Gated Ensemble Learning Method for Demand-Side Electricity Load Forecasting

Eric M. Burger, Scott J. Moura

Energy, Control, and Applications Lab, University of California, Berkeley

Abstract

The forecasting of building electricity demand is certain to play a vital role in the future power grid. Given the deployment of intermittent renewable energy sources and the ever increasing consumption of electricity, the generation of accurate building-level electricity demand forecasts will be valuable to both grid operators and building energy management systems. The literature is rich with forecasting models for individual buildings. However, an ongoing challenge is the development of a broadly applicable method for demand forecasting across geographic locations, seasons, and use-types. This paper addresses the need for a generalizable approach to electricity demand forecasting through the formulation of an ensemble learning method that performs model validation and selection in real time using a gating function. By learning from electricity demand data streams, the method requires little knowledge of energy end-use, making it well suited for real deployments. While the ensemble method is capable of incorporating complex forecasters, such as Artificial Neural Networks or Seasonal Autoregressive Integrated Moving Average models, this work will focus on employing simpler models, such as Ordinary Least Squares and k-Nearest Neighbors. By applying our method to 32 building electricity demand data sets (8 commercial and 24 residential), we generate electricity demand forecasts with a mean absolute percent error of 7.5% and 55.8% for commercial and residential buildings, respectively.

Keywords: Smart Grid, Buildings, Electricity Forecasting, Ensemble Learning

1. INTRODUCTION

Commercial and residential buildings account for 74.1% of U.S. electricity consumption, more than either the transportation sector or the industrial sector (0.2% and 25.7%, respectively) [1]. Maintaining a continuous and instantaneous balance between generation and load is a fundamental requirement of the electric power system [2]. To reliably match supply with demand, the forecasting of grid-level electricity loads has long been a central part of the planning and management of electrical utilities [3]. The accuracy of these forecasts has a strong impact on the reliability and cost of power system operations. Trends, such as vehicle electrification and distributed renewable generation, are expected to pose new challenges for grid operators and may undermine the accuracy of load forecasts.

To improve the accuracy of electricity demand forecasts and aid in the management of power systems, recent attention has been placed on short-term building-level electricity demand forecasting using a wide range of models [4][5]. The ability to accurately and adaptively forecast demand-side loads will play a critical role in maintaining grid stability and enabling renewables integration. Additionally, many novel optimal control schemes, under research umbrellas such as demand response and microgrid management, require short-term building electricity demand forecasts to aid in decision making [6].

The supply-side and load-side time series forecasting of electricity demand has been a topic of research for many decades. The literature is filled

*Corresponding Author: Eric M. Burger,
Email: ericburger@berkeley.edu, Phone: 317-385-1768,
**Affiliation Address: Energy, Control, and Applications Lab (eCAL), 611 Davis Hall, Department of Civil and Environmental Engineering, University of California, Berkeley, Berkeley, CA 94720, USA.

with a variety of well-cited modelling approaches, each differing in algorithmic complexity, estimation procedure, and computational cost. Of particular note are the variants of Artificial Neural Networks (ANN) [3][4][5][7][8][9][10], Support Vector Regression (SVR) [11][12][13][14] and Autoregressive Integrated Moving Average (ARIMA) models [3][12][13][15][16][17][18]. Lesser but nonetheless noteworthy attention has been given to approaches such as Multiple Linear Regression [3][11][19], Fuzzy Logic [3][20], Decision Trees [4], and k-Nearest Neighbors (k-NN).

These studies provide a broad catalog of use-cases and demonstrate the performance of certain forecasting algorithms when applied to specific building types. In particular, [3][4][10][17] provide a survey of electricity forecasting methods and a high-level comparison of techniques. [8] provides a detailed description of ANNs and their application to load forecasting, including data pre-processing and ANN architectures. [5] details the development of a seasonal ANN approach and the advantage over a Seasonal ARIMA (SARIMA) model when applied to 6 building datasets. [18] focuses on the introduction of motion sensor data to improve the accuracy of an ARIMA model. In [9][11][15][18][20], the authors perform an in-depth analysis of the power demand patterns of a particular building in order to customize a forecasting model.

In papers with experimental results, the authors have generally applied their electricity demand forecasting technique to only a small number of datasets. Consequently, the literature is rich with forecasting algorithms customized for individual buildings. This leads us to the following question: Is it possible to design a single minimally-customized forecasting algorithm that is widely applicable across a diversity of building types, enabling scalability? We pursue this question by proposing a novel ensemble learning method for electricity demand forecasting.

Specifically, due to unique building characteristics, occupancy patterns, and individual energy use behaviors, we argue that no single model structure is capable of accurately forecasting electricity demand across all commercial and residential buildings.

For example, some forecasting models may produce accurate predictions under certain observable or unobservable conditions, such as a seasonal trend, a morning routine, or an extended absence. Other models may be ideal for buildings with energy use behaviors that are stable over long periods of time. For buildings with frequent changes in occupancy patterns, models that are trained over a moving horizon may yield the highest accuracy. In short, this work will develop an ensemble learning method that trains and validates multiple forecasting models before applying a gating method to select a single model to perform electricity demand forecasting.

In this way, the ensemble method is able to learn from real-time data and to produce short-term electricity demand forecasts that are automatically tailored to a particular building and instance in time. In addition to forecast accuracy, this paper will place an emphasis on method adaptability and ease of use. While we have implemented certain forecasting models, the method is intended to allow the models to be interchangeable.

To demonstrate the use of our ensemble method to produce short-term forecasts, this paper includes 3 experimental studies: Single Model Studies, Multiple Model Study, and Residential Study. For each of these studies, we will make the following assumptions with respect to the availability of building electricity demand data:

A1. We have access to hourly historical building electricity demand at the meter.

A2. We have access to hourly historical weather data near the building location.

A3. We do not have access to submetered electricity demand data or building operations data, such a occupancy measurements or mechanical system schedules.

The limited access to input data with which to produce forecasts is representative of the challenge faced by grid operators. Accordingly, this paper will demonstrate the potential of our ensemble method to non-invasively forecast total electricity demand using data-driven methods. Additionally, unlike in [9][11][15][18][20], where the authors perform an in-depth analysis of the power demand patterns in order to customize a model to a particular building, this paper will focus on developing a forecasting approach that is generally applicable to all buildings without customization.

This paper is organized into five sections: Regression Models, Single Model Studies, Ensemble Method, Multiple Model Study, and Residential Study. Section II. Regression Models briefly

presents background theory for 5 regression models that will be employed in this paper. In Section III. Single Model Studies, we apply the forecasting models to 8 commercial/university building electricity demand datasets using batch and moving horizon training approaches. Section IV. Ensemble Method presents our method for training and validating multiple models and for selecting the optimal model using a gating method. Section V. Multiple Model Study applies our ensemble learning method to 8 commercial/university building electricity demand datasets and quantifies and qualifies the advantage over a single model approach. Finally, in Section VI. Residential Study, we apply our ensemble learning method to 24 residential building electricity demand datasets and summarize the results. Key conclusions and future research directions are summarized in Section VII.

2. Regression Models

In this paper, we will consider one parametric regression model, Ordinary (Linear) Least Squares with ℓ_2 Regularization (Ridge), and four nonparametric models, Support Vector Regression with Radial Basis Function (SVR), Decision Tree Regression (DTree), k-Nearest Neighbors with uniform weights and binary tree data structure (k-NN), and Multilayer Perceptron (MLP), a popular type of feedforward Artificial Neural Network (ANN). In this section, we will briefly describe the structure of each regression model.

2.1. Ordinary Least Squares with ℓ_2 Regularization

Ordinary Least Squares with ℓ_2 Regularization (Ridge) fits a linear model with coefficients $w \in \mathbf{R}^n$ to minimize the residual sum of squared errors between the observed and predicted responses while imposing a penalty on the size of coefficients according to their ℓ_2-norm. The linear model of a system with univariate output is given by

$$\begin{aligned}\hat{y} &= w_0 x_0 + w_1 x_1 + \ldots + w_n x_n \\ &= \sum_k w_k x_k = w^T x\end{aligned} \tag{1}$$

with variables $x \in \mathbf{R}^n$, the model input, $\hat{y} \in \mathbf{R}$, the predicted response, n, the number of inputs or features in x, and $k = 1, \ldots, n$.

The linear model is trained on a set of inputs and observed responses by optimizing the function

$$\underset{w}{\text{minimize}} \sum_i \|w^T x_i - y_i\|_2^2 + \lambda \|w\|_2^2 \tag{2}$$

with variables $x_i \in \mathbf{R}^n$, the model input for the i-th data point, $y_i \in \mathbf{R}$, the i-th observed response, $w \in \mathbf{R}^n$, the weighting coefficients, and $i = 1, \ldots, N$, where N is the number of data samples and n is the number of features in x_i. Lastly, λ is a weighting term for the regularization penalty.

For a system with a multivariate output $\hat{y} \in \mathbf{R}^m$, we will treat the outputs as uncorrelated and define a set of coefficients $w_j \in \mathbf{R}^n$ for each predicted response $\hat{y}_j \in \mathbf{R}$ for $j = 1, \ldots, m$. Thus, the multivariate linear model is given by

$$\hat{y}_j = w_j^T x, \ \forall j = 1, \ldots, m \tag{3}$$

The weights of the multivariate model are determined by optimizing the function:

$$\underset{w}{\text{minimize}} \sum_i \sum_j \|w_j^T x_i - y_{i,j}\|_2^2 + \sum_j \lambda \|w_j\|_2^2 \tag{4}$$

with variables $x_i \in \mathbf{R}^n$, the model input, $y_i \in \mathbf{R}^m$, the observed multivariate response, $w_j \in \mathbf{R}^n$, the weighting coefficients of the j-th response, $i = 1, \ldots, N$, and $j = 1, \ldots, m$, where N is the number of data samples, n is the number of features in x_i, and m is the number of observations in y_i.

2.2. Support Vector Regression

In non-linear Support Vector Regression (SVR), the model input x is transformed into a higher dimensional feature space using a mapping function ϕ. Then, a linear model is constructed in this feature space, as given by

$$\hat{y} = f(x) = w^T \phi(x) + b \tag{5}$$

where variable $b \in \mathbf{R}$ is a bias term, $w \in \mathbf{R}^n$ the linear coefficients, and ϕ a non-linear mapping function. Using an ϵ-intensive loss function, the support vector regression model can be trained by optimizing

$$\begin{aligned}\underset{w,b,\zeta,\zeta*}{\text{minimize}} \ & \frac{1}{2}\|w\|_2^2 + C\sum_i (\zeta_i + \zeta_i^*) \\ \text{subject to } & y_i - w^T\phi(x_i) - b \leq \epsilon + \zeta_i \\ & w^T\phi(x_i) + b - y_i \leq \epsilon + \zeta_i^* \\ & \zeta_i, \zeta_i^* \geq 0\end{aligned} \tag{6}$$

where constant $\epsilon \in \mathbf{R}$ denotes the radius of the ϵ-intensive region (i.e. region in which the value of the loss function is 0) and constant $C > 0$ denotes the trade-off between the empirical risk (i.e. deviations beyond ϵ) and the regularization term (i.e. the flatness of the model). The positive slack variables $\zeta_i \in \mathbf{R}$ and $\zeta_i^* \in \mathbf{R}$ denote the magnitude of the deviation from the ϵ-intensive region.

By applying Lagrangian multipliers and Karush-Kuhn-Tucker conditions, the primal problem can be reformulated into a dual form that is easier to solve.

$$\begin{aligned} \underset{\alpha,\alpha*}{\text{minimize}} \; & \frac{1}{2}\sum_i\sum_j(\alpha_i - \alpha_i^*)^T K(x_i, x_j)(\alpha_i - \alpha_j^*) \\ & + \epsilon\sum_i(\alpha_i + \alpha_i^*) - \sum_i y_i(\alpha_i - \alpha_i^*) \\ \text{subject to } & \sum_i(\alpha_i - \alpha_i^*) = 0 \\ & 0 \leq \alpha_i, \alpha_i^* \leq C \end{aligned} \tag{7}$$

with variables $\alpha_i, \alpha_i^* \in [0, C]$, the Lagrangian multipliers, and K, the kernel function given by the inner product of the mapping functions, $K(x_i, x_j) = \phi(x_i)\phi(x_j)$. In this paper, we will employ the (Gaussian) Radial Basis Function kernel, given by

$$K(x_i, x) = \exp\left(-\frac{\|x_i - x\|_2^2}{2\sigma^2}\right) \tag{8}$$

where variable $\sigma \in \mathbf{R}$ is a free parameter.

Using the Lagrangian multipliers, the support vector regression model with univariate output can be rewritten as

$$\hat{y} = f(x) = \sum_i(\alpha_i - \alpha_i^*)K(x_i, x) + b \tag{9}$$

For a system with a multivariate output $\hat{y} \in \mathbf{R}^m$, we will treat the outputs as uncorrelated and define m support vector regression models (i.e. one for each output). Using m models to independently predict each of the m outputs is simple to implement, but may be less accurate than a single model capable of simultaneously predicting all m outputs. We refer the reader to [21][22] for a more detailed description of the support vector regression algorithm.

2.3. Decision Tree Regression

In Decision Tree Regression (DTree), the feature space is recursively sub-divided or partitioned into smaller regions or leaves (i.e. terminal node). Once this splitting is complete, a simple regression model is fit to the data samples that have been grouped into each leaf. The objective is to form a tree data structure with which a new observation can be assigned to a leaf using nested boolean logic (i.e. moving right or left down each branch according to a threshold value). Once the correct leaf has been identified, the observation can be mapped to a continuous target value using a simple model.

Expressed mathematically, we can represent the data at node a by B. For each potential division $\theta = (k, t_a)$ composed of feature k and threshold value $t_a \in \mathbf{R}$, we may partition the data into two subsets or branches, $B_L(\theta)$ and $B_R(\theta)$, given by

$$\begin{aligned} B_L(\theta) &= (x, y) \text{ if } x_k \leq t_a \\ B_R(\theta) &= (x, y) \text{ if } x_k > t_a \end{aligned} \tag{10}$$

The accuracy of a decision tree regression model can be represented by the impurity of each branch. In this paper, we will define the impurity function H as the mean squared error between each response and the mean response in a branch. Thus, for node a, H is given by

$$\begin{aligned} \bar{y}_a &= \frac{1}{N_a}\sum_{i=1}^{N_a} y_{a,i} \\ H(X_a) &= \frac{1}{N_a}\sum_{i=1}^{N_a}(y_{a,i} - \bar{y}_a)^2 \end{aligned} \tag{11}$$

with variable $N_a \in [N_{min}, N]$, the number of data points in branch a, N_{min}, the minimum number of data points in a branch, X_a, the set of all data points $x \in \mathbf{R}^n$ in branch a, and y_a, the set of all observed responses $y \in \mathbf{R}$ in branch a.

The decision tree is trained or grown by recursively selecting the parameters that minimize the impurity of the tree and of each branch. In other words, we begin with a node a containing all data points. Then, we partition the data according to the optimization function

$$\theta^* = \underset{\theta}{\text{argmin}}\frac{N_L}{N_a}H(B_L(\theta)) + \frac{N_R}{N_a}H(B_R(\theta)) \tag{12}$$

with variable $N_L, N_R \in [N_{min}, N_a]$, the number of data points in the left and right branches, respectively. The optimization function is recursively applied to each new branch until the maximum tree depth is reached, one of the resulting branches

would contain less than N_{min} data points, or there is no split that will decrease the impurity of the branch by more than some threshold $\delta \in \mathbf{R}$.

For each leaf a, we will define the regression model as the mean of the contained univariate observations.

$$\hat{y} = \frac{1}{N_a} \sum_{i=1}^{N_a} y_{a,i} \tag{13}$$

For a system with multivariate output $\hat{y} \in \mathbf{R}^m$, each leaf in the tree will store observations of length m rather than 1. Therefore, the impurity function H can be redefined as the mean of the mean squared error between each j-th response and the mean j-th response in a branch for $j = 1, \ldots, m$.

$$\begin{aligned} \bar{y}_{a,j} &= \frac{1}{N_a} \sum_{i=1}^{N_a} y_{a,i,j} \ \forall j = 1, \ldots, m \\ H(X_a) &= \frac{1}{mN_a} \sum_{j=1}^{m} \sum_{i=1}^{N_a} (y_{a,i,j} - \bar{y}_{a,j})^2 \end{aligned} \tag{14}$$

And the multivariate regression model can be defined as

$$\hat{y}_j = \frac{1}{N_a} \sum_{i=1}^{N_a} y_{a,i,j} \ \forall j = 1, \ldots, m \tag{15}$$

We refer the reader to [23][4] for a more detailed description of the decision tree regression algorithm.

2.4. k-Nearest Neighbors Regression

In k-Nearest Neighbors Regression (k-NN), an input $x \in \mathbf{R}^n$ is mapped to a continuous output value according to the weighted mean of the k nearest data points or neighbors, as defined by the Euclidean distance. In this paper, we will use uniform weights. In other words, each point in a neighborhood a contributes uniformly and thus the predicted univariate response $\hat{y} \in \mathbf{R}$ is the mean of the k-nearest neighbors.

$$\hat{y} = \frac{1}{k} \sum_{i=1}^{k} y_{a,i} \tag{16}$$

with variable y_a, the set of k observed responses $y \in \mathbf{R}$ in neighborhood a. For a system with multivariate output $\hat{y} \in \mathbf{R}^m$, the model is defined as the mean of each observation j over the k-nearest neighbors.

$$\hat{y}_j = \frac{1}{k} \sum_{i=1}^{k} y_{a,i,j} \ \forall j = 1, \ldots, m \tag{17}$$

Given a new input x, it is possible to determine the neighborhood by computing the Euclidean distance (i.e. ℓ_2-norm of the difference) between the new input x and every data point in the training data set x_i for $i = 1, \ldots, N$ and then ordering the distances to identify the nearest neighbors. However, this brute-force search is computationally inefficient for large datasets.

To improve the efficiency of the neighborhood identification, the training data points are partitioned into a tree data structure. A commonly used approach for organizing points in a multidimensional space is the ball tree data structure, a binary tree in which every node defines a D-dimensional hypersphere or ball. At each node, data points are assigned to the left or right balls according to their distance from the ball's center. At each terminal node or leaf, the data points are enumerated inside the ball.

We refer the reader to [24] for a description of ball tree construction algorithms.

2.5. Multilayer Perceptron Artificial Neural Network

Artificial Neural Networks (ANN) are a class of statistical learning algorithms inspired by the neurophysiology of the human brain. These numerical models are composed of interconnected "neurons" which use stimulation thresholds to predict how a system will respond to inputs.

The most popular feedforward (i.e no feedback) neural network is the multilayer perceptron (MLP). Figure 1 illustrates an example of a 3 layer perceptron network consisting of a 3 neuron input layer, 4 neuron hidden layer, and 2 neuron output layer. The structure of a neuron in the hidden layer is presented in Figure 2 where variables $x_1, x_2, x_3 \in \mathbf{R}$ are inputs to the neuron and $w_1, w_2, w_3 \in \mathbf{R}$ are synaptic weights. Variable $y \in \{0, 1\}$ is the output and $z_1, z_2 \in \mathbf{R}$ are the synaptic weights of the next neurons in the network. The weighted sum of the inputs is the excitation level of the neuron

$$v = \sum_{i=1}^{n} w_i x_i - h \tag{18}$$

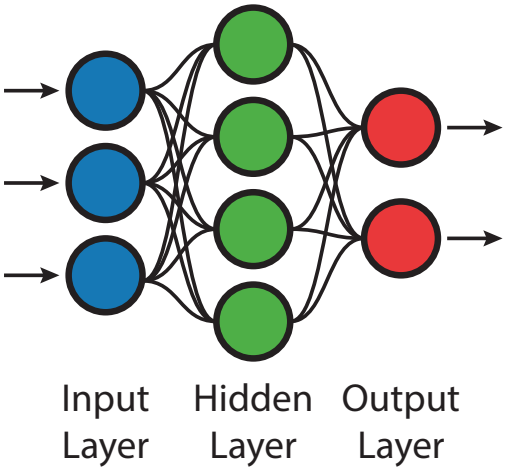

Figure 1: Artificial Neural Network

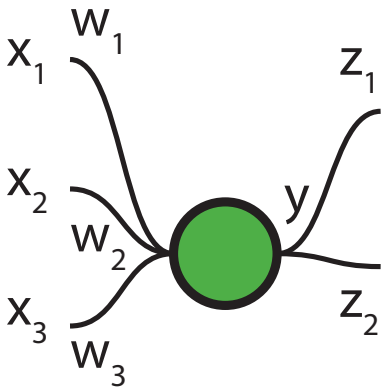

Figure 2: Structure of a Neuron

where variable $v \in \mathbf{R}$ is the excitation, $h \in \mathbf{R}$ is a threshold, and n is the number of inputs to the neuron. Next, we want to define the output or activation function f of the neuron such that if $v \geq 0$ then $y = 1$ otherwise $y = 0$. The simplest activation function is the hard limiter,

$$y = f(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases} \tag{19}$$

However, the hard limiter function cannot be practically implemented because it is not differentiable. Thus, artificial neural network algorithms employ differentiable activation functions that have horizontal asymptotes at both 0 and 1 (i.e. $\lim_{v\to\infty} f(v) = 1$ and $\lim_{v\to-\infty} f(v) = 0$). An example is the sigmoid function,

$$y = f(v) = \frac{1}{1 + e^{-v}} \tag{20}$$

For the neurons in the output layer, it is common to use a linear activation function,

$$\hat{y} = f(v) = v \tag{21}$$

The structure of the multilayer perceptron artificial neural network makes it capable of both univariate and multivariate predictions. Networks are trained using a backpropogation algorithm which adjusts the weights and thresholds of each neuron to minimize the error between the observations and the network outputs. In this paper, we will employ a 3 layer feedforward ANN with a 30 neuron sigmoid hidden layer and 6 neuron linear output layer. The size of the linear input layer will vary. The ANN will be trained using gradient descent backpropagation for 200 epochs.

We refer the reader to [25] for a further discussion of artificial neural networks and backpropogation training algorithms.

2.6. Software Packages

This work employs the PyBrain Artificial Neural Network library [26] and the Sci-Kit Learn Ordinary Least Squares, Support Vector Regression, Decision Tree, and k-Nearest Neighbors libraries [27]. All plots are generated with Matplotlib [28].

3. SINGLE MODEL STUDIES

To motivate the advantage of our ensemble approach, we will begin by considering single model approaches to electricity demand forecasting. In this section, we will apply the regression models above to 8 building datasets containing 2 years of metered hourly electricity demand (kW). This time-series data has been provided by the facilities team at the University of California, Berkeley and will be used as the observation data for each forecasting model. Submetered electricity demand data and building operations data, such a occupancy measurements and mechanical system schedules, are not available.

The 8 buildings are located on the University of California, Berkeley campus and have been selected to represent a heterogeneous population. The buildings A, B, and D are occupied by the physics, civil engineering, and environmental engineering departments, respectively, and are primarily comprised of faculty offices and research laboratories. Building C is a university library and building E houses faculty and departmental offices

for multiple humanities departments. Buildings F and G are university administrative buildings and building H is comprised mainly of lecture halls and classrooms. Table 1 presents the square footage as well as basic statistics regarding the electricity demand of each building.

	Building							
	A	B	C	D	E	F	G	H
Size (10^3 sq.ft.)	97	140	67	142	306	111	153	140
Mean (kW)	333	33	96	109	113	114	23	73
SD (kW)	44	5	26	8	36	33	2	30
Min (kW)	190	20	48	61	60	69	5	29
Max (kW)	602	69	221	150	271	236	73	195

Table 1: Building Electricity Demand Statistics

The models will be used to generate short-term multivariate electricity demand forecasts, specifically 6 consecutive hourly electricity demand predictions (i.e. $\hat{y} \in \mathbf{R}^6$). The accuracy of each forecast $\hat{y}$ will be measured by the root mean squared error (RMSE). To allow for comparison between buildings, the performance of forecasting models will be measured by the mean absolute percent error (MAPE).

$$\text{Forecast } i \text{ RMSE} = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(y_{i,j} - \hat{y}_{i,j})^2} \quad (22)$$

$$\text{Model MAPE} = \frac{100\%}{mN}\sum_{i=1}^{N}\sum_{j=1}^{m}\left|\frac{y_{i,j} - \hat{y}_{i,j}}{y_{i,j}}\right| \quad (23)$$

with variables $y_i \in \mathbf{R}^m$, the i-th observation, and $\hat{y}_i \in \mathbf{R}^m$, the i-th prediction, where m represents the number of outputs in the prediction ($m = 6$) and N, the number of predictions.

The regression models will employ 4 different input types: electricity demand (D), time (T), electricity demand and time (DT), and electricity demand, time, and exogenous weather data (DTE). The electricity demand input type (D) consists of the 24 hourly records that precede the desired forecast ($x \in \mathbf{R}^{24}$). The time input type (T) is the current weekday and hour represented as a sparse binary vector ($x \in \{0,1\}^{31}$). The demand and time input type (DT) combines the demand and time inputs ($x \in \mathbf{R}^{55}$). The demand, time, and exogenous weather data input type (DTE) is the demand and time input with current outdoor air temperature (°C) and relative humidity (%RH) data retrieved from a local weather station ($x \in \mathbf{R}^{57}$)[29]. The output of each forecasting model is a prediction of the hourly electricity demand over the following 6 hours ($y \in \mathbf{R}^6$).

Throughout this paper, we will use the term "regression model" to refer to the model structure and algorithm used to perform regression, "input type" to refer to the subset of features used by each model, and "forecasting model" to refer to the pairing of regression model and input type.

3.1. Batch Study

For each of the 8 buildings, we train the forecasting models with demand data from that building. Electricity demand data from one building is not used to fit the models of another building. We consider 5 regression models (Ridge, SVR, DTree, k-NN, and ANN) and 4 input types (D, T, DT, and DTE) for a total of 20 forecasting models per building. The forecasting models for each building are trained in a batch manner (i.e. trained once on a large dataset) using 18 months of hourly input data from January 1st, 2012, to July 1st, 2013 (i.e. 13,128 training data points). For each model, the training dataset depends on the input type (D, T, DT, and DTE) but may include hourly electricity demand records for the respective building (D), time records represented as a sparse binary vector (T), and hourly outdoor air temperature and relative humidity records (E).

For validation, the trained models are used to generate a 6 hour electricity demand forecast ($\hat{y} \in \mathbf{R}^6$) for each building for every hour from July 1st, 2013, to January 1st, 2014 (i.e. 4,416 testing data points). The results of the batch regression study are presented in Figure 3. In the figure, each data point represents the MAPE of one forecasting model (i.e. 8 buildings with 20 models each for a total of 160 models).

Examples of the electricity demand forecasts produced by the Building E Ridge regression model with demand (D) input are presented in Figure 4. Note that each blue line represents a multivariate forecast $\hat{y}$ and that the figure plots $\hat{y}$ starting at but excluding the most recent power demand record.

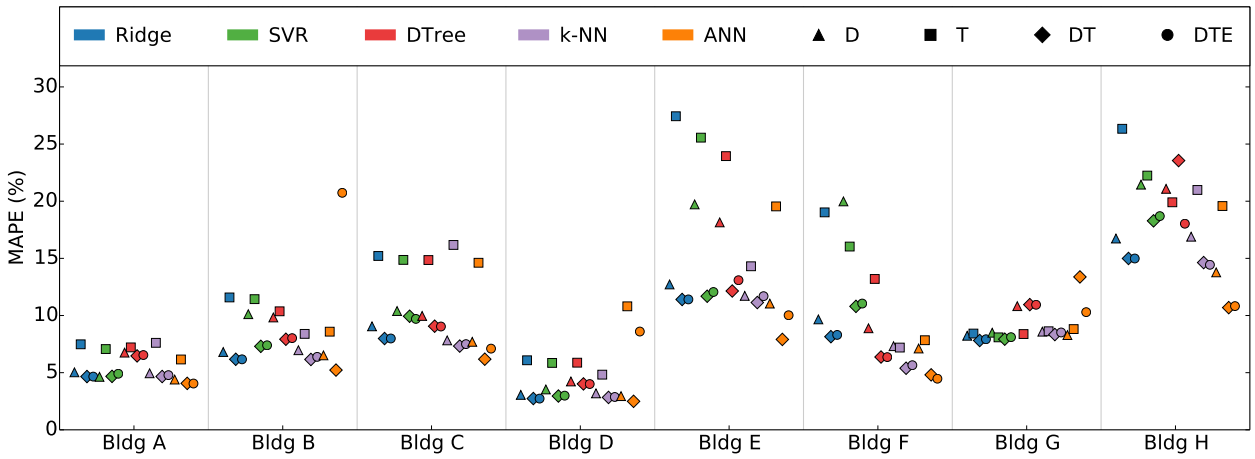

Figure 3: Batch Study Results

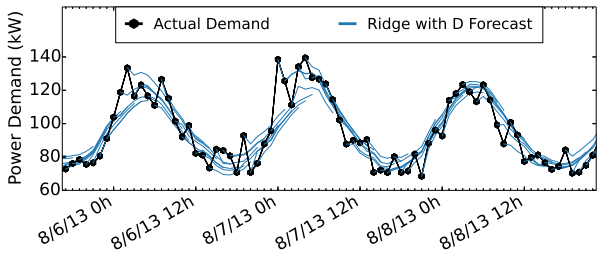

Figure 4: Building E Ridge Forecast Sample

By comparing the results in Figure 3 for each building, we can immediately distinguish forecasters that consistently perform poorly (e.g. T input) from forecasters that perform well (e.g. ANN, Ridge, and k-NN with DT input). We also observe that certain forecasting models perform inconsistently across the different buildings. For example, SVR with D input performs well in Building A but relatively poorly in Buildings E and F. Furthermore, there is dispersion among the results, particularly in Buildings E, F, and H. This dispersion represents a challenge for building level applications. To produce the best results using a batch approach, an engineer must perform model selection for every deployment. Just because a certain regression model and input type has performed well for one building does not guarantee it will do the same for another building.

As shown, an ANN model outperforms the Ridge, SVR, DTree, and k-NN models in 7 out of 8 building. However, there is no input type that performs best in each building. In Building B, we observe that the addition of the exogenous weather input decreases the performance of the ANN. A possible explanation is that the ANN found a link between the weather input and the electricity demand in the training data. However, this link may not have continued to appear in the test data, resulting in a loss in performance.

Additionally, it could be argued that the ANN model does not outperform the much simpler Ridge and k-NN models to an extent that warrants the additional complexity, particularly in Buildings D and G. If further tuned to a particular building and input type and trained over a higher number of epochs, it is possible that the ANNs' performances could be further improved. However, this increase in forecaster accuracy would come at a high computational cost. Instead, this work will focus on developing a computationally efficient ensemble method for producing electricity demand forecasts by learning from data streams and adapting to individual building energy use patterns.

3.2. Moving Horizon Study

In the batch regression study presented above, each forecasting model was trained once on 18 months of data and then used to generate predictions up to 6 months after the last training data point. For buildings with very consistent electricity demand patterns, using such a large training set will help to prevent overfitting and to produce the best possible results. For buildings with inconsistent or changing electricity demand patterns, the absence of the most recent data from the training set may limit the performance of the forecasting model.

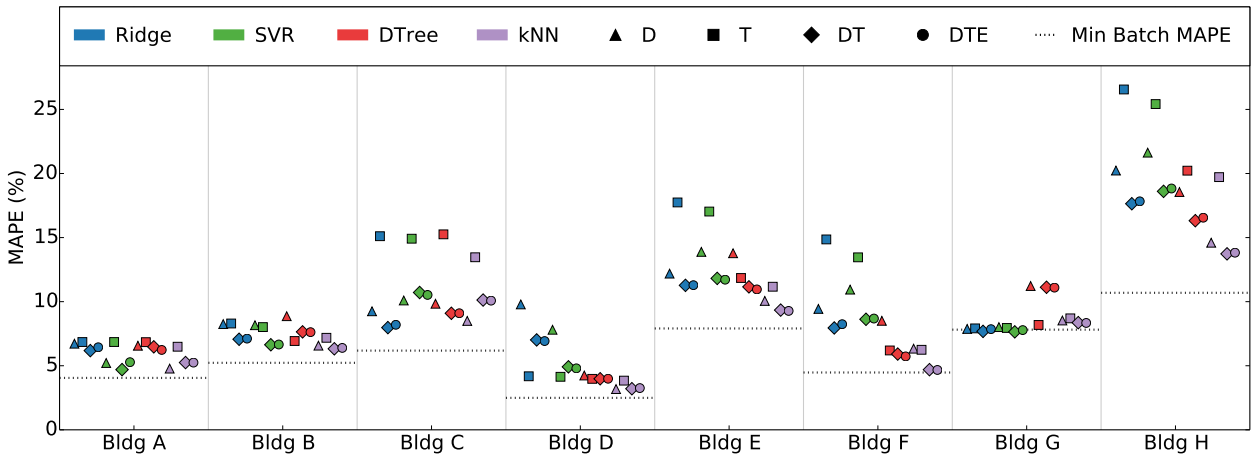

Figure 5: Moving Horizon Study Results

In this section, we will consider training the models over a moving horizon. Specifically, at each hour between July 1st, 2013, and January 1st, 2014, we will train the forecasting models for each building on the 3 months of data that precede that time step (i.e. 2,016 training data points). Then, we will produce a 6 hour forecast ($\hat{y} \in \mathbf{R}^6$) and record the errors. In this way, the start and end times of the training set will move relative to the current time step and we can hope to better capture recent electricity demand patterns. It should be noted that because we have significantly reduced the training set size, we have introduced the potential for overfitting the models, a point that will be addressed by our ensemble method. We will consider 4 regression models (Ridge, SVR, DTree, and k-NN) and 4 input types (D, T, DT, and DTE) for a total of 16 forecasting models per building. The computational cost of training an ANN makes the model unsuitable for a moving horizon approach.

The results of the moving horizon regression study are presented in Figure 5. In the figure, each data point represents the MAPE of one forecasting model (i.e. 8 buildings with 16 models each for a total of 128 models). The horizontal dotted line marks, for each building, the highest performing forecasting model (lowest MAPE) from the batch regression study. In other words, for Buildings B, C, D, E, and H, the dotted line indicates the MAPE of the ANN model with DT input. For Buildings A and F, the line marks the MAPE of the ANN model with DTE input. For Building G, Ridge with DT produced the lowest MAPE in the batch study.

While none of the models trained on a moving horizon show a significant improvement in accuracy over the best batch model, the results for each individual building are generally less dispersed in the moving horizon cases than in the batch cases.

Because we have generated 16 forecasts for every hour between July 1st, 2013, and January 1st, 2014, we are able to compare the performance of the forecasting models at each time step and across all 8 buildings. This analysis will aid in determining which models to include in the ensemble method. Figure 6 shows the fraction of time steps that a specific regression model produced the most accurate (lowest RMSE) electricity demand prediction (regardless of input type). Here, we see that k-Nearest Neighbors models produce the best forecast with the highest frequency of any of the regression models considered. Because we plan to incorporate several forecasters in the ensemble method, we are also interested in identifying models that consistently rank among the top. Figure 7 shows the fraction of time steps that a specific regression model produced a forecast that was among the best four predictions. Here, we see that Ridge models most consistently produced such a prediction. Repeating this analysis for input types (not displayed), we find that every input scores between 20% and 30% for both the top and top four predictions, suggesting no clear relative advantage.

Recognizing that a specific forecasting model may have the highest accuracy in one time step and the lowest accuracy in the next, we are also interested in which regression models and inputs show the poorest performance (highest RMSE). For the model that generates the worst forecast at each

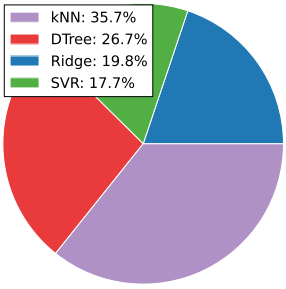

Figure 6: Best Prediction

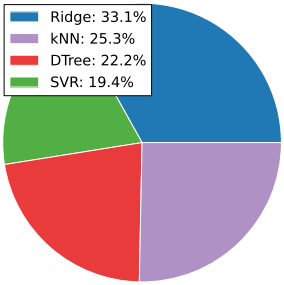

Figure 7: Best Four Predictions

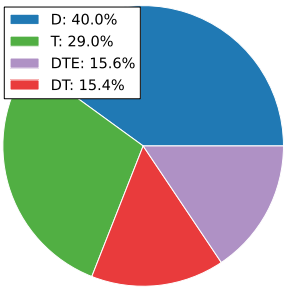

Figure 8: Worst Prediction

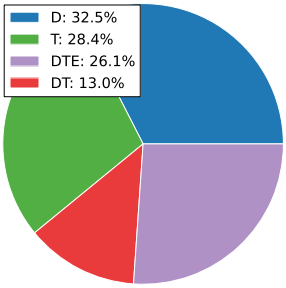

Figure 9: Worst Four Predictions

time step (not displayed), Decision Tree Regression ranks poorest (44%) followed by Ridge (31%). For the worst four predictions, Decision Tree Regression ranks poorest again (42%) followed again by Ridge (26%). Figures 8 and 9 show the fraction of time steps that a specific input type produced the least accurate electricity demand prediction. Not surprisingly, the demand only input (D) ranks worst in both cases. For the buildings studied, exogenous weather inputs do not appear to significantly improve the forecast accuracy. Buildings located in less temperate climates could be expected to show greater correlation between temperature and electricity demand.

3.3. Single Model Study Conclusions

Based on the results from the batch and moving horizon studies, we assert that no single forecasting model (regression model and input type) will produce the best results across every building. To produce good results using a single model approach, an engineer must perform model and feature selection for each deployment, increasing the cost of energy management applications that require building-level electricity demand forecasts.

However, the results also show that there is a subset of forecasting models that perform well for each building. Furthermore, at each time step, one forecasting model produces a prediction that is more accurate than any other prediction. If we train a set of regression models and determine which model in the set is most likely to produce the best prediction at a given time step, we can formulate an ensemble method that is able to perform model and feature selection by learning from past electricity demand. This is the core motivation of the gated ensemble learning method presented in the next section.

4. ENSEMBLE METHOD

4.1. Background

Given the many unpredictable behaviors of occupants and the unique physical and mechanical characteristics of every building, a single model approach to electricity demand forecasting may perform very well in one case and very poorly in another. Without being able to observe the causes of electricity demand changes (through extensive sub-metering and/or occupancy sensing), it is difficult to justify why a model does or does not perform well. Furthermore, the incorporation of exogenous signals like regional weather conditions may improve a model's accuracy but such benefits cannot be guaranteed. Only through observation and experimentation can the best regression model and input type be identified for a particular building.

The assertion that the best forecasting model can be identified through data driven experimentation underlies the ensemble method presented in this paper. To build upon existing literature and to improve the portability of electricity demand forecasters, we have developed a method that tests multiple models before selecting one that is best suited for a particular building and instance in time.

Our multiple model regression method falls under a category of ensemble learning methods commonly referred to as a "bucket of models" [30][31]. It is important to recognize that unlike other ensemble methods (which average, stack, or otherwise combine the outputs from multiple models), the bucket of models approach selects a single model from the ensemble set. Consequently, a bucket of models approach can perform no better than the best model in the set. Therefore, to produce accurate forecasts, it is important that the individual models perform well, though perhaps only in certain conditions. Additionally, our ensemble method must

be able to identify and avoid models that are likely to perform poorly for a particular building. This capability will alleviate the need for prior knowledge of a building's energy use and, in practice, allow for model and feature selection to be performed in real time.

4.2. Method

Our gated ensemble learning method can be divided into 4 steps: Training, Validation, Gating, and Testing. In the training step, each model is trained on a subset of historic data, with the most recent electricity demand data points reserved for validation, as shown in Figure 10. The size of the training subset may vary by application and by training approach. For models trained in a batch manner, a large data set (e.g. >12 months) is required. Additionally, the training step is either performed only once or periodically (e.g. every 6 months) rather than at every time step.

For models trained in a moving horizon manner, a smaller dataset is required (e.g. 2 to 12 months) and the training step is performed periodically (e.g. daily) or at every time step. Again, the length of the dataset can be customized to the application. For example, because the use patterns of university buildings change according to an academic calendar, a training set size of 2 or 3 months may be ideal. For an office building with very consistent energy use patterns, a training set size of 6 months or more may produce the best results. Stated more explicitly, small training sets are more capable of capturing recent energy use patterns but carry the risk of allowing the regression model to overfit the data. By contrast, a large dataset will capture consistent energy use patterns but may miss recent or short-term changes, thereby appearing to underfit the data. In this paper, we will favor smaller training sets (3 months) for moving horizon models and larger trainings sets (18 months) for batch models.

In the validation step, the most recent historic data is used to generate predictions with each forecasting model, as shown in Figure 10. These forecasts are compared with the electricity demand data (that was not used for model training) to determine each model's performance. Again, the size of the data set used for validation may vary by application, but in our implementation, the length of the validation set is equal to twice the desired forecast length (i.e. for a 6 hour forecast, the previous 12 hourly electricity demand data points are reserved for validation).

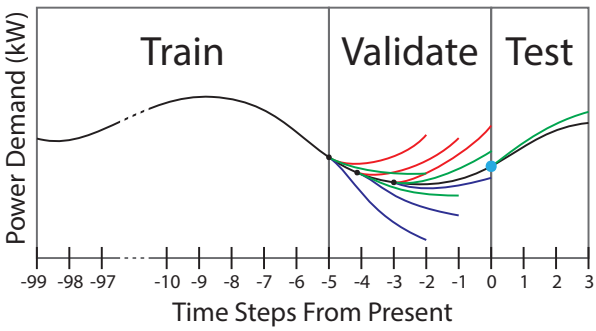

Figure 10: Graphical Representation of Gated Ensemble Regression Method with 3 Models and 3 Hour Forecast Horizon

The measure of a model's performance or the criteria for identifying the "best" model will depend on the application. In some cases, we may want to define "best" as the forecast that will produce the highest pay-off or incur the least risk. In other cases, we may want the forecast with the smallest positive or negative error. In this paper, we will focus on producing the most accurate forecast as defined by the lowest root mean squared error (RMSE).

In the gating step, a method is applied to select a single model from the bucket of models according to its relative performance during the validation step. In other words, the gating method is responsible for choosing which model in the bucket of models will be used in the test step to generate the electricity demand prediction. The objective of the gating method is to select the best model based on present and/or past information from the validation step. In this paper, we will implement and compare 2 alternative gating methods. The first method is cross-validation selection (CV). Put simply, the CV gate will select the forecasting model that performed best (produced the lowest RMSE) during the validation step. The CV gate can be considered a greedy approach because it has no memory of how its past decisions impacted the accuracy of the test prediction. Instead, the model selection is based entirely of the current performances in the validation step.

The second gating method uses a single recursively trained linear regression model (SR) to predict the performance of each forecaster in the bucket of models. Given n forecasters, the SR model input $x \in \mathbf{R}^{n+1}$ is the forecasting model's performance during the validation step (i.e. $x_{n+1} =$ Past RMSE of forecaster j) and a sparse binary

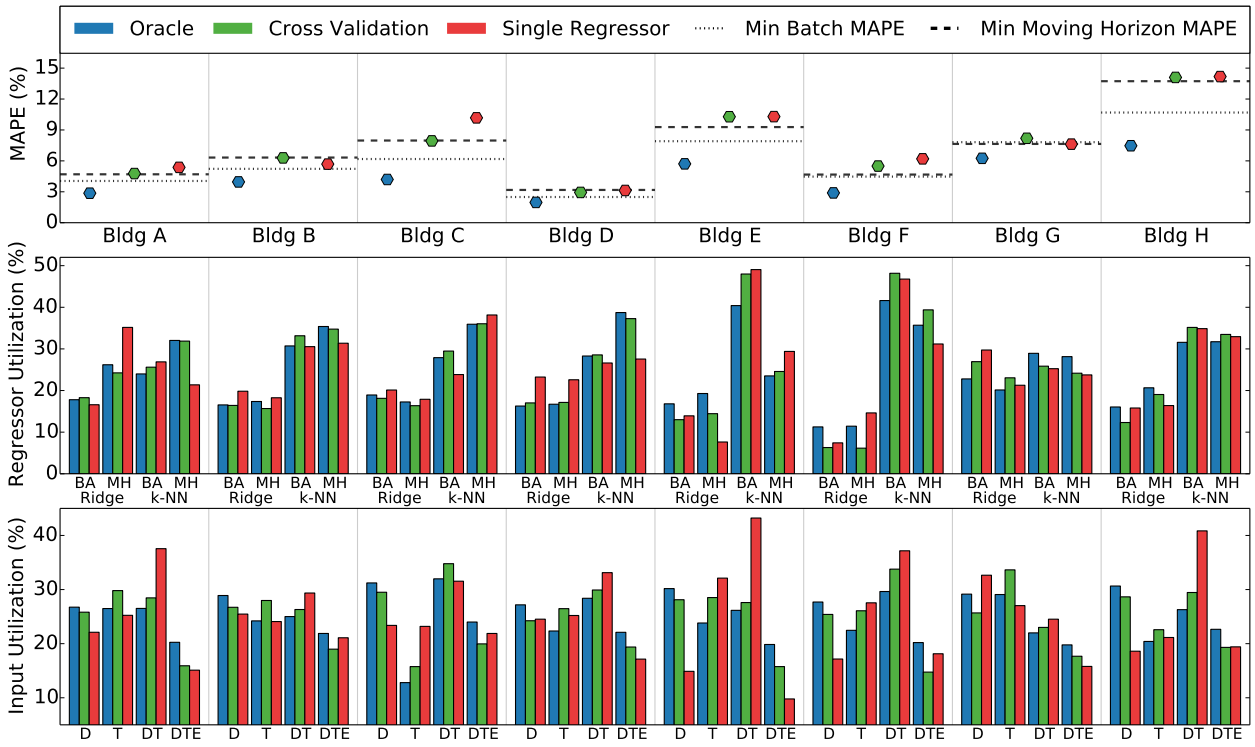

Figure 11: Gated Ensemble Study Results

vector representing each forecaster (i.e. for forecaster j, $x_j = 1$ and $x_i = 0$ for $i = 1, \ldots, j-1, j+1, \ldots, n$). The SR model output $\hat{y} \in \mathbf{R}$ is the forecaster's predicted performance during the test step (i.e. $\hat{y}$ = Predicted RMSE of forecaster j). In other words, the linear model includes a parameter for each of the regression models and a parameter corresponding to the validation RMSE. Thus, given a forecasting model and its performance during the validation step, we will train the linear model to predict the performance during the test step. In this way, the SR gate learns how well a model's validation performance does or does not indicate the performance during the test step. This capability should help to avoid models that perform poorly and favor models that perform well. The model with the best predicted performance (i.e. lowest $\hat{y}$) is selected by the SR gate for use in the test step.

Finally, in the testing step, the model selected in the gating step is used to generate an electricity demand forecast. Our multiple model method can be summarized by the following steps:

1. Train each model on a subset of the historical data.
2. Validate each model using historic data that immediately precedes the current time step.
3. Apply a gating method to select a model according to its performance during validation.
4. Use the selected model to generate a prediction.

5. MULTIPLE MODEL STUDY

To test our approach, we have implemented the gated ensemble learning method using two regression models [Ordinary Least Squares with ℓ_2 Regularization (Ridge) and k-Nearest Neighbors (k-NN)] and four input types [electricity demand (D), time (T), electricity demand and time (DT), and electricity demand, time, and exogenous weather data (DTE)]. Despite the poor performance of the demand only and time only inputs in the batch and moving horizon studies, we have included them to observe if the gating methods choose to avoid the inputs. Therefore, the bucket of models will include the best (k-NN with DT) and the worst (Ridge with T) forecasting models from the single model studies. We have elected to exclude the ANN models from the ensemble due to their computational complexity.

The regression models are trained in both batch (BA) and moving horizon (MH) manners, for a total of 16 forecasting models per building. Next, we generated a 6 hour electricity demand forecast ($\hat{y} \in \mathbf{R}^6$) for every hour between July 1st, 2013, and January 1st, 2014 using the Training, Validation, Gating, and Testing steps described above. The batch models are trained once on 18 months of data and the moving horizon models are trained at every time step on 3 months of data.

The results, employing both of the gating methods described, are presented in Figure 11. For testing purposes, we have also implemented an Oracle gate, which simply selects the best prediction for each time step regardless of the performances of the models in the validation step. The results from the Oracle gate represent the theoretical optimal of the ensemble approach.

In Figure 11, the top subplot presents the overall performance of each gate as measured by the MAPE of the selected predictions. The dotted lines represent the best performance (lowest MAPE) of any single model from the batch study and the dashed lines, from the moving horizon study. The results show that our ensemble method is able to perform comparably to the best forecaster from the batch and moving horizon studies without any prior knowledge of the energy end-use or the relative performance of the contained forecasting models. The Oracle gate is able to outperform the ANN models from the batch study, however, the cross-validation gate (CV) and single regression (SR) gate are not. Among the gating methods studied, there is no clear winner. The SR gate shows a small advantage in buildings B and G, but the worst performance in building C. In each of the buildings, the CV gate performs comparably to the best model from the moving horizon study. The performance of the Oracle gate, particularly in buildings C, E, and H, suggests there is still potential for a gating method (not presented here) to further improve our ensemble approach.

The second subplot shows the percent utilization of each regression model by the Oracle, CV, and SR gating methods, regardless of input type. In other words, the plot shows the percentage of time steps that a particular regression model and training manner was selected by the respective gating method. Because the Oracle gate represents the optimal decision given the forecasting models in the set, we would like to see comparable utilization percentages between the Oracle and the other gates.

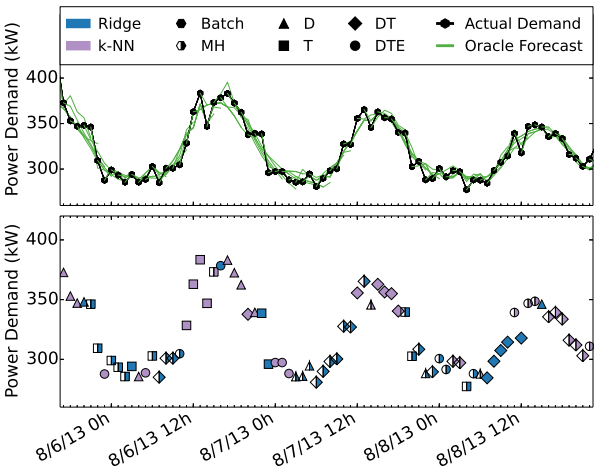

Figure 12: Building A Oracle Forecast Sample

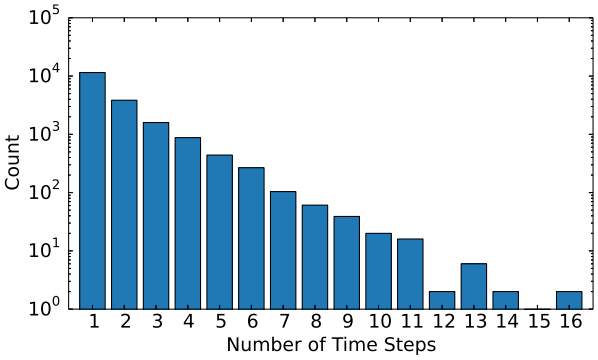

Figure 13: Optimal Prediction Generation by a Forecaster over Consecutive Time Steps When Applied to Commercial Dataset

Based on the subplot, this is generally the case with respect to regression model selection.

The bottom subplot shows the percent utilization of each input type by the Oracle, CV, and SR gating methods. Here, we observe larger differences between the behaviors of the gates. For Buildings A, E, and F, the SR gate favors the electricity demand and time (DT) inputs much more than the Oracle gate. For Buildings E and D, the SR gate underutilizes the electricity demand only (D) input compared to the CV and Oracle gates. For Building C, the CV and Oracle gates avoid the time only (T) input but the SR gate does not. It should be noted that neither of these subplots quantifies the impact of the percent utilizations on the MAPE of the gates. Nonetheless, the percent utilization metric provides useful insight into the decision making of each gate.

In general, we see very similar utilization rates between the Oracle and CV gates. This does not mean that both gates select the same forecaster at the each time step, but does suggest that the CV gate is capable of identifying forecasters that work well for a certain building. In Buildings E and F, we see that all 3 gates favor the k-NN models over the Ridge models. Additionally, in each of the buildings, the inclusion of exogenous weather data does not appear to significantly improve the accuracy of the predictions, as suggested by the Oracle's DTE utilization rate. This could mean that electricity demand is not strongly correlated with weather or that correlation only appears under certain conditions (e.g. an unusually cold or warm day).

Figure 12 presents a sample of the predictions for Building A. The top subplot shows the actual electricity demand data from 8/6/13 to 8/9/13 and a series of multivariate forecasts selected by the Oracle gate (i.e. the most accurate forecast at each time step). The bottom subplot details which forecasting model generated the selected prediction. The marker color denotes the regression model and the marker shape, the input type. A filled marker indicates that the model was trained in a batch manner and a half-filled marker, a moving horizon manner.

Since the Oracle gate chooses the most accurate forecast at each time step, Figure 12 supports the notion that a certain forecaster may generate the best prediction several time steps in a row. Figure 13 shows, for all 8 buildings, the number of times a forecaster produced the best prediction over multiple consecutive time steps and the lengths of such sequences. For the validation step to properly inform the gating method, we would like to observe high frequencies of large repeated model sequences. Using fewer models would, of course, increase the probability of such sequences at the loss of potential performance.

6. RESIDENTIAL STUDY

To further test our ensemble approach, we have repeated the single model batch study, single model moving horizon study, and multiple model ensemble study using 24 residential electricity demand datasets. This time-series demand data has been downloaded by the individual customers from the electric utility's website and provided to our research group. Each dataset is from a home in northern California but has been anonymized such that we do not know its exact location. Accordingly, for this study, we have excluded the forecasting models that use weather data as inputs.

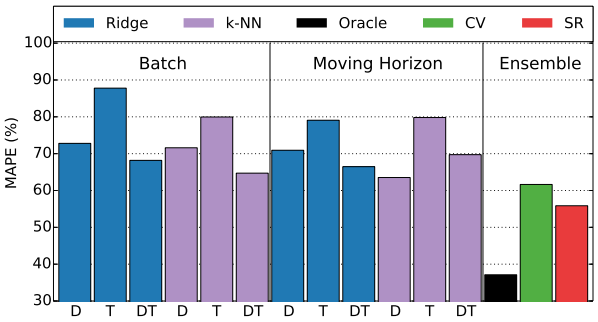

Figure 14: Residential Study Results

We utilize two regression models [Ordinary Least Squares with ℓ_2 Regularization (Ridge) and k-Nearest Neighbors (k-NN)] and three input types [electricity demand (D), time (T), and demand and time (DT)]. The regression models are trained in both batch (BA) and moving horizon (MH) manners, for a total of 12 forecasting models. In each of the studies, we generate a 6 hour electricity demand forecast ($\hat{y} \in \mathbf{R}^6$) for every hour between October 1st, 2014, and January 1st, 2015. The batch models are trained once on 9 months of data and the moving horizon models are trained at every time step on 3 months of data.

The results from the single model batch, single model moving horizon, and multiple model ensemble studies are presented in Figure 14. In the figure, each bar represents the mean absolute percent error (MAPE) of the respective predictions across all 24 residential building datasets. Moving from left to right, the first 6 bars represent the results from the single model batch study and the next 6 bars, the single model moving horizon study. Here, we observe that the MAPEs are much larger for the residential predictions than for the commercial/university predictions. This can be explained by the fact that the residential electricity demands are much smaller and are composed of fewer loads than the commercial/university demands. Therefore, any change to the residential demand produces a larger percent change and any prediction error will correspond to a larger percent error. In the single model studies, the highest performing forecaster is the k-Nearest Neighbors regression model with demand input (D) trained in a moving horizon manner (MAPE: 63.5%).

In Figure 14, the Oracle, CV, and SR bars

show the results produced by our ensemble learning method. As previously stated, the results from the Oracle gate (MAPE: 37.1%) represent the optimal potential given the forecasters in the bucket of models. The CV gate represents the results (MAPE: 61.5%) using cross-validation to select which forecaster to utilize. Here, we observe that the CV gate performs comparably to the best forecaster in the single model studies. The SR gate represents the results (MAPE: 55.8%) using a single recursively trained linear regression model to select a forecaster by predicting the performance of each model. These results suggest that the SR gate outperforms the CV gate as well as any of the single model approaches. In other words, the SR gate is able to learn from the past performance of each model and to make better decisions about which model to select at each time step.

The Oracle gate's percent utilization of each forecaster in the bucket of models is shown in Figure 15. These results show that, across the 24 residential electricity demand datasets, the Oracle gate shows a slight preference for the k-Nearest Neighbors models. This does not indicate that the k-NN models perform significantly better (lower RMSE) than the Ridge models, only that the k-NN models produce the best prediction with a higher frequency.

Lastly, Figure 16 shows, for all 24 residential buildings, the number of times a forecaster produced the best prediction over multiple consecutive time steps and the lengths of such sequences. Compared to the commercial/university buildings (Figure 13), we observe larger sequences of optimal prediction generation when applying the ensemble method to the residential buildings. It should be noted that for the residential studies we have not incorporated weather data and are thus using 4 fewer forecasting models than in the commercial/university studies. Nonetheless, the Oracle's forecaster utilization rates (ranging from 5.0% to 14.6%) and the presence of large optimal prediction generation sequences support the underlying assertion of our ensemble learning method: by selecting between multiple models at each time step, we can obtain better results than a single model approach.

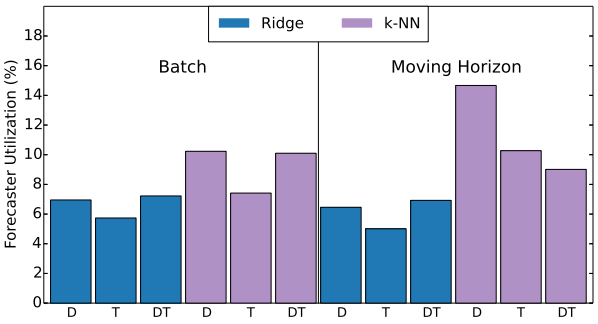

Figure 15: Oracle Forecaster Utilization

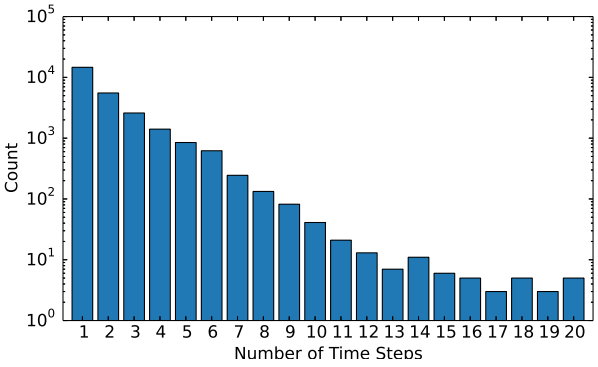

Figure 16: Optimal Prediction Generation by a Forecaster over Consecutive Time Steps When Applied to Residential Dataset

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a gated ensemble learning method for short-term electricity demand forecasting. The contribution of this method is to allow for the incorporation of multiple forecasting models trained in both batch and moving horizon manners. Stated more explicitly, rather than choosing a single approach, an engineer can utilize multiple models with the intention of improving the reliability of the forecaster to produce useful results. This makes the method well suited for real world applications. At deployment, the moving horizon models can be utilized until sufficient data is available to train the batch models. This adaptability also makes the method suitable for control applications. Rather than assuming that demand behaviors are time invariant, the method will work to recognize both long and short-term electricity demand patterns.

The relative performance of the Oracle gate suggests that there is potential for continued development of the gating functions. For instance, none of the gating functions attempt to identify repeated model selection sequences. Given the optimal model in the previous few time steps, a gating method could determine the probability that a model will be optimal in the next time step. Also, in our implementation, the validation step uses the

RMSE of 6 multivariate forecasts to measure the performance of each model. It may be more effective to use fewer forecasts or even univariate predictions in order to identify the optimal model at the given time step. Finally, the addition of a feature selection procedure may help to reduce the dimensionality of the regression models or even eliminate certain input types from consideration.

By applying our gated ensemble learning method to 32 unique building electricity demand data sets (8 commercial/university and 24 residential), we demonstrate that the incorporation of multiple models can yield better results than a single model approach. While the development of the gating methods is ongoing, the ability of each gate to perform model validation and selection in real time greatly improves the method's adaptability and ease of use. Utilizing this data-driven approach, we empirically show that the ensemble method is capable of aiding in the production of accurate multivariate electricity demand forecasts for building-level applications.

[1] U.S. Energy Information Administration, Monthly Energy Review (March 2015).
URL `http://www.eia.gov/mer`

[2] E. Hirst, B. Kirby, Separating and measuring the regulation and load-following ancillary services, Utilities Policy 8 (1999) 75–81.

[3] H. K. Alfares, M. Nazeeruddin, Electric load forecasting: literature survey and classification of methods, International Journal of Systems Science 33 (1) (2002) 23–34. doi:10.1080/00207720110067421.

[4] G. K. F. Tso, K. K. W. Yau, Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks, Energy 32 (9) (2007) 1761–1768. doi:10.1016/j.energy.2006.11.010.

[5] J. G. Jetcheva, M. Majidpour, W.-P. Chen, Neural network model ensembles for building-level electricity load forecasts, Energy and Buildings 84 (2014) 214223. doi:doi:10.1016/j.enbuild.2014.08.004.

[6] F. Javeda, N. Arshada, F. Wallinb, I. Vassilevab, E. Dahlquistb, Forecasting for demand response in smart grids: An analysis on use of anthropologic and structural data and short term multiple loads forecasting, Applied Energy 96 (2012) 150–160.

[7] A. Azadeh, S. F. Ghaderi, S. Tarverdian, M. Saberi, Integration of artificial neural networks and genetic algorithm to predict electrical energy consumption, Applied Mathematics and Computation 186 (2) (2007) 1731–1741. doi:10.1016/j.amc.2006.08.093.

[8] H. S. Hippert, C. E. Pedreira, R. C. Souza, Neural networks for short-term load forecasting: A review and evaluation, IEEE Transactions on Power Systems 16 (1) (2001) 44–55. doi:10.1109/59.910780.

[9] S. Karatasou, M. Santamouris, V. Geros, Modeling and predicting building's energy use with artificial neural networks: Methods and results, Energy and Buildings 38 (8) (2006) 949–958.

[10] K. Metaxiotis, A. Kagiannas, D. Askounis, J. Psarras, Artificial intelligence in short term electric load forecasting: a state-of-the-art survey for the researcher, Energy Conversion and Management 44 (9) (2003) 1525–1534. doi:10.1016/S0196-8904(02)00148-6.

[11] J. Massana, C. Pous, L. Burgas, J. Melendez, J. Colomer, Short-term load forecasting in a non-residential building contrasting models and attributes, Energy and Buildings 92 (2015) 322–330. doi:doi:10.1016/j.enbuild.2015.02.007.

[12] P. F. Pai, W. C. Hong, Support vector machines with simulated annealing algorithms in electricity load forecasting, Energy Conversion and Management 46 (17) (2005) 2669–2688. doi:10.1016/j.enconman.2005.02.004.

[13] A. Kavousi-Fard, H. Samet, F. Marzbani, A new hybrid modified firefly algorithm and support vector regression model for accurate short term load forecasting, Expert Systems with Applications 41 (13) (2014) 6047–6056. doi:10.1016/j.eswa.2014.03.053.

[14] W.-C. Hong, Electric load forecasting by support vector model, Applied Mathematical Modelling 33 (5) (2009) 2444–2454. doi:10.1016/j.apm.2008.07.010.

[15] H. T. Yang, C. M. Huang, C. L. Huang, Identification of ARMAX model for short term load forecasting: An evolutionary programming approach, IEEE Transactions on Power Systems 11 (1) (1996) 403–408.

[16] G. A. Darbellay, M. Slama, Forecasting the short-term demand for electricity - do neural networks stand a better chance?, International Journal of Forecasting 16 (1) (2000) 71–83. doi:10.1016/S0169-2070(99)00045-X.

[17] J. W. Taylor, L. M. de Menezes, P. E. McSharry, A comparison of univariate methods for forecasting electricity demand up to a day ahead, International Journal of Forecasting 22 (1) (2006) 1–16. doi:10.1016/j.ijforecast.2005.06.006.

[18] G. R. Newsham, B. J. Birt, Building-level occupancy data to improve arima-based electricity use forecasts, in: Proceedings of the 2nd ACM workshop on embedded sensing systems for energy-efficiency in building, ACM, 2010, pp. 13–18.

[19] S. Rahman, O. Hazim, A generalized knowledge-based short-term load-forecasting technique, IEEE Transactions on Power Systems 8 (2) (1993) 508–514. doi:10.1109/59.260833.

[20] K. Li, H. Su, J. Chu, Forecasting building energy consumption using neural networks and hybrid neuro-fuzzy system: A comparative study, Energy and Buildings 43 (2011) 2893–2899.

[21] A. J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and computing 14 (3) (2004) 199–222.

[22] C.-J. Lu, T.-S. Lee, C.-C. Chiu, Financial time series forecasting using independent component analysis and support vector regression, Decision Support Systems 47 (2) (2009) 115–125.

[23] G. De'ath, K. E. Fabricius, Classification and regression trees: a powerful yet simple technique for ecological data analysis, Ecology 81 (11) (2000) 3178–3192.

[24] S. M. Omohundro, Five balltree construction algorithms, International Computer Science Institute Berkeley, 1989.

[25] J. E. Dayhoff, J. M. DeLeo, Artificial neural networks, Cancer 91 (S8) (2001) 1615–1635.

[26] T. Schaul, J. Bayer, D. Wierstra, Y. Sun, M. Felder, F. Sehnke, T. Rückstieß, J. Schmidhuber, Pybrain: an

open source machine-learning library written in python, Journal of Machine Learning Research 11 (2010) 743–746.

[27] 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, E. Duchesnay, Scikit-learn: Machine learning in python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[28] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing In Science & Engineering 9 (3) (2007) 90–95.

[29] R. L. Snyder, California irrigation management information system, American Journal of Potato Research 61 (4) (1984) 229–234.
URL `http://www.cimis.water.ca.gov/`

[30] C. C. Aggarwal, Outlier ensembles, ACM SIGKDD Explorations Newsletter 14 (2) (2012) 49–58.

[31] S. Deroski, B. enko, Is combining classifiers with stacking better than selecting the best one?, Machine Learning 54 (3) (2004) 255–273.