Temporal Ensemble Learning of Univariate Methods for Short Term Load Forecasting

Chung Ming Cheung*, Rajgopal Kannan**, and Viktor K. Prasanna*

*University of Southern California, Los Angeles, California 90089
**US ARL-West, Los Angeles, California 90094
{*chungmin, rajgopalk, prasanna*}@*usc.edu*

***Abstract*—Short term load forecasting (STLF) is a fundamental component of demand response programs in smart grids. Recent literature has focused on complex neural networks and deep learning models for solving STLF. While these models work well for load forecasting with complex non-linear relationships, they have been shown to be less effective than simpler univariate models for STLF problems with under a 6 hours horizon. Given the lack of multivariate data (such as temperature) in many practical datasets, we need better univariate prediction models for STLF. By partitioning the dataset by temporal features, we develop a novel ensemble learning method that exploits daily seasonality in electricity consumption data to improve accuracy of popular univariate models. We train an ensemble of models from the dataset partitions. We develop a variety of methods, including Ridge Regression, to increase the robustness of the ensemble prediction.**

To show the effectiveness of our approach, we perform detailed evaluation using an aggregated user electricity consumption dataset collected by the Los Angeles Department of Water and Power (LADWP). We select four popular prediction algorithms in literature for our experiments, including Kernel Regression (KR), Support Vector Regression (SVR) and neural network approaches. We compare the performance of these algorithms applying our ensemble approach to training only one single model. Our approach leads to an 11.2% decrease in mean absolute percentage error (MAPE) and 21.3% decrease in root mean squared error (RMSE) over the single model approach for KR, and a 30% and 32.4% decrease in MAPE and RMSE respectively for SVR. These ensemble models also outperform the neural network approaches.

***Keywords*-Smart grids; Short Term Load Forecasting; Machine Learning; Ensemble Learning;**

I. Introduction

Accurate short term load forecasting (STLF) is essential for the successful implementation of smart grids. In particular, it plays an important role in facilitating demand response programs [1]. Demand response programs bring many benefits to smart grids, including reduction in electricity prices and reducing the risks of power outages by balancing power demand and supply in the market. In order to make appropriate demand side management decisions like curtailment strategies, accurate predictions of demand over short horizons are needed [2].

[1]This work has been funded by the U. S. Department of Energy under Award Number DE-EE0008003, and the U.S. National Science Foundation under grant number ACI 1339756

STLF is very well studied and many models have been proposed for accurate load predictions [3]. One class of models used for STLF are univariate methods, where only the consumption time series is used as input for predicting future load consumption. Such models are studied because non-load data can be missing for various reasons, for example, in [4], their dataset from Brazil has no weather information due to difficulty in acquiring such data.

In recent years, advancements in processor computational power facilitated the rise of deep learning techniques. These complex models have the capacity to model highly non-linear dependencies. Researchers have attempted to bring deep neural network methods into load forecasting and acquired promising results [5] [6]. However, such methods usually require large amount of training data and long training time as a tradeoff for being able to learn the complex relationships between the inputs and outputs. In [7], it is also shown that neural network models may not work too well if the problem is not complex enough (prediction horizon is under 6 hours) to require the high non-linearity of such models. STLF is one such case, especially when the data is an aggregation dataset where the time series is an aggregation of the load of many customers. The aggregation process smooths the data and reduces the effects of noise. Therefore, for improving the accuracy of this problem, developing more complex models may not be a good direction to proceed. Instead, we believe ensemble learning is a better way to enhance the accuracy of existing models. Ensemble learning [8] is the use of multiple models, typically simple ones, to model one hypothesis. It is especially effective in the case where there is great diversity among the models.

We propose a novel ensemble learning method that partitions the dataset by temporal features. We call this Temporal Ensemble Learning (TEL). Due to the lack of non-load data, we utilize the only exogenous information available - the temporal dimension. The motive for this approach is the high daily seasonality in customer consumption patterns, e.g. load peaks at similar daily times. To exploit this information, we partition the data temporally and develop sub-problems of forecasting at specific time ranges per day. Each sub-problem is then trained on datasets with lower variance, allowing it to find a better fit. Our model differs from popular ensemble models like Random Forest [9] in that there is no randomness in the partitioning of the dataset. Instead, we exploit the inherit temporality of the dataset by choosing time of day as the partitioning feature and vary the threshold for partitioning as a hyperparameter. The models may become less robust due to each model seeing a smaller amount of data. We overcome this problem by combining the results of the ensemble with three different methods, which will be detailed in Section IV.

Our paper makes the following novel contributions:

1) We develop a novel ensemble learning method partitioning with temporal features that significantly improves the accuracy of existing prediction models for STLF.
2) We develop a novel method for combining results of the models in the ensemble through ridge regression that overcomes the challenge of training with smaller partitioned datasets.
3) We show high accuracy prediction results on a real-life electricity consumption dataset from Los Angeles with our approach.

II. Problem Definition

In this paper, the problem of short term load forecasting (STLF) is studied. Given a time series with T values $\mathbf{X} = \{X_1, X_2, X_3, ..., X_T\}$, the goal is to solve the multi step ahead prediction problem. With values of the time series up to time t, we would like to predict the values of the following W intervals. We take the approach of multi-step forecasting instead of one-step ahead in this paper, i.e., $W > 1$.

Denote $\mathbf{X}_{a:b}$ as the values of the time series from interval a to b. The prediction problem is then formally written as finding the prediction function f where

$$\mathbf{\hat{X}_{t+1:t+W}} = f(\mathbf{X_{1:t}}) \tag{1}$$

such that the error function $E(\mathbf{X}_{t+1:t+W}, \hat{\mathbf{X}}_{t+1:t+W})$ is minimized.

The training and testing dataset is formed by shifting a sliding window across the full time series. Size of the window S determines the history size to take per data point, i.e., the number of intervals before the prediction is made to take as input. Each shift of the window creates a data point, where the S values inside the window is taken as input, and the W values following it are the target values to predict.

The accuracy of the prediction is evaluated by Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE), which are two commonly used evaluation measures for STLF problems. We denote W as the number of intervals to predict and N as the number of testing data points. $\hat{\mathbf{X}}^i = \{\hat{X}_1^i, \hat{X}_2^i, \hat{X}_3^i, ..., \hat{X}_W^i\}$ for $i = 1..N$ denotes the predicted values and $\mathbf{X}^i = \{X_1^i, X_2^i, X_3^i, ..., X_W^i\}$ for $i = 1..N$ denotes the actual values.

$$MAPE = \sum_{i=1}^{N} \sum_{j=1}^{W} | \frac{\left(\hat{\mathbf{X}}_j^i - \mathbf{X}_j^i\right)}{\mathbf{X}_j^i} | \frac{1}{NW} \tag{2}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \sum_{j=1}^{W} \left(\hat{\mathbf{X}}_j^i - \mathbf{X}_j^i\right)^2}{NW}} \tag{3}$$

MAPE computes the mean percentage errors of all the predicted values from its actual values. This measure of evaluation normalizes the results such that the magnitude of error is not affected by how large the values of the outputs are. RMSE on the other hand is greatly affected by the actual values of the outputs, however, it is more sensitive to outliers because the squared of the deviation between the prediction and actual value is taken.

III. Background

A. Related Work

Popular approaches for STLF include the use of statistical methods for modelling time series, e.g. ARIMA [10]. Traditional machine learning models like Support Vector Regression [11] are also suitable for solving STLF.

Artificial Neural Networks are also popular for their ability to model any function with no assumptions on the data structure. Multi-layer Perceptron networks are widely used for their simplicity [12]. Other kinds of neural networks are also used for performing dimension reduction on time series data, for example, self organizing maps [13]. Recently, deep learning techniques are gaining popularity, and researchers have tried various deep network models to improve the accuracy of STLF models [5], [6].

Our proposed approach draws inspiration from ensemble methods like Random Forest [9]. However, unlike these techniques, we do not randomize our partitioning step.

We are not the first work to propose the use of temporal dimension for STLF. [14] defines one step ahead prediction models for each timestep in a day. [15] solves the hour ahead prediction problem by defining 12 models, one for each hour of the day. Our work is unique in that we define models that contain overlaps in their prediction horizon. We also discover similar models via ridge regression so that their knowledge can be shared. We show that combining the results of multiple models can improve accuracy of the predictions.

B. Baselines

In our experiments, we used the following algorithms as baselines for fitting the prediction model. We picked machine learning based techniques that have been shown to work well for the STLF problem, and experimented with using TEL on these algorithms.

1) Kernel Regression (KR): Linear regression is the most simple kind of regression, where we try to find a weight vector $\mathbf{W}$ such that it the multiplication of it with the input data $\mathbf{X}$, that is, $\mathbf{WX}$ is as close to the output $\mathbf{Y}$ as possible.

However, this only captures linear relationships, so it is often kernelized [16] to allow non-linear dependencies to be modeled. We use RBF kernels in our experiments.

2) Support Vector Regression (SVR): SVR [17] is an application of the popular idea of support vectors in machine learning for solving the regression problem. Similar to KR, we use RBF kernels [18] to allow it to capture non-linear dependancies.

3) Multi-layer Perceptron (MLP): MLP is the most basic kind of neural networks. [19] It is composed of perceptrons that take a weighted average of the inputs, pass it through an activation function, and output it. Its output would usually be used as the input for the next layer of perceptrons, creating multiple layers that can model complex functions.

4) Recurrent Neural Networks (RNN): RNN is a kind of neural network that specializes in capturing sequential relationships. Each iteration takes both input features and weights from the previous iteration as the input. Long Short Term Memory (LSTM) [20] is a variation of RNN which makes use of LSTM nodes that can retain values through many iterations by using gates to control when and how much to "forget" old values. This overcomes the problem with traditional RNNs where it is difficult to capture long term dependencies. We use LSTM networks for our experiments.

IV. Methodology

Electricity load consumption time series show great daily seasonality. Load usually rise and drop in similar times of the days. This can be seen in the actual load plot of Fig. 2 and 3. We want to capture this information in a prediction model without making it more complex by increasing its number of inputs. To do so, we propose the Temporal Ensemble Learning (TEL) model, which is to partition the dataset by the prediction time of the day, and train models for each partition.

Suppose we were to partition the dataset into G groups. We first split the time of a day into G consecutive periods of equal length. For example, if $G = 3$, we can split the day into periods 12am-8am, 8am-4pm, and 4pm-12am. The data points of the training and testing dataset is then partitioned into the corresponding period if the starting prediction time is in the period. In the previous example, a data point for predicting the values from 2pm would be grouped into period 8am-4pm.

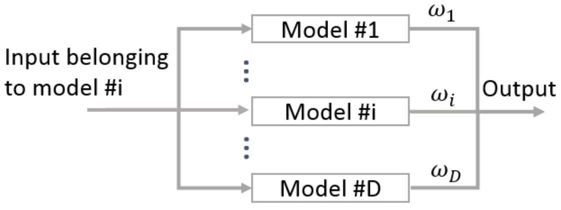

Fig. 1. Combining results by taking weighted average output from all models.

By doing this partitioning, we expect data points within each cluster to have lower variance and less outliers. This allows each model to be trained to find a better fit for its partition of data. However, with more than one model for making predictions, there needs to be a way to combine the results of all models to give a final result. We develop three distinct methods that differ in the way the model weights are allocated.

1) Take the result from the model that was trained on samples in the same group. To predict the load at 4pm, use the model with 4pm in its period of time of the day.
2) Take a weighted average of the outputs of all models. The weights are determined arbitrarily. In our experiments, we take the weight 0.5 for the model of the same group, and the remaining weight spread evenly among the rest of the models.
3) Take a weighted average of the outputs of all models, with the weights trained by the ridge regression defined below.

Denote $\mathbf{X}^i$ as the W actual values of the i-th data point in the dataset, where W is the prediction horizon and D is the number of models, then $\hat{\mathbf{X}}^i$ is the $W \times D$ matrix with the predicted values given by the models for the i-th data point. The ridge regression for method 3 can then be written as

$$\min \sum_{i \in \{training\}} \| \omega^T \hat{\mathbf{X}}^i - \mathbf{X}^i \|_2^2 + \alpha \| \omega \|_2^2 \qquad (4)$$

where ω is the vector of weights we are training for.

We train a set of weights for data points belonging to each partition. Given the training set of data points belonging to one partition, we find the prediction that is outputted by each model, then find the weights by optimizing the objective function in Equation (4). The intuition behind this is that there may be models that are similar, thus we can place higher confidence on the outputs of these models and give their outputs a higher weight. As an example, the data points from 2am and 6am may follow similar patterns, thus the regression can learn to take higher weight from the model for 6am when predicting for 2am.

Figure 1 illustrates the process of combining outputs of the models. Considering an incoming input i that is in the same partition as data points model #i trained on. For method 1, the weights have values 0 except for ω_i with value 1. For method 2, the weights are arbitrarily determined as described previously. For method 3, the weights are trained by ridge regression as defined above.

V. Experimental Results

A. Dataset

Our experiments are carried out on electricity consumption data collected by LADWP from May 21st 2015 to October 21st 2015. The data is an aggregation of 50k customers' data and gives the aggregated power consumption in 15 minute intervals. This gives us a total of 154 days of data with 96 values per day. We divide the data into 3 groups: the first 70% of the days for training, the following 15% for validation, and the last 15% for testing.

12 intervals contain values of 0. Data points containing these values are discarded as they are invalid.

B. Experimental Setup and Hyperparameters

We investigated the problem of forecasting 1 hour of load into the future, i.e., number of prediction intervals $W = 4$.

Our experiments are performed on a Ubuntu 14.04.4 LTS system with 32 cores, 128 GB RAM and a clock speed of 2.6 GHz. The GPU used was Nvidia Tesla K40C.

For input features, we experimented with taking $\{4, 16, 32\}$ intervals' values before the intervals to be predicted, we call this the history size. Also, the values of the intervals $\{0, 1, 2\}$ day(s) before at the same time of the day and the history on that day are taken as inputs. For example, if we take a history size of 4 and number of days in the past as 1, the input would consist of a total of 12 values - 4 values from before the prediction time on the prediction day and 4 more for that of the previous day, plus 4 more values which are the values in the prediction period but on the previous day. A binary value indicating whether the time of prediction is on a weekday or weekend is also included. Temperature data was not used because geographical location of the customers were not provided in the dataset.

For TEL, information regarding the time of prediction is already embedded in the method itself, so we do not include input features that specify such information. We use training a single model as a baseline. For the single model case, a binary vector is used to indicate which partition the data point would have been categorized if TEL is used. This provides the time of day information for the model. For both methods, we vary the number of partitions as a hyperparameter.

For MLP and RNN models, besides the hyperparameters that relates to the input features, we vary other hyperparameters, like the number of hidden layers and units, and use a validation dataset to determine the optimal hyperparamters by grid searching.

C. Results

TABLE I
SUMMARY OF STLF RESULTS FOR VARIOUS MODELS AND METHODS.

	TEL		Single Model	
	MAPE(%)	RMSE	MAPE(%)	RMSE
KR	1.03	124.41	1.16	158.13
SVR	1.05	126.41	1.50	186.97
MLP	2.63	292.52	1.56	181.10
RNN	2.58	266.75	1.56	200.54

TABLE II
SUMMARY OF BEST HYPERPARAMETERS CHOSEN FOR EACH METHOD.

	History size	No. of Groups	No. of days into past	Method of combination
KR (TEL)	4	96	1	Method 3
KR (Single)	16	24	1	N/A
SVR (TEL)	4	96	1	Method 3
SVR (Single)	16	1	1	N/A
MLP (TEL)	32	16	0	Method 1
MLP (Single)	32	1	2	N/A
RNN (TEL)	32	16	2	Method 1
RNN (Single)	32	1	0	N/A

Results of the experiments are summarized in Table I. It shows the MAPE and RMSE for the combination of each algorithm and TEL or single model approach. Figure 2 and 3 plots the one hour ahead predictions made by KR and SVR on the testing dataset in 5 days to illustrate its accuracy.

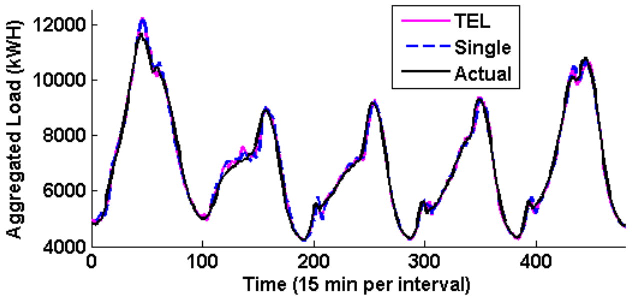

Fig. 2. One hour ahead predictions in 5 days made by KR with the testing dataset.

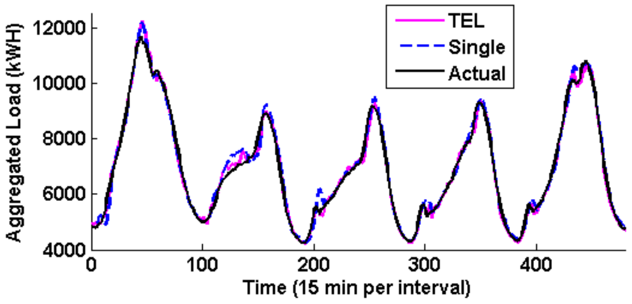

Fig. 3. One hour ahead predictions in 5 days made by SVR with the testing dataset.

The hyperparameters which gave the best model for each algorithm is shown in Table II. History size is the number of intervals to take as input before the prediction. Number of groups is the number of partitions. Number of days into past is the number of past days data to take as input. Method of combination is the method out of the three described in Section IV chosen to be used for combining results of the multiple models.

D. Discussion

From the results shown in Table I, we can see TEL performed better than single model for KR and SVR. The best performance was given by Kernel Regression, and using TEL over single model improved the RMSE by 21.3% and the MAPE by 11.2%. The decrease in error was even greater for SVR, with RMSE dropping by 32.4% and MAPE by 30%. This shows that multiple simpler models lead to more accurate models than trying to model the behavior with one model for a high complexity problem. The opposite holds true for MLP and RNN, where TEL performed worse than single model. Overall, TEL for KR and SVR outperformed single model for MLP and RNN. We draw a similar conclusion as [7] where neural network approaches does not perform well for STLF compared to simpler models. This may be due to the problem being simple enough for KR and SVR to easily be trained to capture the patterns, while neural networks are harder to train comparatively due to its greater number of weights and hyperparameters. For the same reason, KR and SVR benefits from using TEL while neural network approaches performed worse, since each model in TEL solves a further simplified STLF problem than the single model approach.

Observing the hyperparameters that gave the best performing models in Table II, we can see how some of the design intuitions led to a better performing model. For all models, we can see that using at least 1 previous day of data for the same interval gave better results than using only current day data in the validation step. For the simpler models like KR and SVR, 96 groups gave the best results. Since these models are unable to capture complex relationships between the input features compared to neural network approaches, they perform better when the prediction task is as simplified as possible. This statement is further supported by the fact that KR and SVR chose 4 as the history size, indicating that the most important features for predicting 1 hour ahead is the hour before it and also the values of the data for the hour of prediction in the previous day.

Finally, for the best performing models KR (TEL) and SVR (TEL), method 3 is chosen for the method of combining results over the other two methods. This shows that allowing models in the ensemble to make use of knowledge learned in models with similar characteristics improved results.

VI. Future Work

We will explore how the results hold when the prediction horizon is extended. As the problem gets more complex, it is possible that neural network models will perform better than KR due to their high modeling capability. We will investigate the results for 6 and 24 hours ahead prediction horizon.

Aside from splitting the dataset by time of day, other features should also be looked into. The most relevant one is training separate models for similar users. Aggregation of users with similar characteristics [22] is a popular topic. Performing STLF on a more local level [23] is also gaining interest. By grouping similar users to train models, it can improve results for local level STLF, as information learned by the model can be shared among similar users to overcome the challenge of higher noise.

We will investigate the application of TEL to statistical methods like ARIMA [10] and see how it compares to machine learning approaches. The performance of the model can also be compared to other ensemble learning methods like Random Forest [9] to highlight the benefits of utilizing temporal features for the partitioning.

VII. Conclusion

In this paper, we investigated the STLF problem for electricity load consumption prediction. We proposed the Temporal Ensemble Learning approach which partitions the dataset by temporal features to create an ensemble of univariate models. For 1 hour prediction horizon, this method was shown to improve accuracy for simpler machine learning models like Kernel Regression and Support Vector Regression, while the relatively more complex neural network models did not work well. One reason for this was the simpler models suffice for the 1 hour ahead STLF problem. Partitioning the dataset also reduced the number of data points for the training of each model, making it more difficult to train neural networks than simpler models.

References

[1] M. H. Albadi and E. F. El-Saadany, "A summary of demand response in electricity markets," *Electric power systems research*, vol. 78, no. 11, pp. 1989–1996, 2008.

[2] C. C. Sanmukh R. Kuppannagari, Rajgopal Kannan and V. K. Prasanna, "Implementation of learning-based dynamic demand response on a campus micro-grid," *25th International Joint Conference on Artificial Intelligence, IJCAI-16 Demo Track.*, 2016.

[3] A. Srivastava, A. S. Pandey, and D. Singh, "Short-term load forecasting methods: A review," in *International Conference on Emerging Trends in Electrical Electronics & Sustainable Energy Systems (ICETEESES)*. IEEE, 2016, pp. 130–138.

[4] L. J. Soares and L. R. Souza, "Forecasting electricity load demand: Analysis of the 2001 rationing period in brazil," *Economics Working Papers (Ensaios Económicos, 486)*, 2003.

[5] S. Hosein and P. Hosein, "Load Forecasting using Deep Neural Networks," *7th IEEE International Conference on Smart Grid Communications (SmartGridComm 2016)*, pp. 7046–7051, 2016.

[6] E. Mocanu, P. H. Nguyen, M. Gibescu, and W. L. Kling, "Deep learning for estimating building energy consumption," *Sustainable Energy, Grids and Networks*, vol. 6, pp. 91–99, 2016.

[7] J. W. Taylor, L. M. De Menezes, and P. E. McSharry, "A comparison of univariate methods for forecasting electricity demand up to a day ahead," *International Journal of Forecasting*, vol. 22, no. 1, pp. 1–16, 2006.

[8] T. G. Dietterich, "Ensemble learning," *The handbook of brain theory and neural networks*, vol. 2, pp. 110–125, 2002.

[9] A. Liaw, M. Wiener *et al.*, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[10] J. W. Taylor and P. E. McSharry, "Short-term load forecasting methods: An evaluation based on European data," *IEEE Transactions on Power Systems*, vol. 22, no. 4, pp. 2213–2219, 2007.

[11] A. Kavousi-Fard, H. Samet, and F. Marzbani, "A new hybrid modified firefly algorithm and support vector regression model for accurate short term load forecasting," *Expert systems with applications*, vol. 41, no. 13, pp. 6047–6056, 2014.

[12] P. A. González and J. M. Zamarreño, "Prediction of hourly energy consumption in buildings based on a feedback artificial neural network," *Energy and Buildings*, vol. 37, no. 6, pp. 595–601, 2005.

[13] L. Hernández, C. Baladrón, J. M. Aguiar, B. Carro, A. Sánchez-Esguevillas, and J. Lloret, "Artificial neural networks for short-term load forecasting in microgrids environment," *Energy*, vol. 75, pp. 252–264, 2014.

[14] G. Escrivá-Escrivá, C. Álvarez-Bel, C. Roldán-Blay, and M. Alcázar-Ortega, "New artificial neural network prediction method for electrical consumption forecasting based on building end-uses," *Energy and Buildings*, vol. 43, no. 11, pp. 3112–3119, 2011.

[15] C. Guan, P. B. Luh, M. A. Coolbeth, Y. Zhao, L. D. Michel, Y. Chen *et al.*, "Very short-term load forecasting: Multilevel wavelet neural networks with data pre-filtering," *2009 IEEE Power & Energy Society General Meeting*, vol. 28, no. 1, pp. 30–41, 2009.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Kernel methods in machine learning," 2008.

[17] D. Basak, S. Pal, and D. C. Patranabis, "Support vector regression," *Neural Information Processing-Letters and Reviews*, vol. 11, no. 10, pp. 203–224, 2007.

[18] C.-W. Hsu, C.-C. Chang, C.-J. Lin *et al.*, "A practical guide to support vector classification," 2003.

[19] C. M. Bishop, *Neural networks for pattern recognition*. Oxford university press, 1995.

[20] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *9th International Conference on Artificial Neural Networks*, pp. 850–855, 1999.

[21] K. Tornai, A. Olah, R. Drenyovszki, L. Kovacs, I. Pinter, and J. Levendovszky, "Recurrent Neural Network Based User Classification for Smart Grids," *IEEE Innovative Smart Grid Technologies*, 2017.

[22] B. Hayes, J. Gruber, and M. Prodanovic, "Short-Term Load Forecasting at the local level using smart meter data," *2015 IEEE Eindhoven PowerTech*, no. February, pp. 1–6, 2015.