Federated Learning based Energy Demand Prediction with Clustered Aggregation

Ye Lin Tun, Kyi Thar, Chu Myaet Thwal, Choong Seon Hong
Department of Computer Science and Engineering
Kyung Hee University
Yongin-si, 17104, Republic of Korea
{yelintun, kyithar, chumyaet, cshong}@khu.ac.kr

***Abstract*—To reduce negative environmental impacts, power stations and energy grids need to optimize the resources required for power production. Thus, predicting the energy consumption of clients is becoming an important part of every energy management system. Energy usage information collected by the clients' smart homes can be used to train a deep neural network to predict the future energy demand. Collecting data from a large number of distributed clients for centralized model training is expensive in terms of communication resources. To take advantage of distributed data in edge systems, centralized training can be replaced by federated learning where each client only needs to upload model updates produced by training on its local data. These model updates are aggregated into a single global model by the server. But since different clients can have different attributes, model updates can have diverse weights and as a result, it can take a long time for the aggregated global model to converge. To speed up the convergence process, we can apply clustering to group clients based on their properties and aggregate model updates from the same cluster together to produce a cluster specific global model. In this paper, we propose a recurrent neural network based energy demand predictor, trained with federated learning on clustered clients to take advantage of distributed data and speed up the convergence process.**

***Index Terms*—energy, federated learning, recurrent neural network, clustering, long short-term memory**

I. Introduction

Electrical energy is an essential component for a single household to all industrial and manufacturing systems [1]. To effectively utilize the available natural resources, power plants need to efficiently produce and distribute energy based on clients' demand. Nowadays, smart home systems can monitor and log energy usage statistics from various sources across the house. This highly granular information can be valuable for energy companies to effectively manage the power production and distribution operations.

During recent years, increasing computing capabilities and data explosion has led to a significant performance increase for data-driven predictor models [2]. Many existing studies on energy consumption modeling [3]–[7] utilize data-driven methods, including deep neural networks (DNN), decision trees, support vector machines (SVM), and other machine learning methods. In these previous methods, data is centralized for training the data-driven model.

Predicting the future energy demand of a client based on a series of past energy usage data is a time series regression task. Recurrent neural networks (RNNs) are a type of neural network that can accept a series of values as the input to predict the output [8] and they are suitable for tackling time series predictive problems. In our system, the RNN model is chosen for predicting the future energy demand of a client given past energy usage data.

In a traditional neural network training pipeline, clients' data are collected onto a central server where the model is trained. Transmitting data from the clients onto a server is expensive in terms of communication cost, and adapting the model to new usage patterns in the future would require constant re-transmission of newly obtained data. However, most of the existing works [3]–[7] only consider the centralized training approach where data transmission is essential.

To overcome these challenges, we can apply federated learning [9], [10] where clients can collaboratively train a deep neural network on their own local data without needing to centralize. However, training a neural network with federated learning usually suffers from the non-IID data problem [11], where clients contain data distributions that are diverse from each other. As a result, the global model created by aggregating different client model updates has a poor convergence rate and performance. Clustering clients with similar properties is a promising way to alleviated this problem. Model updates produced by the clients of the same cluster can be aggregated together to create a specific global model for that cluster. In this work, we integrated the clustered aggregation in the federated training process of our RNN model, by grouping different model updates based on available client attributes.

II. System Model

Our proposed energy demand predictor system is shown in Fig. 1. We modeled the clients as smart homes capable of monitoring the energy usage across the household. The central server can be an electrical power company. The smart home system in each client is capable of training a neural network on its local data, but each individual client may have low amount

This work was partially supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-01287, Evolvable Deep Learning Model Generation Platform for Edge Computing) and by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. No. 2020R1A4A1018607). *Dr. CS Hong is the corresponding author.

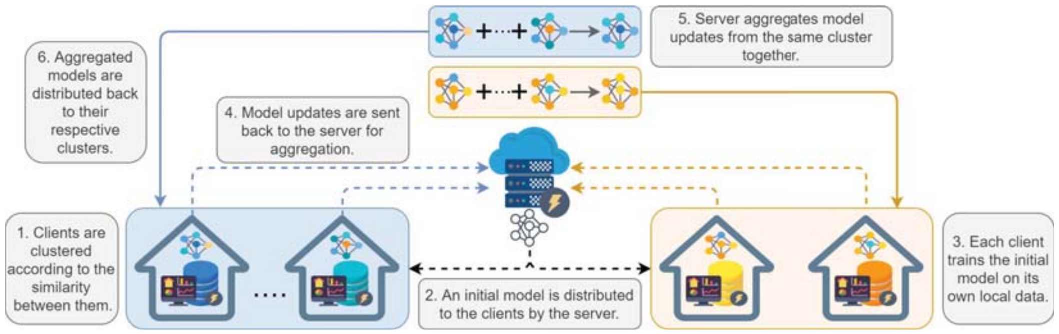

6. Aggregated models are distributed back to their respective clusters.
4. Model updates are sent back to the server for aggregation.
5. Server aggregates model updates from the same cluster together.
1. Clients are clustered according to the similarity between them.
....
2. An initial model is distributed to the clients by the server.
3. Each client trains the initial model on its own local data.

Fig. 1: Federated Learning and Clustering based Model Training Process

of training data. Due to communication resource constraints, client households may also be unable to centralize their data or share among each other. To tackle these challenges, we trained our RNN model with the federated learning paradigm which allows clients to collaboratively train the model on their own local data without needing to share their data with a central entity.

At the start of each federated training round, the central server or the power company clusters the participating client households based on the similarity of their housing attributes. After that, the server distributes a base global RNN model to all client households in each cluster. Each client individually trains the base RNN model on its own data. After training the model, each client sends its model updates back to the server. The central server aggregates the model updates from the clients of the same cluster together to create a global model for each cluster. Then, the cluster specific global models are distributed back to the clients in the respective clusters. This process is independently repeated for each cluster until the respective global model converges.

A client can utilize the trained global model in its smart home system to predict the upcoming energy usage and effectively manage the energy consumption. When the electrical power company requires clients' future energy demand, it can simply request the respective clients to make highly granular predictions on their local usage data without any need for data transmission.

III. Federated Model Training and Client Clustering

In a typical federated learning workflow [9], [10], the server distributes a base global model θ_t to the participating clients at the start of each training round t. Each client i trains the model on its local data D_i using stochastic gradient descent to generate model updates.

$$\Delta\theta_i^{t+1} = \text{SGD}(\theta^t, D_i) - \theta^t,\ i = 1, ..., m \tag{1}$$

Each client sends its model updates back to the server and the server aggregates the model updates by weighted averaging to produce the next global model.

$$\theta^{t+1} = \theta^t + \sum_{i=1}^{m} \frac{|D_i|}{|D|} \Delta\theta_i^{t+1} \tag{2}$$

Then, the global model is distributed back to the clients. Although the central server does not directly take part in the training process, it is responsible for monitoring and orchestrating the federated training and model aggregation process.

To increase the global model convergence rate, we used OPTICS (Ordering points to identify the clustering structure) [12] algorithm to cluster the clients with similar attributes. For each cluster, a global model is produced by aggregating the model updates from the cluster members. OPTICS is an unsupervised clustering algorithm with ϵ and $MinPts$ parameters. ϵ is the distance (radius) around a point to consider for clustering. $MinPts$ defines the minimum number of required points to form a cluster. If there are at least $MinPts$ points in ϵ radius neighborhood N_ϵ of a point p, then p is a core point.

Algorithm 1 shows the details of out proposed method.

Algorithm 1 Federated training with clustered aggregation

```
1:  Input: Attribute data from each of $m$ clients.
2:  Based on attributes, server groups clients into $n$ clusters.
3:  Server randomly initializes a base model $w_{rand}$.
4:  for each cluster $C^k$ with $k = 1, 2, ..., N$, in parallel do
5:      Initialize cluster specific model weights, $w_k \leftarrow w_{rand}$
6:  end for
7:  for each cluster $C^k$ with $k = 1, 2, ..., N$, in parallel do
8:      for communication round $t = 1, 2, ...., T$ do
9:          for each client $c^i$ in cluster $C^k$, $i = 1, 2, ..., L$ in
            parallel do
10:             Synchronize local model with latest cluster specific
                model: $w_k^i \leftarrow w_k$
11:             Update local model $w_k^i$ by training on local data,
                $D^i = \{X^i, Y^i\}$
12:             Transmit updated $w_k^i$ back to the server.
13:         end for
14:         Update model weights for cluster specific model by
            aggregation: $w_k \leftarrow \frac{1}{l}\sum_{i=1}^{l} w_k^i$
15:     end for
16: end for
17: Output: cluster specific models with trained weights: $w_k$,
    $k = 1, 2, ..., N$
```

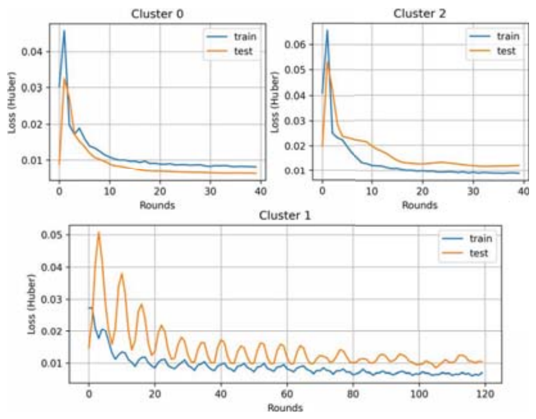


Fig. 2: Training History for Cluster 0, 1 and 2 Models

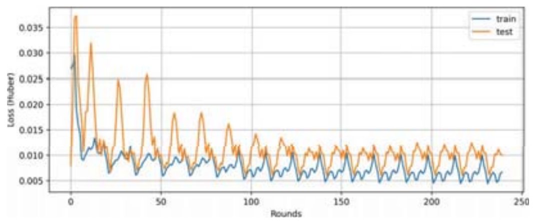
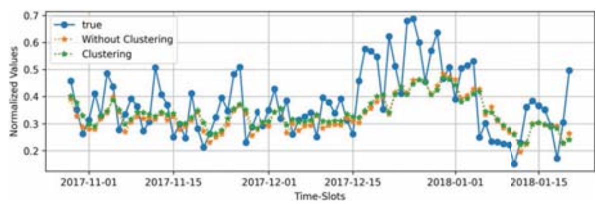
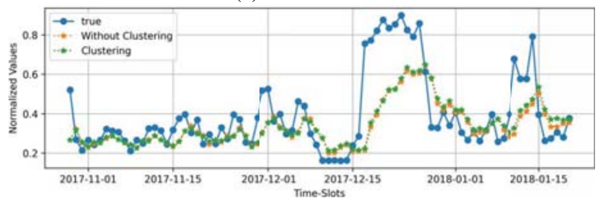


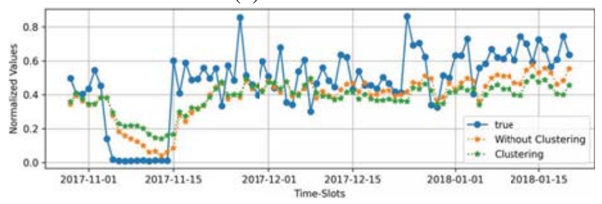
Fig. 3: Training History for Model Trained Without Clustering



(a) Cluster 0



(b) Cluster 1



(c) Cluster 2

Fig. 4: Predictions from Cluster Specific Models and Model Trained Without Clustering

IV. Experimental Settings

For our experiments, we used "HUE: The Hourly Usage of Energy Dataset for Buildings in British Columbia" [13] dataset. It contains almost three years worth of hourly energy usage data for twenty-two households as well as the housing attributes. We limited the time period for training data between '2015-09-29' and '2017-09-29' and based on the availability, houses with IDs 3-15, 19, and 20 are chosen. Available energy readings after '2017-09-29' for each house are kept as hold out test data. Hourly energy usage readings are re-sampled into daily energy usage data.

Initially, we input housing attributes into the OPTICS algorithm and to get the clustered house IDs as output. Types of attributes used for clustering include: house type (eg.bungalow or apartment), facing direction, region, rental units, heating type (electricity or gas). In our experiments, we used scikit-learn [14] implementation of OPTICS algorithm and we set the minimum number of cluster samples as two and other parameters are left as default. On our clients, clustering produced three clusters and a set of clients are sampled as noise. Cluster 1 includes house IDs 3, 5, and 14. Cluster 2 includes house IDs 4, 8, 9, 10, 13 and 19. Lastly, cluster 3 includes house IDs of 6 and 12. For each of the three clusters, we train an RNN model in a federated learning manner.

We use Keras API [15] with TensorFlow [16] backend for implementing the RNN model. Bidirectional LSTM (long short-term memory) [17] layers are used for constructing the RNN model. Since LSTM layers can recognize patterns of long-term dependencies, it is suitable for predicting time series data. In a bidirectional LSTM layer [18], neurons are split into two parts in which one part is responsible for the forward states and the other for the backward states. A bidirectional LSTM layer can utilize both past and future information of an input time series for training. Our RNN model is a sequential model that contains two bidirectional LSTM layers, two dense layers and an output dense layer. Our RNN model is configured to receive thirty days' worth of energy usage data and predict the next day demand.

RNN models for each cluster are trained separately in federated learning paradigm. At the start of every training round, the latest global model is synchronized to the participating clients as the base model. In every training round each client trains the base model for five epochs. At the end of every training round, the resulting global model is evaluated on the test data of its respective cluster.

V. Evaluation and Results

Federated training history plots for each cluster are shown in Fig. 2. For comparison, we also trained an RNN model in the federated learning manner on all available clients without clustering and the history plot is shown in Fig. 3. Cluster specific models for cluster 0 and 2 converged after training for forty rounds and that for cluster 1 converged after training for a hundred and twenty rounds. The model with all available clients is trained for two hundred and forty training rounds. Table I shows the evaluation results the cluster specific models and the model trained without clustering, on corresponding clients' test data. Cluster specific models are evaluated on

TABLE I: Test Loss Comparison

		Cluster Specific Model		Model Trained Without Clustering	
Cluster ID	House ID	Rounds	Test Loss (Huber)	Rounds	Test Loss (Huber)
0	3	40	0.00818	240	0.00861
	5		0.00506		0.00516
	14		0.00519		0.00546
1	4	120	0.01060		0.01245
	8		0.01398		0.01333
	9		0.00953		0.00955
	10		0.00844		0.00840
	13		0.01295		0.01315
	19		0.00766		0.01241
2	6	40	0.01817		0.01336
	12		0.00702		0.00452

the test data of clients from the respective clusters while the model trained on all available clients is evaluated on all clients' test data. In Table I, we can observe that there is no significant performance loss between cluster specific models which are trained with a relatively fewer number of participating clients and the model trained without clustering where all clients participate. From the Fig. 2 and 3, it is evident that cluster specific models achieved the optimum significantly faster compared to the single federated model trained with all clients. In Fig. 3, we can observe that training and testing losses for the model trained without clustering slowly felled, but they never fully flattened out because different client households have different housing attributes and thus diverse energy usage patterns. In Table I, for cluster 2, we can observe that the model trained without clustering performs slightly better than the cluster 2 specific model. This is because the cluster 2 specific model is trained on only two clients and the amount of training data was significantly less than that is available for the model trained with all clients. Fig. 4 visualizes the predictions between the cluster specific models and the model trained without clustering on test samples of a client from each of the cluster.

VI. Conclusion

In this paper, we studied the process of developing an RNN based energy demand predictor system by training it using clustering and federated learning based methods. Federated learning allows us to take advantage of distributed data in clients by training local models without the need for data transmission. But different clients can have different data distributions and federated training process of a model can take a long time to converge. To speed up the convergence rate of the global model, clients with similar attributes can be clustered together and the model updates from the clients' of the same clusters can be aggregated together. We experimented by training an RNN model for each cluster of clients and compare the results with a single RNN model trained for all available clients. Our method can reduce the communication cost required for collecting training data onto a server since the local data never left the clients. And from our experiments, we observe that by clustering clients, cluster specific models converge significantly faster compare to a model trained without clustering.

References

[1] V. I. Ugursal], "Energy consumption, associated questions and some answers," *Applied Energy*, vol. 130, pp. 783 – 792, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S030626191300980X

[2] K. Amasyali and N. M. El-Gohary, "A review of data-driven building energy consumption prediction studies," *Renewable and Sustainable Energy Reviews*, vol. 81, pp. 1192 – 1205, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1364032117306093

[3] R. F. Berriel, A. T. Lopes, A. Rodrigues, F. M. Varejão, and T. Oliveira-Santos, "Monthly energy consumption forecast: A deep learning approach," *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 4283–4290, 2017.

[4] J.-Y. Kim and S.-B. Cho, "Electric energy consumption prediction by deep learning with state explainable autoencoder," *Energies*, vol. 12, no. 4, p. 739, feb 2019.

[5] B. Dong, C. Cao, and S. E. Lee, "Applying support vector machines to predict building energy consumption in tropical region," *Energy and Buildings*, vol. 37, no. 5, pp. 545 – 553, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378778804002981

[6] J. G. Jetcheva, M. Majidpour, and W.-P. Chen, "Neural network model ensembles for building-level electricity load forecasts," *Energy and Buildings*, vol. 84, pp. 214 – 223, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378778814006458

[7] R. E. Edwards, J. New, and L. E. Parker, "Predicting future hourly residential electrical consumption: A machine learning case study," *Energy and Buildings*, vol. 49, pp. 591 – 603, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378778812001582

[8] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," *ArXiv*, vol. abs/1909.00590, 2019.

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

[10] J. Konecný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *ArXiv*, vol. abs/1610.02527, 2016.

[11] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, pp. 50–60, 2020.

[12] M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," vol. 28, 06 1999, pp. 49–60.

[13] S. Makonin, "HUE: The Hourly Usage of Energy Dataset for Buildings in British Columbia," 2018. [Online]. Available: https://doi.org/10.7910/DVN/N3HGRN

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

[15] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: https://www.tensorflow.org/

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, Nov. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[18] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[19] Y. L. Tun, K. Thar, and C. Hong, "Federated learning based energy demand prediction of individual households," 2020.