Fast Demand Forecast of Electric Vehicle Charging Stations for Cell Phone Application

Mostafa Majidpour, Charlie Qiu, Ching-Yen Chung, Peter Chu, Rajit Gadh
Smart Grid Energy Research Center
UCLA
Los Angeles, California USA
mostafa@ee.ucla.edu

Hemanshu R. Pota
School of Engineering & Information Technology
The University of NSW
Canberra ACT 2610 Australia

***Abstract*—This paper describes the core cellphone application algorithm which has been implemented for the prediction of energy consumption at Electric Vehicle (EV) Charging Stations at UCLA. For this interactive user application, the total time of accessing database, processing the data and making the prediction, needs to be within a few seconds. We analyze four relatively fast Machine Learning based time series prediction algorithms for our prediction engine: Historical Average, k-Nearest Neighbor, Weighted k-Nearest Neighbor, and Lazy Learning. The Nearest Neighbor algorithm (k Nearest Neighbor with k=1) shows better performance and is selected to be the prediction algorithm implemented for the cellphone application. Two applications have been designed on top of the prediction algorithm: one predicts the expected available energy at the station and the other one predicts the expected charging finishing time. The total time, including accessing the database, data processing, and prediction is about one second for both applications.**

***Index Terms*—Prediction Methods, Electric Vehicles, Nearest Neighbor Searches, Cellphone Applications.**

I. INTRODUCTION

Electric Vehicles (EVs) (or Plug-in Hybrid Electric Vehicles (PHEV)) will be an important part of the Smart Grid. The big challenge for EVs is their charging within the existing distribution system infrastructure. According to the EV33 rule (33 miles driving range for a single charge [1]), the minimum battery size in EVs varies from 8.6 kWh to 15.2 kWh [2]. Charging batteries with the aforementioned size will take between four and eight hours in a Level 1 household charger (120V, 16 A). As an alternative, EV owners can charge their vehicle at their place of employment, provided that the employer has installed the chargers in the parking lot. Other than workplace parking lots, cities and private operators have invested in charging stations for public use. Extensive research has focused on EV charging algorithms and charging station infrastructures [3]-[6]. As of October 2013, the number of these stations in the US has reached 19730 which is roughly a sixth of the number of gas stations in the US [7]. Electric power, has to be consumed simultaneously upon generation. As a result, the EV charging station needs to have a good idea of how many vehicles will need charging at each moment. On the other hand, since a reasonable amount of charging will take tens of minutes, it would be useful for a customer to know when and how much energy is expected to be available at a given charging station in a given time window. Both the customer and moderator will substantially benefit from an algorithm that can predict the power consumption at charging stations. With emerging smart phones, it would be beneficial to customers if they could receive the prediction information on their cellphone, giving them an estimate of the predicted charging time. This paper discusses prediction algorithms that can run in less than a few seconds, so that EV users can query the system and get the results in a reasonable time on their cellphones. To our knowledge, this is the first paper that discusses fast prediction of the available energy and/or expected charging finishing time at the charging station for use as a cellphone application. The data used in the algorithm is obtained from the charging records. A charging record in this paper is a datum that contains start and end time of a charging transaction as well as the total amount of received energy by EV in kWh. The charging records database consists of the historical charging data for a given charging station.

Time series prediction (forecasting) methods predict the future of a certain variable given its past history. There is a rich literature on different methods of time series prediction evolved from statistics, mathematics, computer science, economics, and engineering [9]-[11]. EV related research has started to utilize the forecasting algorithms [12]-[18]. Some papers apply forecasting algorithms to EV driving habits and predict the State of Charge (SOC) of a particular EV and when it needs to be charged [13],[14]. Authors in [12] have applied Artificial Neural Network (ANN) forecasting algorithms to predict the charging profile of the EV within the Building Energy Management System (BEMS) in order to improve the overall energy efficiency of the building. Reference [15] and [16] discuss the prediction of the EV charging profile while taking into account various sources of data, such as vehicle driving/usage data. Authors in [17],[18] consider forecasting at the charging station level based on the EV user classification and Monte Carlo simulations method.

The work in this paper is different from all the previously mentioned works: 1) we have used just one type of recorded

This work has been sponsored in part by grant from the LADWP/DOE fund 20699 & 20686, Smart Grid Regional Demonstration Project.

data, Charging Records, which only contains the start and end of the charging transaction and the total amount (a scalar value; not time dependent) of energy received in the charging transaction rather than geographical or any driving habit related data; 2) our predictions is at the charging station level (not parking lot or the whole building level); 3) our method is online and fast and the whole process takes about a second. Indeed, the reason that we don't use other sources of data is our speed requirement and the fact that adding more data will slow the process.

As mentioned above, the aim of this work is to focus on relatively fast time series forecasting algorithms in which the whole process of prediction (including pre-processing) takes a reasonably short amount of time for a user that sends prediction-related queries. Machine Learning (ML) based heuristic methods provide a good performance in forecasting [8]. For our predictor application, we are interested in the following four ML based algorithms: Historical Average, k-Nearest Neighbor, Weighted k-Nearest Neighbor, and Lazy Learning that have low computation requirements and are relatively faster. A brief introduction of these methods will follow in Section III.

The rest of this paper is organized as follows: Section II formulates the problem, Section III explains the methods that are applied in this paper. Section IV reports and analyzes the result of applying the algorithms on the UCLA (University of California, Los Angeles) parking structures' data. Section V explains the selected algorithm, and Section VI is on conclusion and future work.

II. Problem Formulation

The objective is to predict the available energy in the next 24 hours at each charging station with a minimum time for processing. Formally, we assume there is some function relating future available energy and the past consumed energy:

$$\hat{E}(t) = f\big(E(t-i), \varepsilon(t)\big) \quad i \in \{1,2,\dots\} \qquad (1),$$

where $E(t)$ is the actual energy consumption at time t, $\hat{E}(t)$ is the prediction of the energy consumption at time t, $\varepsilon(t)$ is the set of all variables (such as noise) other than past energy consumption records, $E(t-i)$, that f might depend on.

As the usual practice in forecasting, we are interested to find some estimation of $E(t)$ according to some distance criteria. For the distance (error) measurement, we have chosen Symmetric Mean Absolute Percentage Error (SMAPE). For the day i, the SMAPE is defined as:

$$SAMPE(i) = \frac{1}{H}\sum_{t\in Day(i)} \frac{|E(t)-\hat{E}(t)|}{E(t)+\hat{E}(t)} \times 100, \qquad (2),$$

where H(=24 in this paper) is the horizon of prediction in a given day.

Also, in practice, since there is no access to future data, the last portion of the data (last 10% in this paper) is set aside as the test set to evaluate the performance of the algorithm. Thus, our goal is to find an algorithm that minimizes the error between actual value and its prediction on the test set. Note that the test set is not used in either parameter selection or training phase.

We use the notation $\hat{\underline{E}}(t)$ as the vector of prediction for the next 24 hours ending at t (Fig 1. a). Also, $t_r = \{1,2,\dots,N_{tr}\}$ and $t_s = \{N_{tr}+1,\dots,N\}$ are the set of indexes for training and test sets respectively.

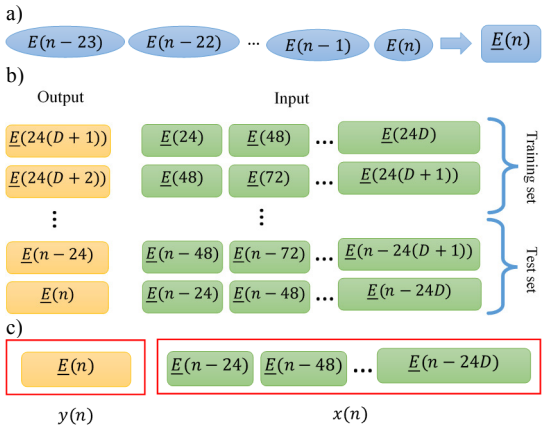

Figure 1. a) energy consumption vector ($\underline{E}$) for 24 hours , b) input-output pairs and division of data into training and test sets, c) labeling inputs as x and outputs as y.

III. Algorithms

Four employed prediction algorithms have been described here briefly. A detailed description can be found in [3].

A. Historical Average

This algorithm is one of the simplest algorithms that sometimes is referred to as naïve approach and is used only for comparison with other methods. According to this approach, the available charging energy in the future is the average of the charging energy consumption in the past. Formally:

$$\hat{E}(t) = \frac{1}{D}\sum_{d=1}^{D} E(t-24d), \qquad (3)$$

where D is the depth of the averaging. For instance, the predicted energy for 3pm on the next day is equal to the average of the energy consumed today, yesterday…, and up to the past D days at 3pm. Here, D is a parameter that needs to be selected before the evaluation on the test set.

B. K-Nearest Neighbor

This algorithm is a well-known algorithm in the machine learning community [22]. Based on k-Nearest Neighbor (kNN) algorithm, each sample (training, test or validation) is composed of input and output pairs. In our application, the output is the energy consumption for the next 24 hours, $y(t) = \underline{E}(t)$, and the input is the concatenation of the consumption records for up to D previous days, $x(t) = \{\underline{E}(t-24), \underline{E}(t-48), \dots, \underline{E}(t-24D)\}$ (Fig. 1. c). This concatenation repeats for all days: if there are N days in the data set, there will be N-D+1 of these input-output pairs (Fig 1.b). The total number of data points is $n = 24N$. Now, in order to find an estimate for $y(t_{s^*})$ where $t_{s^*} \in t_s$ is an instance of test set indexes, first the distance between $x(t_{s^*})$

and all other $x(t_r)$ that belong to the training set is computed. Distance could be any norm of their difference; we have used the Euclidian distance here. After determining the k closest $x(t_r)$ to $x(t_{s^*})$, the average of their corresponding $y(t_r)$ is generated as $y(t_{s^*})$. In this algorithm, the parameter k needs to be determined. Fig. 2 illustrates the algorithm.

k-Nearest Neighbor Algorithm

Inputs: $x(t_r), y(t_r), x(t_{s^*}), k$
Output: $y(t_{s^*})$
1. **for** $j \in t_r$
2. $\quad dist[j] = \|x(t_{s^*}) - x(j)\|$
3. **for** $i \in \{1, \dots, k\}$
4. $\quad idx[i] = \text{index of } i^{th} \text{ smallest}(dist)$
5. $y(t_{s^*}) = \frac{1}{k}\sum_{i\in\{1,\dots,k\}} y(idx[i])$

Figure 2. k-Nearest Neighbor Algorithm.

C. Weighted k-Nearest Neighbor

This algorithm is very similar to the original kNN algorithm with the exception that instead of averaging the k closest training outputs (Fig. 2, step 5) their weighted average is used where the weights are a function of the distance between input pairs. Thus, these two steps will substitute for Step 5 in Fig. 2:

$$w_p = \frac{1/dist(p)}{\sum_{q\in idx} 1/dist(q)} \quad (4),$$

$$y(t_{s^*}) = \sum_{p\in idx} w_p y(p) \quad (5).$$

Weighted kNN is based on the idea that closer points to the query should contribute more to the output and in this way improve the accuracy of the prediction in comparison to kNN.

D. Lazy Learning

Lazy Learning (LL) is a generic term which refers to algorithms that postpone the learning until a query is submitted to system. In fact, weighted kNN and kNN are themselves simple forms of LL. A version of LL according to [23] has been implemented in this paper. This version of the LL algorithm is similar to the kNN algorithm in principle, with the difference that for each query, the optimum number of neighbors (k) is not fixed and is estimated separately. The idea is that for some queries it seems better to look at more neighbors and for some others, fewer neighbors would be enough. For each query, kNN is performed k_{max} times for $k = \{1,2, \dots, k_{max}\}$. Then, based on Leave-one-out (LOO) cross validation, the error for each k is estimated, and the output corresponding to the k with a lower LOO cross validation error is selected. We use the PRESS statistic [23] to estimate LOO for each k. For a fixed $k \in \{1,2, \dots, k_{max}\}$, suppose $j^* \in \{1,2, \dots, k\}$ indicates the index of the j[th] closest neighbor to the query $x(t_{s^*})$. For each j^*, we define an error term

$$e_k(j^*) = k\frac{y(j^*) - y(i)}{k-1} \quad (6),$$

which gives the LOO cross validation error for the specific k:

$$e_{LOO}(k) = \frac{1}{k}\sum_{j^*=1}^{k} (e_k(j^*)^T * (e_k(j^*)) \quad (7).$$

The k with smallest e_{LOO} will be selected as the optimum k.

The steps are detailed in Fig. 3.

Lazy Learning Algorithm

Inputs: $x(t_r), y(t_r), x(t_{s^*}), k_{max}$
Output: $y(t_{s^*})$
1. **for** $j \in t_r$
2. $\quad dist[j] = \|x(t_{s^*}) - x(j)\|$
3. **for** $i \in \{1, \dots, k_{max}\}$
4. $\quad idx[i] = \text{index of } i^{th} \text{ smallest}(dist)$
4. $\quad$ **for** $k \in \{2, \dots, k_{max}\}$
5. $\quad\quad y(k) = \frac{1}{k}\sum_{i\in\{1,\dots,k\}} y(idx[i])$
6. $\quad\quad$ Calculate $e_{LOO}(k)$ according to Eq. (7)
7. $\quad k^* = \arg(\min_{k\in\{2,\dots,k_{max}\}} e_{LOO}(k))$
8. $y(t_{s^*}) = y(k^*)$

Figure 3. Lazy Learning Algorithm.

IV. Simulation and Analysis

A. Data and Preprocessing

The algorithms described above are applied to charging stations located on the UCLA campus. The data used in this paper were recorded from December 7, 2011 to October 16, 2013; however, not all stations were in use all days. Among charging stations at UCLA, 15 stations have charging data for more than 60 effective days (days that some nonzero charging has been reported); these stations have been used in our implementation. The number of effective days for each station is reported in Table III.

Data for each station is in the format that is called Charging Records. Each charging record contains the beginning and end of the charging time as well as the acquired energy. The Charging Records are converted to time series by uniformly dividing the acquired energy to the charging interval; e.g. if charging interval is 3 hours and the acquired energy is 3kWh, it is assumed that the EV received 1kWh of energy in each hour. In pre-processing the data, if all the values in an input-output pair $(x(i), y(i))$ are zero or not reported, the pair was removed from the data set.

There was no normalization or feature extracting from the data. The only implemented pre-processing was to force energy records that were mistakenly recorded as more than the physical maximum of the charging device (E_{max}) and less than zero to the interval of $[0, E_{max}]$.

B. Parameter Selection

The following parameters need to be determined for our algorithms: Depth, D, for all algorithms, which is the number of previous days considered in the input vector and the number of neighbors, k, in kNN and weighted kNN.

In order to select the parameters in each algorithm a k-fold cross validation method [19] has been employed. In k-fold cross validation, for evaluating a certain set of candidate parameters, the training data is divided into k parts and algorithm trains on k-1 parts while the error is calculated on the remaining part, i.e., the validation set. This process iterates k times, and each time one of the parts will be the validation

set and the other k-1 parts will make up the training set. The average error of the algorithm on these k iterations will determine the final error for the current selection of parameters. The whole process is repeated for different combinations of parameters and the combination that yields the least final error is selected as algorithm parameters. For k-fold cross validation, k=10 is a typical choice for k [20], and this value was used in our experiments too.

In cross validation, the depth parameter, D, is varied between 1 to 60 (equal to looking only at yesterday or up to past two months) and the number of neighbors varied between 1 to 10 for kNN and 2 to 10 for weighted kNN (weighted kNN with k=1 is the same as kNN with k=1).

C. *Results and analysis*

Table II and III show the depth (D) and number of neighbor (k) parameters, selected through cross validation, for each site and each parameter.

Fig. 4 shows the prediction for a sample test day from Station 2. The SMAPE for this specific day is 15.2 %.

Table IV shows the average and standard deviation of SMAPE on test days for each algorithm and each station.

Interestingly, as can be seen in Table II, the optimum number of neighbors is chosen to be equal to 1 for kNN algorithm for all stations. This shows that, regardless of the optimum number of previous days to forecast, it is always better to look at the most similar event in the past and copy its future energy consumption values as the prediction. Also, the optimum number of neighbors for weighted kNN in all stations is 2; considering that k ranges from 2 to 10 and the kNN algorithm with k=1 shows better performance, one can conclude that k=1 is the optimum parameter. Thus, for all stations (except station no. 12) the kNN (or weighted kNN) with k=1 is the best algorithm to pursue.

TABLE I. SELECTED DEPTH PARAMETER FOR EACH ALGORITHM BASED ON CROSS VALIDATION RESULTS

No	Station	Historical Average	kNN	Weighted kNN	Lazy Learning
1	PS3L401LIA3	1	39	35	36
2	PS8L201LIA1	1	59	59	59
3	PS8L201LIA3	1	60	59	59
4	PS8L201LIA4	1	57	57	57
5	PS8L202LIIA1	1	3	3	3
6	PS8L202LIIA2	1	11	4	11
7	PS8L202LIIA3	1	6	12	12
8	PS9L401LIA1	1	20	53	1
9	PS9L401LIA2	1	22	24	24
10	PS9L401LIA3	1	28	28	28
11	PS9L401LIA5	1	58	58	58
12	PS9L401LIA6	1	19	59	59
13	PS9L601LIA1	1	58	51	51
14	PS9L601LIA3	1	54	3	3
15	PS9L601LIA4	1	36	47	47

In LL simulations, the maximum number of neighbors, k_{max}, was not that sensitive since the algorithm checks all the neighbors up to k_{max}. As a representative sample, the histogram of optimum number of neighbors (k*) in test days of the LL algorithm for Station 12 is illustrated in Fig. 5. According to the figure, k_{max}=10 is a good selection for maximum number of neighbors since the k* is often 2, and occasionally it is some number in the range of 4 to 7.

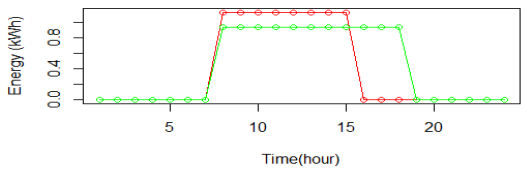

Figure 4. Actual enrgy consumption (green) and its prediction with kNN (red) for a sample test day in Station 2. The SMAPE for this day is 15.2%.

Historical average is the worst algorithm to implement. Average SMAPE on test days in all stations in the best case (station no .12) is 76.96% while in some other station (station no. 2) it is 97.62%. Therefore, even in the best case, the historical average is far worse than other algorithms. However, if someone insists on using the historical average as the prediction algorithm, it seems best to copy today's values for tomorrow's prediction, as for all stations the optimum depth is equal to 1.

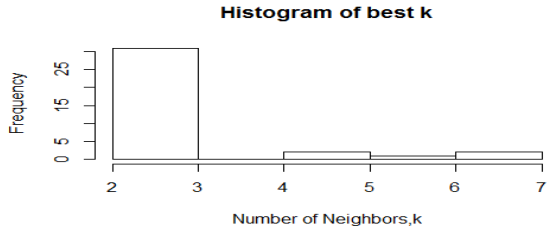

Figure 5. Histogram of selected number of neighbors, k*, in Lazy Learning algorithm for Station 12 data. All of the k*s are less than k_{max} = 10.

Comparing LL results with kNN, it seems that LL was not successful in choosing the best number of neighbors (k) for each query; otherwise, it would have better results compared to kNN (which has a constant k). Stations no. 8 and 12 are the exceptions to this, where LL does (slightly) better than kNN.

All simulations were run with RStudio Version 0.97.551 on an Intel Core i-7 CPU at 3.40 GHz with 16 GB RAM.

TABLE II. SELECTED NUMBER OF NEIGHBOURS PARAMETER FOR EACH ALGORITHM BASED ON CROSS VALIDATION RESULTS

No	Station	kNN	Weighted kNN
1	PS3L401LIA3	1	2
2	PS8L201LIA1	1	2
3	PS8L201LIA3	1	2
4	PS8L201LIA4	1	2
5	PS8L202LIIA1	1	2
6	PS8L202LIIA2	1	2
7	PS8L202LIIA3	1	2
8	PS9L401LIA1	1	2
9	PS9L401LIA2	1	2
10	PS9L401LIA3	1	2
11	PS9L401LIA5	1	2
12	PS9L401LIA6	1	2
13	PS9L601LIA1	1	2
14	PS9L601LIA3	1	2
15	PS9L601LIA4	1	2

TABLE III. NUMBER OF EFFECTIVE DAYS FOR EACH STATION

No	Effective Days
1	95
2	84
3	97
4	171
5	163
6	168
7	151
8	178
9	126
10	307
11	189
12	269
13	206
14	178
15	124

TABLE IV. AVERAGE AND STANDARD DEVIATION (IN PARANTHESES) OF SMAPE (%) ON TEST DAYS FOR EACH ALGORITHM

No	Historical Average		kNN		Weighted kNN		Lazy Learning	
1	95.98	(2.36)	5.38	(10.78)	6.52	(11.05)	6.20	(11.16)
2	97.62	(1.64)	14.71	(32.25)	16.89	(31.64)	17.28	(31.77)
3	97.45	(1.16)	49.48	(32.97)	49.54	(33.01)	49.60	(33.05)
4	94.10	(4.75)	0.60	(2.80)	10.04	(16.14)	10.34	(16.30)
5	87.04	(8.91)	28.22	(16.00)	35.15	(17.20)	37.23	(17.89)
6	83.19	(10.96)	31.26	(15.21)	37.45	(22.68)	36.00	(17.03)
7	87.07	(11.61)	27.89	(14.65)	34.21	(15.39)	34.63	(15.40)
8	80.44	(14.54)	29.82	(34.89)	30.04	(29.18)	24.93	(25.13)
9	95.95	(2.55)	12.19	(9.38)	20.76	(20.28)	21.51	(20.52)
10	92.82	(10.43)	8.51	(13.84)	12.25	(14.73)	12.45	(14.65)
11	87.71	(11.71)	7.74	(14.54)	14.78	(14.82)	16.47	(16.61)
12	76.69	(12.23)	17.23	(23.20)	3.75	(13.34)	3.75	(13.34)
13	91.99	(4.84)	14.43	(8.44)	16.62	(9.23)	17.60	(9.88)
14	90.94	(5.28)	9.68	(11.08)	16.69	(12.23)	16.73	(12.13)
15	87.49	(5.91)	7.89	(12.47)	9.73	(15.07)	10.38	(16.40)

V. ALGORITHM SELECTION FOR CELLPHONE APPLICATION

Based on the results analyzed in the previous section, kNN has higher accuracy on most of the stations; therefore, it was selected as the algorithm for the cellphone application. We implemented two algorithms on top of the prediction algorithm:

- One application takes the station name, required energy in kWh needed to charge the vehicle, and the start of charging time as input. The output is the predicted end time of charging,
- Another application takes the station name, starting time and ending time for the charging as input. The output is the predicted amount of available energy in kWh.

The total time for the algorithm to (a) run in C# under Microsoft Visual Studio 2012, (b) access the database through Microsoft SQL Server 2012, and (c) generate the output is about a second (less than a second for less crowded stations). This is well within the acceptable time for a mobile application response time. The algorithm is now running on the mobile application and is available to UCLA EV owners.

VI. CONCLUSION AND FUTURE WORK

In this paper, we analyze four relatively fast Machine Learning algorithms for prediction of energy consumption in EV charging stations. The speed of the algorithm is important since it should be in an acceptable range to be incorporated into a cell phone application. Out of these four algorithms, historical average, kNN, weighted kNN, and LL; Nearest Neighbor (kNN with k=1) was the most successful algorithm with the SMAPE measure. Two application based on this core algorithm were designed and are currently in use by EV owners at UCLA.

In the future work, we plan to investigate the effect of different forecasting strategies (such as multi-output vs. direct), and pre-processing stages to extract "features" (some statistics of the data vs. raw data).

REFERENCES

[1] M. Kintner-Meyer, K. Schneider, R. Pratt, "Impacts Assesment of Plug-in Hybrid Vehicles on Electric Utilities and Regional U.S. Power Grids Part 1: Technial Analysis", PNNL, *Jrnl. of EUEC*, Paper #04, Volume 1, 2007 [Online]. Avilable: http://www.euec.com/getattachment/euecjournal/Paper_4.pdf.aspx

[2] M. Majidpour, W.P. Chen, "Grid and Schedule Constrained Electric Vehicle Charging Algorithm Using Node Sensitivity Approach", *Proc. 2012 Intl. Conf. Connected Vehicles and Expo (ICCVE),* pp. 304-310.

[3] C. Chung, A. Shepelev, C. Qiu, C. Chu, R. Gadh, "Design of RFID Mesh Network for Electric Vehicle Smart Charging Infrastructure", *IEEE RFID TA 2013*, Johor Bahru, Malaysia, 4-5 September, 2013.

[4] S. Mal, A. Chattopadhyay, A. Yang, R. Gadh, "Electric vehicle smart charging and vehicle-to-grid operation", *Intl Journal of Parallel, Emergent and Distributed Systems*, vol. 27, no. 3. March 2012.

[5] C.Chung, E. Youn, J. Chynoweth, C. Qiu, C. Chu, R. Gadh, "Safety Design for Smart Electric Vehicle Charging with Current and Multiplexing Control", *2013 IEEE International Conference on Smart Grid Communications,* Vancouver, Canada, 21-24 October, 2013.

[6] C. Chung, J. Chynoweth, C. Qiu, C. Chu, R. Gadh, "Design of Fast Response Smart Electric Vehicle Charging Infrastructure", *IEEE Green Energy and Systems Conf.*, Long Beach, CA, Nov 25, 2013.

[7] Alternative Fuels Data Center, U.S. Department of Energy, [Online]. Available http://www.afdc.energy.gov/fuels/stations_counts.html

[8] N.K. Ahmed, A.F. Atiya, N. El Gayar, H. El-Shishiny, "An empirical comparison of machine learning models for time series forecasting", *Econometric Reviews*, 29 (2010), pp. 594–621.

[9] T. G. Dietterich, "Machine Learning for Sequential Data: A Review", *Springer Lecture Notes in Computer Science*, vol.2396, 2002, pp.15-30.

[10] GEP Box, GM Jenkins, GC Reinsel, *Time series analysis: forecasting and control*, Wiley Series in Probability and Statistics, 2013 (4th ed.)

[11] J. D. Hamilton, *Time Series Analysis*, Princeton University Press, 1994.

[12] K.N. Kumar, P.H. Cheah, B. Sivaneasan, P.L. So, D.Z.W. Wang, "Electric vehicle charging profile prediction for efficient energy management in buildings", in *Proc. 2012 IEEE Conference on Power & Energy,* pp. 480 – 485.

[13] A. Aabrandt , P. B. Andersen , A. B. Pedersen , S. You , B. Poulsen , N. O'Connell and J. Ostergaard "Prediction and optimization methods for electric vehicle charging schedules in the EDISON project", *IEEE Power Energy Soc. Innovative Smart Grid Tech. Conf.*, pp.1-7, 2012.

[14] R. Shankar, J. Marco, "Method for estimating the energy consumption of electric vehicles and plug-in hybrid electric vehicles under real-world driving conditions" *IEEE Intelligent Transport Systems*, vol. 7, pp. 138 – 150, March 2013

[15] A. Ashtari , E. Bibeau , S. Shahidinejad and T. Molinski "PEV charging profile prediction and analysis based on vehicle usage data", *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp.341 -350, 2012

[16] M. Alizadeh, A. Scaglione, J. Davies, K. S. Kurani, "A Scalable Stochastic Model for the Electricity Demand of Electric and Plug-In Hybrid Vehicles", *IEEE Trans. Smart Grid*, no. 99, pp.1-13, Sept 2013

[17] Y. Q. Li, Z. H. Jia, F. L. Wang, Y. Zhao, "Demand Forecast of Electric Vehicle Charging Stations Based on User Classification", *Applied Mechanics and Materials*, pp. 291-294, 2013

[18] J. Wang, K. H. Wu, F. Wang, Z. H. Li, Q. S. Niu, Z. Z. Liu, "Electric Vehicle Charging Station Load Forecasting and Impact of the Load Curve", 2012, *Applied Mechanics and Materials*, 229-231, 853

[19] G. Seymour, *Predictive Inference*. NY: Chapman and Hall, 1993

[20] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection". *Proc. of the Fourteenth International Joint Conference on Artificial Intelligence* 2 (12): pp.1137–1143, 1995.

[21] C. M. Bishop, N. M. Nasrabadi, *Pattern recognition and machine learning*. Vol. 1. New York: springer, 2006.

[22] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression". *The American Statistician* 46 (3): pp. 175–185, 1992.

[23] G. Bontempi, S. Ben Taieb, and Y. Le Borgne. "Machine Learning Strategies for Time Series Forecasting." In *Business Intelligence*, pp. 62-77. Springer Berlin Heidelberg, 2013.

[24] D. M. Allen, "The relationship between variable selection and data augmentation and a method for prediction." *Technometrics* 16, no. 1 (1974): 125-127.