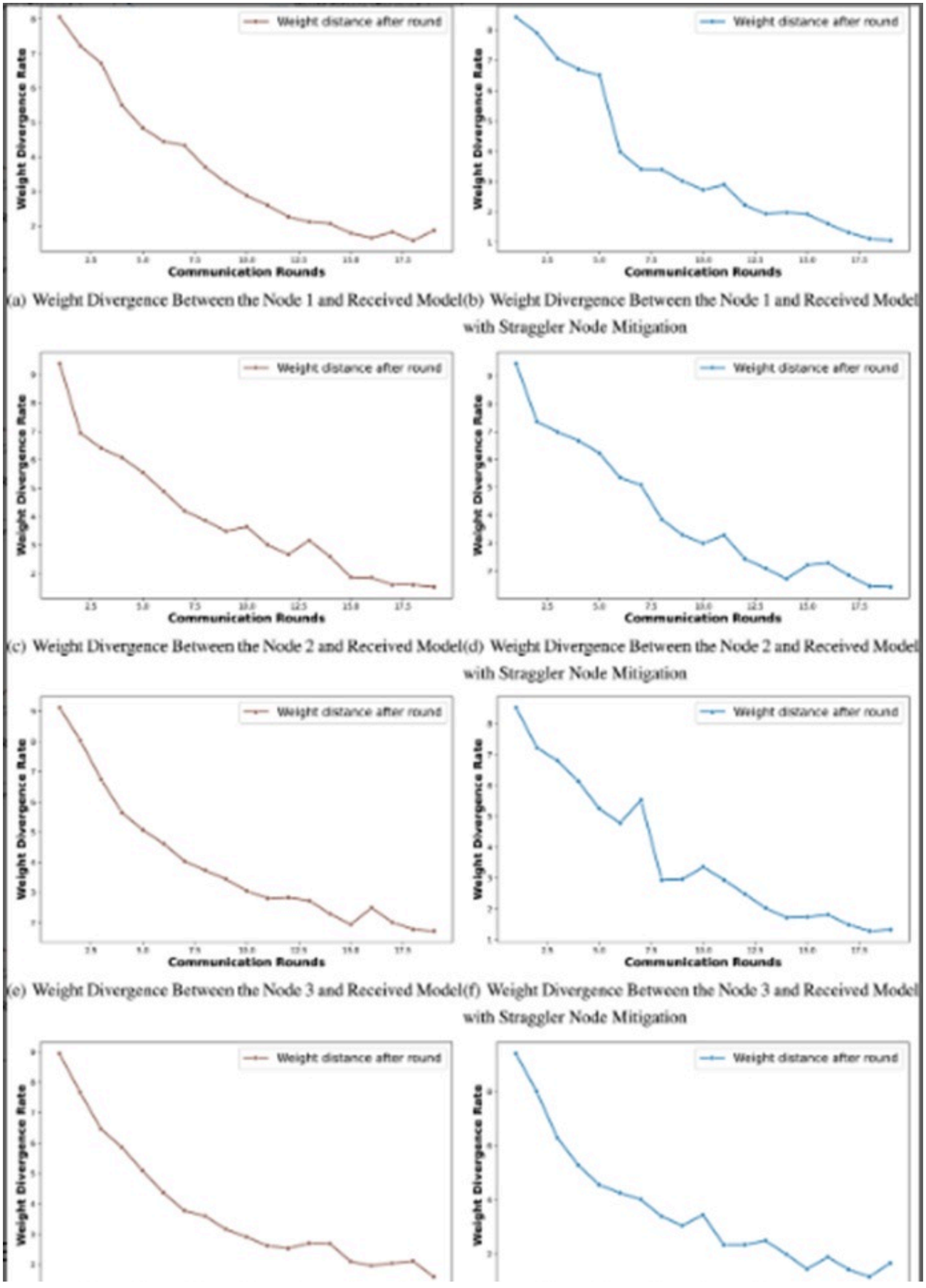
Weight divergence calculation between local and received model

Upon acquiring a model from a preceding node, the receiving device calculates the weight divergence of the models using **Eq. 9**. The theoretical expectation is that this weight divergence metric should remain low, and the overarching trajectory of divergence should exhibit a downward trend over time.

$$d_{r,l}(i) = \left\| W_r(i) - W_l(i) \right\| / \left\| W_l(i) \right\| \quad (9)$$

Where Wr(i) and Wl(i) are the weights of the received and local models at the ith iteration, and dr,l (i) is a weight divergence value at the ith communication round. This research evaluates weight divergence before aggregating local and incoming model weights to assess the degree of similarity between the received parameter patterns and the localized model. The graphical representation of the weight divergence between the local and acquired models is provided in **Figure 11**. The objective of this visual plot is to track the evolving pattern of weight disparities. As demonstrated in **Figure 11**, the magnitude of difference between the local and received models progressively diminishes across successive communication rounds. This indicates that integrating the incoming model improves the accuracy of energy consumption predictions. Following the initial communication cycle, the recorded divergence distance across all nodes ranges between 8.5 and 9. However, by the 19th communication round, this metric significantly contracts to a range of 1 to 1.4. Furthermore, **Figure 11** illustrates that the overall trend of weight divergence remains highly consistent regardless of whether straggler mitigation is actively applied.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

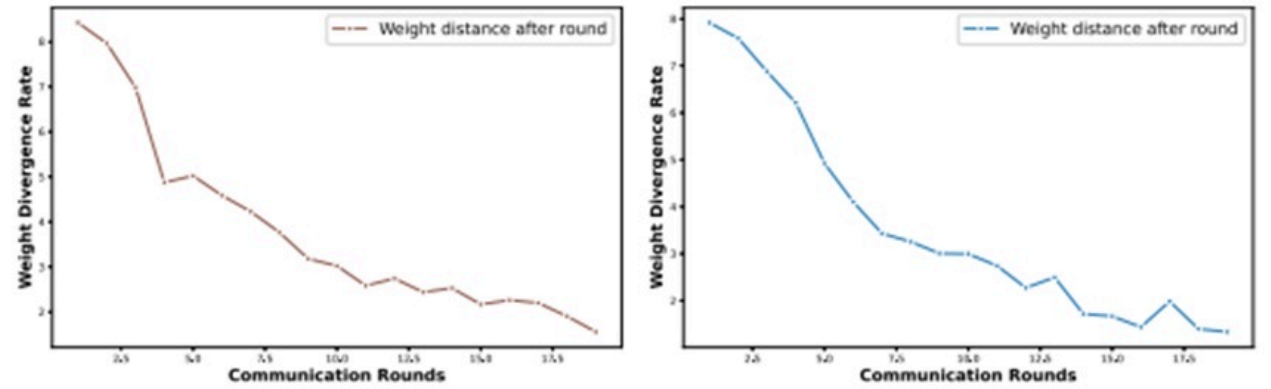


(a) Weight Divergence Between the Node 1 and Received Model(b) Weight Divergence Between the Node 1 and Received Model with Straggler Node Mitigation

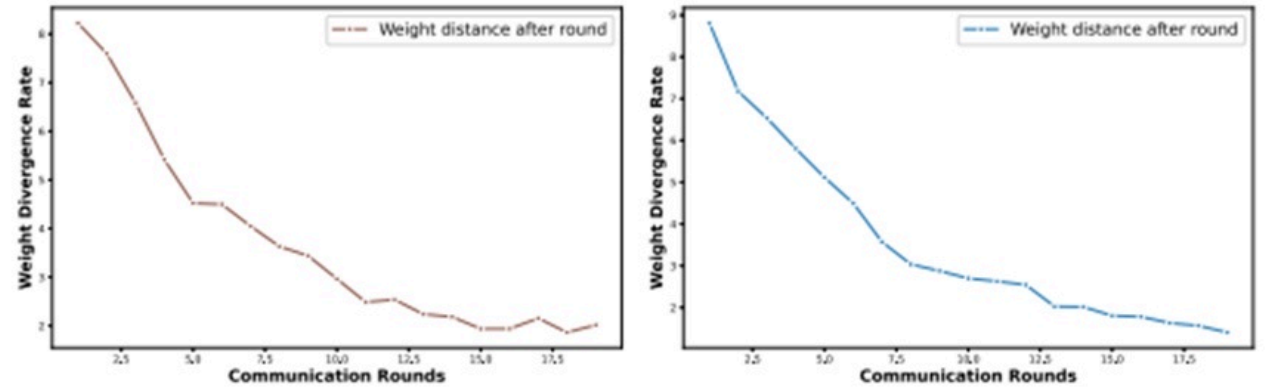
(c) Weight Divergence Between the Node 2 and Received Model(d) Weight Divergence Between the Node 2 and Received Model with Straggler Node Mitigation

(e) Weight Divergence Between the Node 3 and Received Model(f) Weight Divergence Between the Node 3 and Received Model with Straggler Node Mitigation

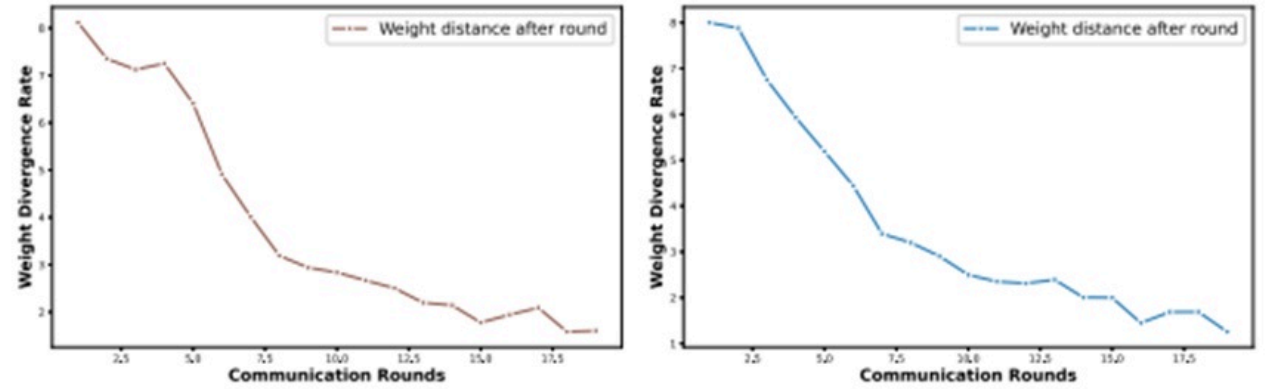
© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press



(i) Weight Divergence Between the Node 5 and Received Model (j) Weight Divergence Between the Node 5 and Received Model with Straggler Node Mitigation



(k) Weight Divergence Between the Node 6 and Received Model (l) Weight Divergence Between the Node 6 and Received Model with Straggler Node Mitigation



(m) Weight Divergence Between the Node 7 and Received Model (n) Weight Divergence Between the Node 7 and Received Model with Straggler Node Mitigation

Figure 11. (continued) Weight divergence between the local and received model for straggler node mitigation with and without straggler node mitigation.

Conclusion

This study introduced a Decentralized Federated Learning (DFL) architecture designed to forecast energy consumption while rigorously preserving data privacy. As an initial step, the predictive performance of the model was evaluated independently of the latency and operational inefficiencies introduced by straggler nodes. Establishing this baseline was

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

essential for validating the framework's baseline efficacy and providing a reference point for subsequent enhancements. Subsequently, we integrated a targeted strategy designed to minimize straggler interference, thereby improving the synchronization and communication efficiency among network participants. The proposed framework, operating with straggler mitigation, relies on an adaptive, neighbor-oriented strategy. Experimental analyses confirm that this mitigation technique effectively reduces operational wait times while fully preserving the model's forecasting accuracy, making it highly suitable for real-time, collaborative energy prediction in smart building environments. Because the structural design eliminates the need for a centralized server for weight aggregation, it represents a highly scalable and cost-efficient solution for modern energy management infrastructures. Given its demonstrated efficacy in energy forecasting, this methodology exhibits substantial potential for broad implementation across intelligent built environments. Future research will explore alternative network topologies and develop personalized aggregation protocols to refine decentralized model training further.

Acknowledgements

None

Conflict of interest

None

Financial support

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (No. 2022-0-00438, Development of an operation and prediction core technology for federated intelligent digital twin). Please address any correspondence related to this paper to Do Hyeun Kim.

Ethics statement

None

Received: 14 Feb 2023 Revised: 22 Apr 2023 Accepted: 09 May 2023
Published online: 15 June 2024

Rights and permissions

Open Access The author(s) retain copyright. This article is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. It may be shared and adapted for non-commercial purposes with appropriate attribution, an indication of changes, and distribution of adaptations under the same license. Third-party material may be subject to separate terms identified in its credit line. View the license at https://creativecommons.org/licenses/by-nc-sa/4.0/.

References

1. Hu S, Yan D, Guo S, Cui Y, Dong B. A survey on energy consumption and energy usage behavior of households and residential building in urban China. Energy Build. 2017;148:366-78.

https://doi.org/10.1016/j.enbuild.2017.03.064.

2. Berardi U, Jafarpur P. Assessing the impact of climate change on building heating and cooling energy demand in Canada. Renew Sustain Energy Rev. 2020;121:109681. https://doi.org/10.1016/j.rser.2019.109681.

3. Zhao D, Zhou Q, Liu Y, Xu X, Li H. Interaction effects of building technology and resident behavior on energy consumption in residential buildings. Energy Build. 2017;134:223-33. https://doi.org/10.1016/j.enbuild.2016.10.049.

4. Martins F, Felgueiras C, Smitková M, Caetano N. Analysis of fossil fuel energy consumption and environmental impacts in European countries. Energies. 2019;12(6):964. https://doi.org/10.3390/en12060964.

5. Surahman U, Hartono D, Setyowati E, Adib M, Sulistiawati. Investigation on household energy consumption of urban residential buildings in major cities of Indonesia during Covid 19 pandemic. Energy Build. 2022;261:111956. https://doi.org/10.1016/j.enbuild.2022.111956.

6. Khalil M, Berawi MA, Heryanto R, Rizalie A. Waste to energy technology: the potential of sustainable biogas production from animal waste in Indonesia. Renew Sustain Energy Rev. 2019;105:323-31. https://doi.org/10.1016/j.rser.2019.02.011.

7. Liu Y, Chen H, Zhang L, Wu X, Wang X. Energy consumption prediction and diagnosis of public buildings based on support vector machine learning: a case study in China. J Clean Prod. 2020;272:122542. https://doi.org/10.1016/j.jclepro.2020.122542.

8. Mishuchkov V, Pushkareva M, Belov S. Smart metering and scheduling of electrical loads of buildings. J Phys Conf Ser. 2021;2096(1):012132. 6596/2096/1/012132. https://doi.org/10.1088/1742.

9. Ullah A, Khan N, Ahmed I, Ali A, Khan S. Deep learning assisted buildings energy consumption profiling using smart meter data. Sensors. 2020;20(3):873. https://doi.org/10.3390/s20030873.

10. Savi M, Olivadese F. Short term energy consumption forecasting at the edge: a federated learning approach. IEEE Access. 2021;9:95949-69. https://doi.org/10.1109/access.2021.3094089.

11. Nightingale JS, Hong J, Park H, Kim Y, Lee S. Effect of clustering in federated learning on non iid electricity consumption prediction. In: 2022 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT Europe). IEEE; 2022. p. 1-5.

12. Mendes N, Faria P, Vale Z, Ramos C. Federated learning enabled prediction of energy consumption in transactive energy communities. In: 2022 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT Europe). IEEE; 2022. p. 1-5.

13. Li J, Wang Y, Zhang X, Chen H, Liu Z. Federated learning based short term building energy consumption prediction method for solving the data silos problem. Build Simul. 2022;15(6):1145-59. https://doi.org/10.1007/s12273-021-0871-y.

14. Tang L, Liu Y, Zhang W, Chen J, Wang S. Privacy preserving knowledge sharing for few shot building energy prediction: a federated learning approach. Appl Energy. 2023;337:120860. https://doi.org/10.1016/j.apenergy.2023.120860.

15. Tun YL, Thwal CM, Kang JM, Park S. Federated learning based energy demand prediction with clustered aggregation. In: 2021 IEEE International Conference on Big Data and Smart Computing (BigComp). IEEE; 2021. p. 164-7.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

16. Elbeltagi E, Wefki H. Predicting energy consumption for residential buildings using ANN through parametric modeling. Energy Rep. 2021;7:2534-45. https://doi.org/10.1016/j.egyr.2021.04.053.

17. Olu Ajayi R, Alaka H, Owolabi H, Ajayi S, Oyedele L. Building energy consumption prediction for residential buildings using deep learning and other machine learning techniques. J Build Eng. 2022;45:103406. https://doi.org/10.1016/j.jobe.2021.103406.

18. Khan AN, Ahmad N, Ali A, Khan S, Khan W. An ensemble energy consumption forecasting model based on spatial temporal clustering analysis in residential buildings. Energies. 2021;14(11):3020. https://doi.org/10.3390/en14113020.

19. Khan AN, Ahmad N, Ali A, Khan S, Khan W. Ensemble prediction approach based on learning to statistical model for efficient building energy consumption management. Symmetry. 2021;13(3):405. https://doi.org/10.3390/sym13030405.

20. Khan QW, Ahmad N, Ali A, Khan S, Raza M. Decentralized machine learning training: a survey on synchronization, consolidation, and topologies. IEEE Access. 2023;11:68031-50. https://doi.org/10.1109/access.2023.3284976.

21. Banabilah S, Aloqaily M, Alsayed E, Jararweh Y. Federated learning review: fundamentals, enabling technologies, and future applications. Inf Process Manag. 2022;59(6):103061. https://doi.org/10.1016/j.ipm.2022.103061.

22. Wang R, Tsai WT. Asynchronous federated learning system based on permissioned blockchains. Sensors. 2022;22(4):1672. https://doi.org/10.3390/s22041672.

23. Abreha HG, Hayajneh M, Serhani MA. Federated learning in edge computing: a systematic survey. Sensors. 2022;22(2):450. https://doi.org/10.3390/s22020450.

24. Zhao J, Chen Y, Zhang L, Wang H, Li X. Pvd fl: a privacy preserving and verifiable decentralized federated learning framework. IEEE Trans Inf Forensics Secur. 2022;17:2059-73. https://doi.org/10.1109/tifs.2022.3176191.

25. Gupta R, Alam T. Survey on federated learning approaches in distributed environment. Wireless Pers Commun. 2022;125(2):1631-52. https://doi.org/10.1007/s11277-022-09624-y.

26. Hublet F, Basin D, Krstić S. Enforcing the GDPR. In: European Symposium on Research in Computer Security. Springer; 2023. p. 400-22.

27. Fekri MN, Grolinger K, Mir S. Distributed load forecasting using smart meter data: federated learning with recurrent neural networks. Int J Electr Power Energy Syst. 2022;137:107669. https://doi.org/10.1016/j.ijepes.2021.107669.

28. Badr MM, Eltoweissy M, ElSayed M, ElSayed M. Privacy preserving federated learning based net energy forecasting. In: SoutheastCon 2022. IEEE; 2022. p. 133-9.

29. Briggs C, Fan Z, Andras P. Federated learning for short term residential load forecasting. IEEE Open Access J Power Energy. 2022;9:573-83. https://doi.org/10.1109/oajpe.2022.3206220.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

30. Gholizadeh N, Musilek P. Federated learning with hyperparameter based clustering for electrical load forecasting. Internet Things. 2022;17:100470. https://doi.org/10.1016/j.iot.2021.100470.

31. Zhang G, Zhu S, Bai X. Federated learning based multi energy load forecasting method using cnn attention lstm model. Sustainability. 2022;14(19):12843. https://doi.org/10.3390/su141912843.

32. Petrangeli E, Tonellotto N, Vallati C. Performance evaluation of federated learning for residential energy forecasting. IoT. 2022;3(3):381-97. https://doi.org/10.3390/iot3030021.

33. Wang Y, Gao N, Hug G. Personalized federated learning for individual consumer load forecasting. CSEE J Power Energy Syst. 2022. https://doi.org/10.17775/CSEEJPES.2021.09120.

34. Shi Y, Xu X. Deep federated adaptation: an adaptative residential load forecasting approach with federated learning. Sensors. 2022;22(9):3264. https://doi.org/10.3390/s22093264.

35. Gao J, Wang H, Chen Y, Zhang L, Liu X. Decentralized federated learning framework for the neighborhood: a case study on residential building load forecasting. In: Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems. 2021. p. 453-9.

36. Park J, Kim Y, Lee J, Choi W, Han S. Amble: adjusting mini batch and local epoch for federated learning with heterogeneous devices. J Parallel Distrib Comput. 2022;170:13-23. https://doi.org/10.1016/j.jpdc.2022.07.009.

37. Tziotis I, Papadopoulos S, Koutsopoulos I, Karras P. Straggler resilient personalized federated learning. arXiv. 2022. arXiv:2206.02078.

38. Reisizadeh A, Mokhtari A, Hassani H, Jadbabaie A, Pedarsani P. Straggler resilient federated learning: leveraging the interplay between statistical accuracy and system heterogeneity. IEEE J Sel Areas Inf Theory. 2022;3(2):197-205. https://doi.org/10.1109/jsait.2022.3205475.

39. Wolfrath J, Sarker S, Li H, Zhang Y, Chen X. Haccs: heterogeneity aware clustered client selection for accelerated federated learning. In: 2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS). IEEE; 2022. p. 985-95.

40. Sultana K, Rahman M, Islam M, Khan S, Ahmed T. Elastic optimized edge federated learning. In: 2022 International Conference on Networking and Network Applications (NaNA). IEEE; 2022. p. 288-94.

41. Elzohairy M, Abdelkader H, El Gohary A, El Sayed M. Fedlesscan: mitigating stragglers in serverless federated learning. arXiv. 2022. arXiv:2211.05739. https://doi.org/10.1109/bigdata55660.2022.10021037.

42. Xie C, Koyejo S, Gupta I. Asynchronous federated optimization. arXiv. 2019. arXiv:1903.03934.

43. Tao Y, Zhou J. Straggler remission for federated learning via decentralized redundant cayley tree. In: 2020 IEEE Latin American Conference on Communications (LATINCOM). IEEE; 2020. p. 1 6.

44. Park J, Kim Y, Lee S, Choi W, Han S. Sageflow: robust federated learning against both stragglers and adversaries. Adv Neural Inf Process Syst. 2021;34:840-51.

45. Liu J, Wang X, Zhang Y, Chen H, Li Z. Fedpa: an adaptively partial model aggregation strategy in federated learning. Comput Netw. 2021;199:108468.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press

https://doi.org/10.1016/j.comnet.2021.108468.

46. Dhakal S, Prakash S, Yadati S, Gupta A, Ramakrishnan N. Coded federated learning. In: 2019 IEEE Globecom Workshops (GC Wkshps). IEEE; 2019. p. 1-6.

47. Gupta A, Ahmed T, Singh R, Sharma S. Fedcare: federated learning for resource constrained healthcare devices in iomt system. IEEE Trans Comput Soc Syst. 2023;10(4):1587-96.
https://doi.org/10.1109/tcss.2022.3232192.

48. Wu J, Wang X, Zhang Y, Chen H, Li Z. Topology aware federated learning in edge computing: a comprehensive survey. arXiv. 2023. arXiv:2302.02573.

49. Shi Y, Xu X, Wang H, Zhang L, Liu Y. Improving the model consistency of decentralized federated learning. arXiv. 2023. arXiv:2302.04083.

50. Cleland G, O'Sullivan D, Morrison J, Taylor P. Fedcomm: understanding communication protocols for edge based federated learning. arXiv. 2022. arXiv:2208.08764.
https://doi.org/10.1109/ucc56403.2022.00018.

51. Li T, Sahu AK, Talwalkar A, Smith V. Federated learning: challenges, methods, and future directions. IEEE Signal Process Mag. 2020;37(3):50-60.
https://doi.org/10.1109/msp.2020.2975749.

52. Zhao Z, Chen Y, Liu J, Zhang H, Li X. Towards efficient communications in federated learning: a contemporary survey. J Franklin Inst. 2023;360(12):8669-703.
https://doi.org/10.1016/j.jfranklin.2022.12.053.

53. Huang G, Zhao X, Lu Q. A new cross domain prediction model of air pollutant concentration based on secure federated learning and optimized lstm neural network. Environ Sci Pollut Res. 2023;30(2):5103-25.
https://doi.org/10.1007/s11356-022-22454-6.

54. Rizwan A, Ahmad N, Ali A, Khan S, Khan W. Intelligent digital twin for federated learning in aiot networks. Internet Things. 2023;22:100698.
https://doi.org/10.1016/j.iot.2023.100698.

© 2024 The Author(s). | Published by Institute of Smart Energy Systems Press