

AMI-FML: A Privacy-Preserving Federated Machine Learning Framework for AMI

Milan Biswal

Computer Science Department
New Mexico State University, USA
milanb@nmsu.edu

Abu-Saleh Md Tayeen

Computer Science Department
New Mexico State University, USA
tayeen@nmsu.edu

Satyajayant Misra

Computer Science Department
New Mexico State University, USA
misra@nmsu.edu

Abstract—Machine learning (ML) based smart meter data analytics is very promising for energy management and demand response applications in the advanced metering infrastructure (AMI). A key challenge in developing distributed ML applications for AMI is to preserve user privacy while allowing active end-users participation. This paper addresses this challenge and proposes a *privacy-preserving federated learning framework for ML applications* in the AMI. We consider each smart meter as a federated edge device hosting an ML application that exchanges information with a central aggregator or a data concentrator, periodically. Instead of transferring the raw data sensed by the smart meters, the ML model weights are transferred to the aggregator to preserve privacy. The aggregator processes these parameters to devise a robust ML model that can be substituted at each edge device. We also discuss strategies to enhance privacy and improve communication efficiency while sharing the ML model parameters, suited for relatively slow network connections in the AMI. We demonstrate the proposed framework on a use case federated ML (FML) application that improves short-term load forecasting (STLF). We use a long short-term memory (LSTM) recurrent neural network (RNN) model for STLF. In our architecture, we assume that there is an aggregator connected to a group of smart meters. The aggregator uses the learned model gradients received from the federated smart meters to generate an aggregate, robust RNN model which improves the forecasting accuracy for individual and aggregated STLF. Our results indicate that with FML, forecasting accuracy is increased while preserving the data privacy of the end-users.

Index Terms—Smart Meter Data Analytics, Federated Machine Learning, Short-term load forecasting, user privacy

I. INTRODUCTION

Smart meters are being rapidly deployed in household operations with an estimated 132 million smart meters to be operational in the United States alone, by the end of 2020 [1]. Smart meters, with bidirectional communication capability and back-end data management system, make up the advanced metering infrastructure (AMI). The AMI primarily facilitates the demand-side response (DSR) by enabling the active participation of residential consumers. In addition, AMI data is one of the important sources of real-time monitoring data in the smart distribution grid. This has paved the way for *AMI data analytics* and has galvanized a wide range of related applications, such as energy metering, load forecasting, load analysis, load management, electricity theft detection, and power systems monitoring [2]. AMI data has also been used to estimate distribution system topologies for enhancing the resiliency of the grid [3].

The recent advances in machine learning (ML) have taken data analytics to the next level, proving to be extremely efficient for modeling complex non-linear interactions. Particularly, the deep architectures of the neural networks in conjunction with enhanced computing capabilities have been successful in solving highly non-linear sequential time-series modeling tasks, such as speech recognition [4]. The AMI data being essentially sequential time-series has the potential to harness the advancement in ML to develop new applications for *AMI data analytics*.

A. User privacy issues in the AMI

The smart meter data can reveal certain personal attributes about the consumers, which might be unacceptable from a consumer privacy standpoint. For instance, the individual loads and their schedule of operations can be obtained by decomposing the smart meter data [5]. In [6], the authors show that a customer's lifestyle can be categorized by analyzing their household smart meter data. It can reveal sensitive personal information about a customer, such as availability at home, certain habits, social traits, economic status, etc. Therefore, user privacy is a significant concern that needs to be addressed in AMI data analytics applications.

Most of the privacy preservation techniques in the literature concentrate on the aggregation of the raw smart meter data on local aggregators in an attempt to dilute the identity of the individual customers [7]. The major limitation of this approach is that the aggregator can be compromised to get access to raw data. Moreover, aggregated data can still possess significant pointers, which can be exploited to estimate individual customer behavior. Another approach to preserve privacy is to provide confidentiality of data in transit, yet the distribution system operator (DSO) via the aggregators, receives the raw data sensed by the smart meter. Therefore, privacy is an issue if the DSO either wants to use the data for analytics applications or wants to share it with any third-party providers. In the worst case, even data confidentiality is an issue if the aggregator or the DSO are compromised as they contain the raw data in unencrypted form.

In this paper, we argue that the ML-based applications in the AMI can take the advantage of the black-box nature of ML models to inherently preserve privacy and overcome these limitations. The AMI data analytics applications are generally

deployed at the DSO and process the data reported by a group of smart meters. We suggest that the local ML applications can be deployed for each consumer either on a home energy management system (HEMS) device, as a software application on the smart meter, or on an associated embedded device. Considering the progress in Graphical Processing Units (GPUs) and neuromorphic computing, the deployment of ML applications on embedded devices is a reasonable assumption. The ML model weights or parameters are shared with the aggregators (or DSO) alleviating the need to share the raw user consumption data. We treat the smart meters as distributed clients hosting efficient local ML applications whose performances can be enhanced by amalgamating the model parameters of all the smart meters in a cluster composed of consumers in a small geographic area, without any violations in user privacy.

B. Contributions

The *key contributions* of this paper are as follows: **1)** A generalized privacy-preserving AMI federated ML (AMI-FML) framework for AMI data analytics—the framework is general and can be used as the structural basis for distributed/federated ML applications in the smart grid. **2)** Strategies to improve communication efficiency and additionally reinforce privacy by minimizing unintended memorization of ML models. **3)** A proof-of-concept implementation of the framework using a long short-term memory (LSTM) recurrent neural network for STLF. User privacy is better preserved with our AMI-FML framework as we share only the gradient information of the individual smart meter ML models instead of the raw data. Results show the effectiveness and improvement in accuracy due to the use of the federated learning approach.

The rest of this paper is organized as follows. Section II describes the proposed AMI-FML framework. Section III presents the proposed AMI-FML application use case for STLF along with brief background on STLF. The data description, evaluation criteria, and the framework evaluation with a summary of results and associated discussion are described in Section IV. Finally, Section V presents the concluding remarks.

II. AMI-FML FRAMEWORK

The smart meters in AMI are deployed at the distribution systems level, for each household or small industrial consumers. They measure the consumption of a customer at a specified interval (15 minutes, half-hourly, or hourly). We assume that the smart meters are connected to the communication network, capable of receiving pricing information or direct load control commands, and can exchange information with HEMS, which enables optimization of energy usage and demand response. The ML applications can either be installed in the smart meter module, HEMS, or a separate module attached with the smart meter.

One of the prominent applications of ML-based AMI data analytics will be to improve the efficiency of the smart distribution grid by keeping an eye on the changing energy markets.

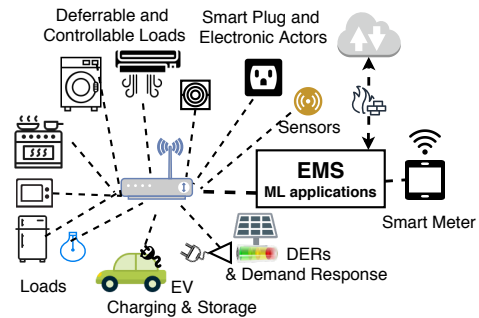


Fig. 1. Smart meters coupled with the Energy Management Systems

The smart meters anchored with HEMS, as illustrated in Fig. 1, can help provide better service to the prosumers. Third-party service providers can offer value-added energy services for the HEMS, leveraging smart meter data analytics [8]. They could provide commercial energy management solutions in near real-time that can manage loads and distributed generation to reduce energy bills. The third-party demand response (DR) services could rely on ML to achieve demand response in coordination with the DR program of the grid operator. ML-based anomaly detection schemes have been very useful in many systems. They can be deployed as smart meter applications to identify anomalous consumption patterns, detect electricity theft, identify power quality disturbances, and monitor equipment health. The smart meters in a particular geographic region form a cluster and are connected to an aggregator or data center maintained by the DSO. In the case of a third-party ML application, the aggregator can also be hosted on a server connected to the communication network. Essentially, the federated learning algorithm runs at the aggregator and creates a global ML model by the fusion of local ML models in a cluster. After achieving sufficient generalization of the ML models to minimize errors, the trained ML models are deployed on the smart meters at the edge. The fusion of the ML model parameters is carried out at a fixed interval determined by the application requirements. We present a brief overview of the smart meter communication infrastructure in the following section, followed by the details of the FML.

A. Smart Meter Communication Infrastructure

The smart meter communication infrastructure illustrated in Fig. 2, is mainly built over three types of architectures [9]. First, the smart meters are directly connected to the data concentrator or aggregator via the mobile network (GPRS, CDMA, LTE, 5G), and the responsibilities for data communications are handled by the mobile operator. Second, the smart meters are connected to the data concentrator through Power Line Communications (PLC) or Broadband over Power Lines (BPL) technologies. Third, there can be a gateway to which the smart meters are connected and the gateway can dynamically choose the communication media among cellular, WiFi hot-spots, and PLC/BPL network. The gateway acts as an interface between the data concentrators and the smart meters. The data concentrators are connected to the substation via BPL, WiMAX, or Ethernet.

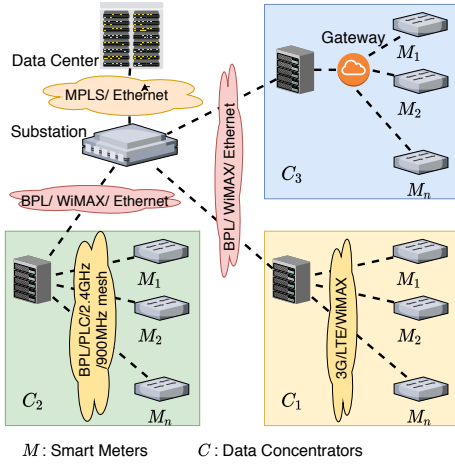


Fig. 2. Smart meter communication infrastructure.

The ML applications at the consumer side can use any of the communication infrastructures to exchange information with the central aggregators. In the near future, the data communications in the smart meter may happen over the *public* or non-dedicated communication networks, such as the internet. This will further allow the third-party service providers to directly interact with the consumers, through the internet.

B. Federated machine learning

The canonical federated ML (FML) deals with building a global statistical model from the data gathered by a group of remote local devices (smart meters) deployed at the edge. The global ML model is created at the aggregator (or DSO) by the fusion of all the local ML model parameters corresponding to each smart meter. The aggregator and the smart meters exchange model parameters through intermediate updates communicated periodically.

The local ML model weights ($W_n^t \in \mathbb{R}$) are updated through gradient descent. The number of epochs for learning is fixed so that sufficient generalization is achieved with the inclusion of new training data. After a fixed time step the learned weight for all the smart meters is sent to the central server or the aggregator, which hosts the global model. Instead of sending the actual weight values to the aggregator, we send the difference in the weight values as the weights update (also referred to as model update). The model update sent by the n^{th} smart meter to the aggregator is expressed as:

$$\Delta W_n^t = W_n^t - \bar{W}^{(t-1)} \quad (1)$$

Here, \bar{W}^{t-1} represents the global ML model weights at the time $(t-1)$.

The aggregator or the centralized server receives the weight updates ΔW_n^t from all the individual smart meters in a set $n \in \mathcal{N}$, at any instant of time t . These weights are then combined according to the federated averaging algorithm [10] to create a global model that encapsulates information abstracted by the local models.

For the experiments in this paper, we use the *FedAvg* [10] algorithm to aggregate the local models, where the parameters are averaged element-wise with weights proportional to the sizes of local datasets. However, the algorithm for model fusion can be chosen based on the application at hand. Typically, the goal of this federated averaging algorithm is to minimize the following objective function:

$$\min_{W^t} \sum_{n \in \mathcal{N}} p_n F(W_n^t) \quad (2)$$

Here, $F(W_n^t)$ is the local objective function or the loss function of the local ML model, for n^{th} smart meter at time t . p_n specifies the relative contribution of each local model corresponding to a smart meter. For experiments in this paper, as the training data sizes for each local model are assumed to be the same, the value of p_n is set as $\frac{1}{N}$. Here, N is the total number of smart meters connected to an aggregator.

We perform the federated averaging according to Eqn. (3), which is essentially the element-wise mean of the model parameters of all the local models.

$$\Delta W^t = \sum_{n \in \mathcal{N}} p_n \Delta W_n^t \quad (3)$$

The global model parameters are updated as:

$$\bar{W}^{(t+1)} = \bar{W}^t + \Delta W^t \quad (4)$$

C. Communication efficiency

The number of parameters required for the acceptable generalization of an ML model depends on the complexity of the data and the nature of the application. A well generalized deep neural network may have thousands of parameters that may burden the communication channel. The transfer of difference in parameter values results in sharing of a small set of the parameters that have been updated. In addition to this, there are two umbrella approaches, structured update and sketched update for reducing communication overhead and additionally strengthen privacy [11]. We discuss these approaches here.

1) *Structured Update*: To reduce the cost of sending ΔW_n^t to the aggregator, we tested a *structured update* approach. In this approach, the update is restricted to have a predefined structure. We created a sparse matrix $Y \in \{0, 1\}$ of the same dimension as ΔW_n^t , called the *random mask*. We created separate random masks for each smart meter independently in every round of federated learning. The sparsity of the random mask is proportional to the number of zeros in the corresponding matrix. In our experiments, we randomly selected a fraction of the elements in the sparse matrix to be zero. The gradient parameter matrix ΔW_n^t is then multiplied element-wise with the random mask, to result in a sparse version of ΔW_n^t , referred to as $\Delta \bar{W}_n^t = \Delta W_n^t \circ Y$. Here, \circ represents element-wise multiplication operation. Each smart meter, now sends $\Delta \bar{W}_n^t$ to the server instead of ΔW_n^t .

2) *Sketched Update*: This is an encoding technique to compress the size of the model parameter vectors. Sub-sampling is one of the easiest ways to achieve this, however, it comes with a cost similar to random masking. A more efficient approach is

to use probabilistic quantization, where the parameters/weights are distributed into encoded buckets. All the data in a bucket consolidate their values to that of the value associated with the bucket, with an associated probability.

A one-bit adaptive quantization of a matrix S is expressed as:

$$S(i, j) = \begin{cases} S_{max}, & \text{with probability } \frac{S(i, j) - S_{min}}{S_{max} - S_{min}} \\ S_{min}, & \text{with probability } \frac{S_{max} - S(i, j)}{S_{max} - S_{min}} \end{cases} \quad (5)$$

where $S_{max} = \max(S)$ and $S_{min} = \min(S)$

D. Privacy in the FML framework

The ML model gradient parameters (ΔW_n^t) provide a sparse and encoded representation of the raw smart meter data and thwart the possibility of deriving the input time-series (raw smart meter data) from just model parameters. This inherent feature of FML introduces definitive user privacy [12]—the gradient parameters cannot be inverted to generate the raw data. However, the unintentional memorization of ML models could be a concern [13]. A small amount of unintended memorization in ML models may occur when the models are either over-trained or not regularized. This means that the gradient parameters could be used to infer either some salient features of the raw data or a partial reconstruction of the input data may be possible, which can undermine privacy.

In the case of the smart meters, the hidden states of the ML model may be exploited to probe for unique consumption patterns. This is alleviated by either randomly dropping some gradient parameters or approximating their values. We perform these inherently to enhance the communication efficiency (for both *random mask* and *adaptive quantization*). This increases the robustness of the AMI-FML against intended or unintended memorization and the privacy is additionally reinforced. The global ML model essentially consumes the latent features or information that are *exclusively* relevant for the application at hand, instead of the raw smart meter data. Therefore, this framework can be used as a foundation to develop a range of other FML applications without compromising user privacy utilizing the proposed AMI-FML framework.

III. FML APPLICATION FOR STLF

In this section, we discuss the proposed AMI-FML framework in the context of a use case: ML application for improving STLF while preserving privacy. First, we present a brief overview of the STLF in III-A. In Section III-B, we briefly present the concepts of LSTM-RNN for STLF.

A. Short-Term Load Forecasting (STLF)

Both short-term and long-term load forecasting are critical for power distribution utilities. The STLF for individual consumers plays a more important role in future grid planning and operation [14]. This enables short-term energy price forecasts which are of particular interest to power portfolio managers. The term *short-term* covers from a few minutes up to a few days ahead. The accuracy of the STLF impacts the efficiency

of the DR mechanisms relying on direct load control (DLC) or locational marginal pricing (LMP). LMP is a driver and determines distribution congestion pricing (DCP) in the day-ahead energy market [15]. The patterns of the time-series corresponding to the load consumption at individual household levels and feeder levels are generally distinctive, with the former being a lot more volatile than the latter.

The intrinsic uncertainties associated with STLF come from external factors, such as weather conditions, variable generation from household DERs, and unexpected changes in the demand. For instance, the outside temperature has a profound effect on the heating and cooling system loads. The amount of sunshine and wind speed affects the amount of distributed generation pumped into the grid. The charging pattern of the electric vehicles and the battery storage further introduce volatility in load consumption.

The load consumption patterns are a result of the complex nonlinear interactions among several factors including the ones stated above. It is essential to model these interactions with sufficient faithfulness to predict future consumption, which is extremely challenging. This complexity has increased with the rapid rise in the number of grid-connected DERs. Several solutions to address this have been proposed, incorporating advances in signal processing and time series forecasting. Autoregressive Integrated Moving Average (ARIMA) type models have been quite successful in modeling non-linear time-varying processes such as STLF [16]. However, most of these statistical models cannot encapsulate the long-term dependencies and have a natural tendency to settle towards the mean values of the past series data.

B. LSTM-RNN for load forecasting

In our problem, the local devices are the smart meters hosting applications with local ML models comprising of the LSTM-RNNs. The goal of our ML application is to accurately forecast future energy consumption. Essentially, this is a sequential time series modeling challenge, and RNNs are the ideal candidate for this task. The recurrent connections among neurons in an RNN preserve the hidden state transitions associated with the temporal variations. However, the RNNs are difficult to train and suffer from the problem of either exploding gradient (where the weight updates become excessively large numbers) or vanishing gradients (where the weight updates become insignificantly small numbers), hindering the learning process. Another key limitation of the basic RNNs is their inability to preserve the learned information for longer periods with the increase in time lag.

The LSTM is a variant of RNN, that alleviates these limitations, paving the way for efficiently modeling sequential data. It has demonstrated quite remarkable short-term load forecasting performance for individual customers [14]. The key elements in an LSTM cell are the memory cell and the gate units (forget, input, output) [17] as depicted in Fig. 3. The *memory cell* is responsible for preserving information regarding the temporal state of the neural network and consequently the temporal patterns of the time-series. The gates consist of

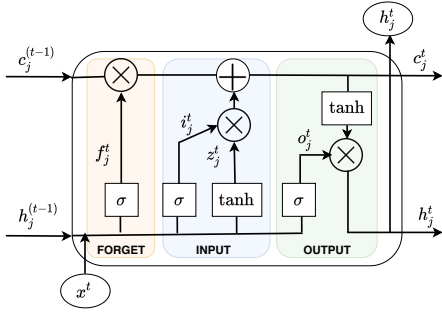


Fig. 3. Inside an LSTM cell

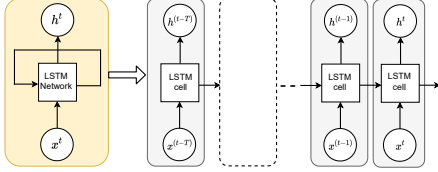


Fig. 4. Unrolled LSTM network architecture for short-term load forecasting.

multiplicative units that control the flow of information, and decide what and how much information should be preserved.

Let x^t represents the input to j^{th} LSTM cell and h_j^t represents the internal state of the LSTM cell, at a time index t . The *forget gate* in the LSTM has a sigmoidal activation function (σ), and is responsible to determine what information is irrelevant and should be erased from the memory parameter $c_j^{(t-1)}$. The *input gate* adds new information i_j^t to the estimated memory state z_j^t . Terms i_j^t and z_j^t are derived from the input x^t by processing it through a sigmoidal and a *tanh* activation function, respectively. The *input gate* also adds the sharing parameter vector h_j^{t-1} to c_j^t . The *output gate* determines the new hidden state h_j^t from the memory state parameter c_j^t , previous hidden state h_j^{t-1} , and the input x^t by utilizing *sigmoid* and *tanh* functions.

The set of equations governing the forward pass of the LSTM network can be expressed as [17]:

$$f^t = \sigma(W_f x^t + R_f h^{(t-1)} + P_f \odot c^{(t-1)} + b_f) \quad (6)$$

$$i^t = \sigma(W_i x^t + R_i h^{(t-1)} + P_i \odot c^{(t-1)} + b_i) \quad (7)$$

$$z^t = \tanh(W_z x^t + R_z h^{(t-1)} + b_z) \quad (8)$$

$$c^t = z^t \odot i^t + c^{(t-1)} \odot f^t \quad (9)$$

$$o^t = \sigma(W_o x^t + R_o h^{(t-1)} + P_o \odot c^t + b_o) \quad (10)$$

$$h^t = \tanh c^t \odot o^t \quad (11)$$

Here, $f^t = [f_1^t f_2^t \dots f_H^t]$, $i^t = [i_1^t i_2^t \dots i_H^t]$, $z^t = [z_1^t z_2^t \dots z_H^t]$, $c^t = [c_1^t c_2^t \dots c_H^t]$, and $o^t = [o_1^t o_2^t \dots o_H^t]$ are vectors comprising of the *sigmoid* activation outputs of the forget gate, the input gate, *tanh* activation output of the input gate, cell state, and the *sigmoid* activation output of the output gate respectively.

$W_i^t, W_z^t \in \mathbb{R}^{H \times D}$ are matrices consisting of *input gate* weights where $D = 1$ is the number of features (one value for each electricity load data point measured by the smart meter) and H is the number of hidden units in the LSTM cell. $W_f^t, W_o^t \in \mathbb{R}^{H \times D}$ are the forget and output gate weight matrices, respectively. $R_z^t, R_i^t, R_f^t, R_o^t \in \mathbb{R}^{H \times H}$ are the

recurrent weights. Similarly, $P_f^t, P_i^t, P_o^t \in \mathbb{R}^H$ are peephole weights and $b_f^t, b_i^t, b_z^t, b_o^t \in \mathbb{R}^H$ are the bias parameters. The cumulative parameter set of an LSTM network corresponding to the n^{th} smart meter at any time index t consists of all the weights and bias values for that smart meter, which is represented as a multidimensional data frame W_n^t [17].

$$W_n^t = \begin{Bmatrix} W_f^t & R_f^t & P_f^t & b_f^t \\ W_i^t & R_i^t & P_i^t & b_i^t \\ W_z^t & R_z^t & - & b_z^t \\ W_o^t & R_o^t & P_o^t & b_o^t \end{Bmatrix} \quad (12)$$

In Fig. 4, we show a general LSTM network architecture and its unrolled representation. The smart meter data $x(t)$ for a given smart-meter is sequentially fed into the LSTM network in the form of an input vector $x(t) = [x_D^{(t-T)} \dots x_D^{(t-1)} x_D^{(t)}]$ corresponding to a time instant t . Here, $T = 48$ is the number of previous time steps (electricity load data points) we want to use to predict the electricity load of the next time step. Since the LSTM cells contain *sigmoid* and *tanh* activation functions which are sensitive to the scale of the data, we normalized our data in the range $[0 - 1]$. After feeding input to the LSTM network, it creates an output of shape $[T, H]$. As we used 50 hidden units for each of our LSTM cells, we have $H = 50$. This implies, in the output for each time step there will be 50 hidden features. In the end, we took 50 hidden features from the last time step of the LSTM output and passed it through a linear layer to get the final output which is the predicted energy consumption at the time $(t + 1)$.

IV. EVALUATION AND RESULTS

We evaluated the proposed AMI-FNL framework on the ML based STLF use-case (discussed in Section III). The dataset for the experiments and the performance metrics are described in Sections IV-A and IV-B. We describe the experimental setups and associated results in Sections IV-C-IV-E.

A. AMI Data Description

We used the dataset released by Smart Metering Electricity Customer Behaviour Trials (CBTs) initiated by Commission for Energy Regulation (CER) in Ireland, for the experiments in this paper. From this dataset, we selected a subset of 3600 smart meters which were deployed for residential consumers. The selected dataset consists of electricity load consumed by the consumers from July 14th, 2009 to December 31st, 2010 (533 days), each comprising of 25,584 data points, with a sampling interval of 30 minutes. We did pre-processing of the raw data to ensure that the bad data (smart meter readings for the days with missing measurements) were removed. In our experiments, we have used the first 503 days of data (24,144 data points) for training and the last 30 days of data (1440 data points) for testing.

We divided the selected dataset into 10 clusters with 50 smart meters in each cluster. In real-world applications, number of smart meters in a geographic area connected to an aggregator could form a cluster. In case of third party applications, contingent on the internet, clusters can be carefully formed

based on geographical or electrical proximity. The number of smart meters in a cluster can be decided based on the efficiency of the communication infrastructure. For the STLF, as the reporting intervals of the smart meters are 30 minutes, we consider a half-hourly forecasting horizon.

B. Performance Metrics

We considered *two* well-established performance metrics [18] to assess the forecasting performance of the proposed method. Let, $\hat{y}(m)$ represents a point forecast (predicted consumption) at a time index m , $y(m)$ is the actual consumption, and the total number of point forecast in an experiment is M , for an individual smart meter.

The performance metrics are defined as:

(a) Normalized Root Mean Square Error (NRMSE)

$$NRMSE = \frac{\sqrt{\frac{\sum_{m=1}^M (y(m) - \hat{y}(m))^2}{M}}}{\max(y) - \min(y)} \quad (13)$$

Here, $\max(y)$ and $\min(y)$ represent the maximum and minimum value of actual consumption y .

(b) Mean Absolute error (MAE)

$$MAE = \frac{1}{M} \sum_{m=1}^M |y(m) - \hat{y}(m)| \quad (14)$$

C. STLF with local ML model

We did not perform any federated averaging in this experiment. The LSTM models were trained on the historical data (of 503 days) corresponding to that specific consumer. We limited the number of training epochs of the LSTM-RNNs to 4 in order to achieve acceptable generalization without over-fitting. After that, the trained models were deployed for forecasting. In this case, the LSTM models entirely rely on the available local data for training and learning. We computed the performance metric values for individual smart meters and averaged them across all the smart meters in a cluster. The overall (average) NRMSE, and MAE across the 10 clusters were: 0.11 and 0.41 respectively. The small value of NRMSE and MAE indicates the efficacy of LSTM networks for STLF. The superior forecasting performance of the LSTM can be attributed to its property that preserved very long-term information that relate the internal state transitions and is essential for accurate forecasting.

D. STLF with FML model

In this case, we trained the LSTM models in a federated setting by sharing the model weights for each epoch (after traversing through all the time steps in an epoch). In the federated ML environment, the model updates are normally shared at specific intervals of time. The true advantage of federated ML to improve forecasting accuracy can be harnessed only when we continuously improve the performance of the global model and substitute it in place of the local models at specific intervals. This enables the trained forecasting model to be further trained to accommodate new patterns of consumption after getting deployed at the edge device.

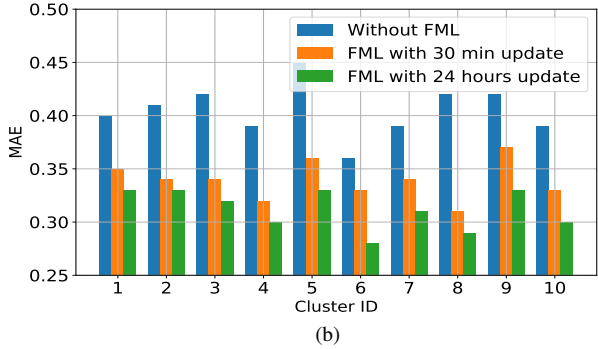
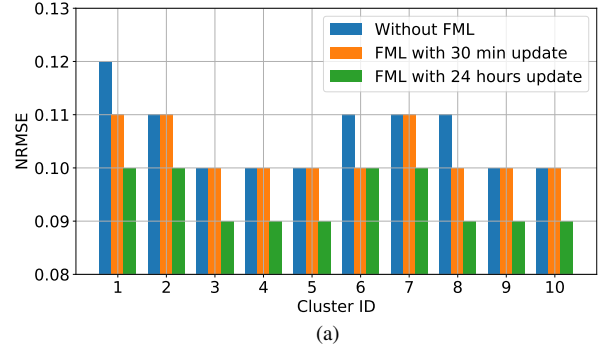


Fig. 5. Bar chart showing average (a) NRMSE and (b) MAE for each cluster.

We conducted two sets of experiments related to this. In the first, the local models were shared every 30 minutes, which was the highest level of granularity that was achievable with our dataset. The average NRMSE and MAE averaged across all the clusters were found to be 0.10, and 0.33. All these error metrics are smaller compared to the case without federated ML. For all the clusters the average errors decreased or the forecasting performance improved with federated ML. The best-case performance improvements (decrease in error) were observed to be 0.11 for MAE. Although small, these improvements will enhance STLF and impact the transactive energy market.

Then, we conducted another set of experiments where we decreased the granularity to 24 hours or daily. The motivation behind this is that the consumption patterns are more prominent when we observe the consumption for several days. In addition, many utilities give increased emphasis on the day ahead price forecasting to commit energy transactions. The average NRMSE and MAE with 24 hourly federated model updates were observed to be 0.094 and 0.312 respectively. These errors are smaller compared to the case without federated ML, and even smaller compared to the federated ML case with half-hourly updates.

The superior performance of the daily model update compared to half-hourly updates can be explained by the functional working principle of the LSTM network. The LSTM learns unique features from the transition patterns embedded in the data, which requires the exploration of sequential data of sufficient length/duration. The inclusion of one point to the

TABLE I
MEAN FORECASTING ERRORS WITH STRUCTURED AND SKETCHED
UPDATE OF MODEL PARAMETERS EVERY 24 HOURS.

| | | NRMSE | MAE |
|--------------|-------|-------------|-------------|
| Random Mask | 10% | 0.12 | 0.42 |
| | 5% | 0.11 | 0.38 |
| | 2% | 0.10 | 0.34 |
| Quantization | 1 bit | 0.12 | 0.38 |
| | 2 bit | 0.11 | 0.37 |
| | 4 bit | 0.10 | 0.32 |

training sequence may not create adequate information to be abstracted in the LSTM. Whereas, the 24 hours update may offer a range of motifs for the consumption patterns. This is because certain loads such as the dishwasher follow a daily routine, and are operational for a specific interval of time.

E. Communication efficiency

In this experiment, we evaluated the techniques for improving communication efficiency. The forecasting performance of *random mask* based structured update and *adaptive quantization* based sketched updates are summarized in Table I. The performance reductions for random mask with 10%, 5%, and even 2% zeros are significant. However, for sketched update with 4-bit adaptive quantization, the performance is close to the original case with 24-hour federated ML updates. The quantization, in addition to communication overhead reduction (4-bit quantization results in $8\times$ compression compared to a 4 byte float) also introduces ancillary privacy features [11].

V. CONCLUSION

We presented a generalized federated learning based paradigm for developing privacy-preserving ML applications (AMI-FML) in the smart grid. We demonstrated techniques to enhance communication efficiency and reinforce privacy in FML. We presented the proof-of-concept through a FML based STLF application supported by the AMI. We used LSTM networks to accurately forecast energy consumption based on the historical smart meter readings. It was demonstrated that the FML improves the forecasting accuracy of the individual smart meters compared to local ML models (without FML), and aids in more efficient HEMS while preserving user privacy. In addition to the facilitation of intelligent services by the grid operators, the proposed framework will specifically encourage third-party service providers to roll out value-added services for the benefit of consumers.

The future direction could be to bring into play the advances in federated ML concepts to improve the performance of smart meter based ML applications in the smart grid. It will be worthwhile to evaluate advanced federated averaging algorithms such as Agnostic Federated Learning, Probabilistic Federated Neural Matching, and matched averaging. Enhancing communication efficiency while embedding differential privacy without reduction in performance is another aspect of potential future work.

REFERENCES

- [1] "Advanced metering infrastructure and customer systems-results from the smart grid investment grant program," U.S. Department of Energy, Tech. Rep., 2016. [Online]. Available: https://www.energy.gov/sites/prod/files/2016/12/f34/AMI%20Summary%20Report_09-26-16.pdf
- [2] Y. Wang, Q. Chen, T. Hong, and C. Kang, "Review of smart meter data analytics: Applications, methodologies, and challenges," *IEEE Transactions on Smart Grid*, vol. 10, no. 3, pp. 3125–3148, May 2019.
- [3] J. Peppanen, M. J. Reno, M. Thakkar, S. Grijalva, and R. G. Harley, "Leveraging ami data for distribution system model calibration and situational awareness," *IEEE Transactions on Smart Grid*, vol. 6, no. 4, pp. 2050–2059, 2015.
- [4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, <http://www.deeplearningbook.org>.
- [5] N. Jin, P. Flach, T. Wilcox, R. Sellman, J. Thumim, and A. Knobbe, "Subgroup discovery in smart electricity meter data," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1327–1336, May 2014.
- [6] J. Kwac, J. Flora, and R. Rajagopal, "Lifestyle segmentation based on energy consumption data," *IEEE Transactions on Smart Grid*, vol. 9, no. 4, pp. 2409–2418, July 2018.
- [7] M. R. Asghar, G. Dán, D. Miorandi, and I. Chlamtac, "Smart meter data privacy: A survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2820–2835, Fourthquarter 2017.
- [8] C. E. Kontokosta, "Energy disclosure, market behavior, and the building data ecosystem," *Annals of the New York Academy of Sciences*, vol. 1295, no. 1, pp. 34–43, 2013. [Online]. Available: <https://nyaspubs.onlinelibrary.wiley.com/doi/abs/10.1111/nyas.12163>
- [9] S. Chren, B. Rossi, and T. Pitner, "Smart grids deployments within eu projects: The role of smart meters," in *2016 Smart Cities Symposium Prague (SCSP)*, May 2016, pp. 1–5.
- [10] B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," *Google Research Blog*, vol. 3, 2017.
- [11] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.
- [12] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Privacy aware learning," *Journal of the ACM (JACM)*, vol. 61, no. 6, pp. 1–57, 2014.
- [13] N. Carlini, C. Liu, J. Kos, Ú. Erlingsson, and D. Song, "The secret sharer: Measuring unintended neural network memorization & extracting secrets," *arXiv preprint arXiv:1802.08232*, 2018.
- [14] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on lstm recurrent neural network," *IEEE Transactions on Smart Grid*, vol. 10, no. 1, pp. 841–851, Jan 2019.
- [15] W. Liu, Q. Wu, F. Wen, and J. Østergaard, "Day-ahead congestion management in distribution systems through household demand response and distribution congestion prices," *IEEE Transactions on Smart Grid*, vol. 5, no. 6, pp. 2739–2747, Nov 2014.
- [16] K. Maciejowska and R. Weron, "Short- and mid-term forecasting of baseload electricity prices in the u.k.: The impact of intra-day price relationships and market fundamentals," *IEEE Transactions on Power Systems*, vol. 31, no. 2, pp. 994–1005, 2016.
- [17] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, Oct 2017.
- [18] G. Zhang and J. Guo, "A novel method for hourly electricity demand forecasting," *IEEE Transactions on Power Systems*, vol. 35, no. 2, pp. 1351–1363, 2020.