ENSEMBLING APPROACHES TO HIERARCHICAL ELECTRIC LOAD FORECASTING

James Barrios
Stanford University
jbarrio5@stanford.edu

Simon Gleeson
Stanford University
sgleeson@stanford.edu

Charlie Natoli
Stanford University
cnatoli@stanford.edu

1 Introduction

Short term electrical load forecasting is critical in ensuring reliability and operational efficiency for electrical systems. With an influx of monitoring data and the growing technical complexity of the grid, there is a great interest and need for accurate forecasting in electricity planning. Our project uses a curated electric load dataset from Kaggle and evaluates the performance of several different load forecasting methods. We first implement a simple parametric regression, where we divide the problem into subproblems by key indicator variables and fit multiple regressions. Second, we use a similar weather load input similar to Chen et. al [3] and fit a Neural Network. Third, we decompose the load into low and high frequency profiles and fit two separate networks to predict these components. Last, we evaluate linear combinations of these models to optimize performance on the validation set.

2 Dataset

The dataset we are working with was sourced from Kaggle [1], which was used for the Global Energy Forecasting Competition (GEFcom) in 2012. The dataset contains hourly load measurements for 20 geographic subareas (zones) from January 2004 to June 2008. It also includes hourly temperature readings from 11 weather stations in that region, but does not include information on which weather station maps to which load zone. Eight separate weeks of load measurements were omitted from the dataset, and the contestants were tasked with backcasting the load for these weeks based on other load measurements and the hourly temperature. For each problem, they also had to aggregate their prediction to estimate the total load over all 20 zones. The evaluation metric used in the competition was a weighted mean squared error of the form

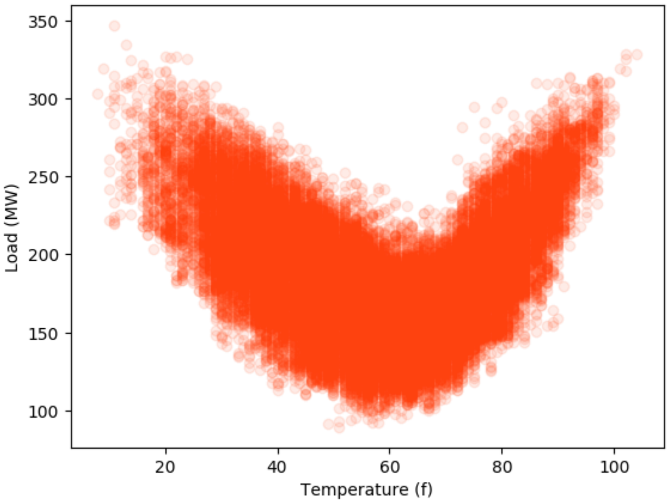
$$WRMSE = \frac{\sum_{j=1}^{21}\sum_{i=1}^{n} w_j (P_i - A_i)^2}{\sum_{j=1}^{21}\sum_{i=1}^{n} w_i} \tag{1}$$

Where P_i is the predicted value and A_i is the actual value. w_j is the weighting for zone j, zones 1 to 20 have weights of 1, and total system load weight is $w_{21} = 20$. n is the number of observations we are predicting per load zone.

2.1 Exploration

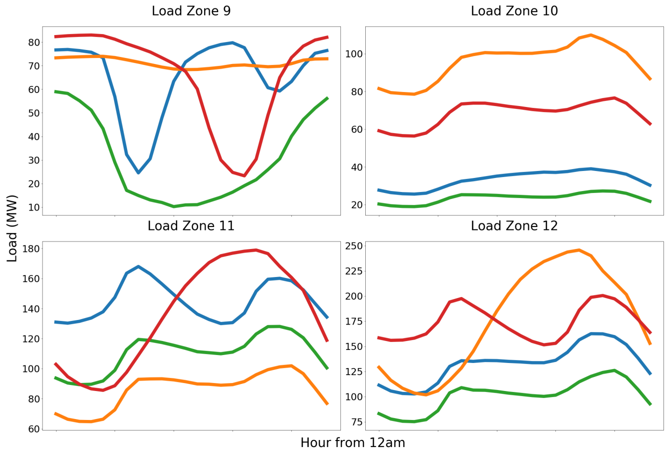
We observe a quadratic relationship between the temperature and load, which agrees with conventional load and temperature intuition (heating in cold weather and cooling in hot weather are the main drivers of energy use). The only exception we found here was zone 9, which is said to be largely industrial and therefore has different electricity usage patterns. Zone 10 also exhibits a slightly weaker relationship between load and temperature. While there is no information on this zone, we assume it is likely to be tied to predominant industrial use as well.

Figure 1: Temperature Versus Load For Zone 7



In order to analyze how loads shapes changed with time, we ran k means clustering on days from each load zone with $k = 4$ to examine the common daily load profiles for each zone, where μ_k is a 24 hour load profile for a given zone, and each day corresponds to a vector $v_d \in \mathbb{R}^{24}$.

Figure 2: Typical load profiles for zones 9 to 12



[1] https://www.kaggle.com/c/global-energy-forecasting-competition-2012-load-forecasting

Zones 11 and 12 exhibit the usual patterns observed in the 20 zones, the load peaks in the early afternoon hours, which is typical of the load on a summer day, with the cooling load occurring in the late afternoon. The patterns with two bumps correspond to heating in the morning and evening during the winter months. Zone 9 and 10's patterns are incongruous, and this atypical behavior motivates us to model each zone separately, as well as consider different models for different seasons.

3 Previous Work

Traditional time series approaches such as ARIMA and ARMA, while still in use [7], have become overshadowed in the face of more complex artificial intelligent modeling approaches, as they do not exploit additional features as effectively as more modern approaches.

Neural Networks have recently emerged as a popular way to forecast load. Marino et. al implemented a Long Term Short Memory (LTSM) Sequence to Sequence approach to model household electricity consumption, with an encoder network to measure current errors and a decoder error to forecast into the future, the result significantly outperformed the standard LTSM network [5]. Chen et. al also implemented a Neural Network approach where they fed in similar days (in terms of weather) from the training set rather than the previous day, as might be common under a more traditional time series approach [3]. They also used wavelet decomposition to separate out fluctuations and underlying trends, and forecasted high frequency and low frequency loads separately, with successful results.

Support vector regression (SVR) forecasting has also emerged as a recent state of the art model for electric load forecasting. Baziar uses SVR to outperform both artificial Neural Network and ARIMA based methods [1]. While SVRs have shown potential with short-term load forecasting, they are hindered by the fact that they do not perform well in a high dimensional data settings and are much better suited for working on smaller datasets with fewer overlapping classes. We initially explored the use of SVR as a potential method for predicting the load in our dataset, but ultimately decided that it was too computationally expensive to incorporate the number of examples and features we have into such a model.

Interestingly, for GEFcom 2012, the teams that performed the best employed completely different approaches to each other [4]. The first place team initially disaggregated the problem into 1920 parametric regressions based on season (summer or winter), day type (weekday or weekend), hour of day and zone number (2 × 2 × 24 × 20), and for each combination of terms, chose the model that minimized the mean squared error (MSE). More formally, for a given group $g = (z, s, h, t)$ where z is the zone, s is the season, h is the hour, and t is the daytype, for each weather zone i they fit:

$$E_i = f_{zsht}(d, t_i),$$

where E_i is the linear model, f is a linear function, d is a variable relating to the day, and t_i is the temperature at station i. They then choose i such that the in-sample squared error was minimized. From there, they implemented minor tweaks, new variables and temperature smoothing to get their end model [2].

4 Methods

4.1 Evaluation

For our project we emulate the structure of the backcasting component of GEFcom 2012, that is, we train our models on the 4.5 years of data with the same 8 weeks of data taken out, and evaluate our model on a randomly chosen subset of the days (25%). We choose this structure for two reasons; first, we can draw on the methods that have been successful in the past and augment them to improve our own forecast. Second, the competition results can be used to benchmark our model performance. After we have completed our models, we evaluate on the whole 8 weeks, as was done in the competition.

4.2 Models

4.2.1 Parametric Regression

For our baseline model, we implemented the initial model of what was proposed by Charlton and Singleton [2], with 4 seasons instead of 2 (3840 models corresponding to season, day type, hour and zone, $4 \times 2 \times 24 \times 20$).

4.2.2 Neural Network

For our Neural Network model, we trained models unique to each load zone to account for zone-specific phenomena as shows in Figure 2. Input data to the neural networks included calendar effects, weather at each of the 11 stations, and, similar to the method used in Chen et al [3], we also included the load at the day in the training set with the most similar weather (measured by the total euclidean distance between the 11 weather readings for the days). Given the zone specific neural networks, we expect that each model will be able to learn which weather stations were most representative of its load. All Neural Networks in this study were trained using Keras [2] .

4.2.3 Wavelet Decomposition Neural Network

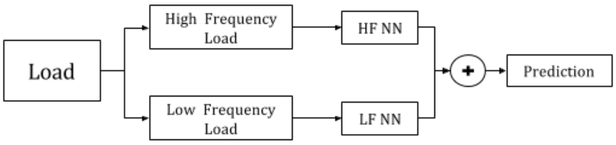
We follow an approach similar to Chen et. al [3] and decompose our load through Daubechies D4 Wavelet Transform [6]. This transform is advantageous as it allows for complete reconstruction of the original load. For each load zone, we take the load profile, s and calculate the low frequency and high frequency load as follows:

$$\begin{aligned} lf_i &= h_0 s_{2i} + h_1 s_{2i+1} + h_2 s_{2i+2} + h_3 s_{2i+3}\, \forall i \leq \frac{n}{2} - 2 \\ hf_i &= h_3 s_{2i} - h_2 s_{2i+1} + h_1 s_{2i+2} + h_0 s_{2i+3}\, \forall i \leq \frac{n}{2} - 2 \end{aligned} \qquad (2)$$

After treatment of edge cases (see Appendix), we have two vectors of length $\frac{n}{2}$. We map these values to their original load to give a smoothed load value and a high frequency variation value.

[2] https://keras.io/

Figure 3: Wavelet Decomposition Neural Network Structure



As shown in Figure 3, two networks are trained to predict the high frequency load and the low frequency load. The inputs to both of the model are same as those for the standard neural network, except for that we do not include the month and year when training the high frequency load. We experimented with different configurations of networks, but these inputs had the strongest performance, and share a similar structure to those of Chen et al.

4.3 Experiments

For our Neural Network, we experimented with different model architectures, activation functions and regularization methods to improve the performance on the validation set. Because of time limitations, we chose a small subset of zones that were most representative of the typical load profile observed in the training data, namely zones 1 and 3.

We were concerned that our model was overfitting to the training data, so we experimented with regularization in two ways. First, we vary the number of training epochs so that the model has less time to over learn the training data structure. Second, we implement dropout regularization, where a random subset of the connections between the neurons are dropped from the model architecture. Implementing dropout does not have any significant effect on the performance of the network, and thus we omit it from our final run. Further, limiting the number of training epochs increased our validation error, so we decide to experiment with larger numbers of training epochs.

After experimenting with a range of training epochs we found that 20 epochs performed best. Similarly, we experimented with different activation functions (ReLu and eLu), number of neurons per layer and number of layers. Our results showed that the model with the lowest validation error was a model with 3 layers with 150, 300 and 150 neurons respectively. They all used a ReLu activation function, and the final output neuron uses a linear activation function. When optimizing over different activation functions, training length and model architecture we minimize training error, not validation error.

We experimented with different inputs to the wavelet decomposition, but training 2 neural networks for 20 zones was computationally expensive, so we evaluated our performance on a subset of zones that were representative of the general load shape. We found that both models had to have hour dummy variables as inputs to be able to capture the variation in frequencies, but that the short term frequency performed strongly without the month or year dummy variables.

5 Results

The results of our models alongside the winner of the Kaggle competition are included below:

Table 2 Model validation set scores.

Model	Validation	Test
Baseline	81,502	84,309
Neural Network	119,133	123,246
Wavelet Decomposition	123,749	120,097
Ensemble	76,639	83,673
Competition Winner		61,890

While our models do not beat the best performance from the competition, they do show that the ensembling of a parametric form of model with deep learning can improve the performance metric, with more time to augment the models and more computational power we expect that these numbers would further decrease, as discussed in the Next Steps section. The best performer of our model was an ensemble of the form

$$p_E = \frac{1}{9}p_N + \frac{7}{9}p_P + \frac{1}{9}p_W \tag{3}$$

With p_E, p_N, p_P, p_W being the predictions for Ensembling, Neural Network, disaggregated Parametric Regression and Wavelet decomposition respectively. These weights were found from testing a selection of different weightings based off our belief on which model would have the strongest performance. We expected the parametric regression to perform the best as the team that won the competition used this structure as their baseline as previously mentioned. The Neural Network and Wavelet decomposition tended to complement the parametric regression. In other words, when the parametric regression overestimates the load target, the neural networks would underestimate and vice versa.

Note that the baseline model has been handcrafted so that different seasons, zones, hours, and day types are modeled differently, whereas the neural networks are only separated by zones, the rest of the behaviors the network captures itself.

We show some of the forecasts on the test set in Figure 4, we observe that the models cannot predict the behaviors in zones 9 and 10 accurately. The wavelet decomposition exhibits especially erratic behaviors in zone 10, and while they are less pronounced, we observe similar behaviors with the Neural Network. This suggests that either our current features, architecture, or the general deep learning framework is failing in capturing the shapes of these atypical zones.

Table 1 shows the normalized WRMSE test scores for each zone. We normalize the WRMSE by dividing it by the average load over the time horizon for a simple measure of which zones the models show lackluster performance. Looking at the ensembling results, in most cases the ensemble model performs at comparable levels to the Parametric regressions, but it usually outperforms them marginally. Note that while the models perform the worst in zones 9 and 10, in most zones the data seems to be quite well fit. For this reason, specifying different features for these atypical zones could increase model performance. That said, not all zonal loads are on the same order of magnitude, thus if the

Table 1: Normalized WRMSE scores for each model per zone

Model	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Parametric	.10	.07	.08	.13	.14	.07	.07	.09	.22	.17	.09	.10	.10	.13	.10	.13	.08	.09	.11	.08
Neural Network	.13	.1	.12	.13	.18	.11	.10	.12	.32	.30	.13	.15	.11	.17	.11	.13	.10	.17	.15	.12
Wavelet	.11	.13	.10	.13	.18	.10	.09	.12	.27	.48	.13	.17	.16	.15	.11	.17	.14	.16	.13	.10
Ensemble	.09	.07	.07	.11	.14	.07	.07	.09	.24	.15	.09	.11	.10	.11	.11	.11	.07	.10	.10	.08

end goal is only to minimize the performance metric, it may be best to concentrate attentions on larger load zones. We consider the dual goal of minimizing WRMSE while capturing each load zone's general structure, although it may not contend as the best model to minimize WRMSE.

6 Conclusion

Predicting electric load in hierarchical structures benefits from disaggregation of the total load into different zones. It is advantageous to model and consider these load zones separately, as they often have completely different behaviors. Our approach ensembled different prediction architectures to show superior performance to each network on its own, we find that through weighting the predictions of our three models, we decrease the WRMSE on the test set. The parametric regression, while the strongest performer, tends to overestimate, and for this reason 'hedging' the prediction through incorporating deep learning predictions helps improve the accuracy of the model performance.

7 Next steps

With time permitting, we would have liked to further experiment with the inputs and design of the wavelet decomposition network. Due to large train times it was not feasible to run multiple tests, and there were experiments, such as running regularization on the models, that we expected could increase model performance as well.

For all models, we would like to hand craft features and architectures for load zones that are not modeled well under our current architecture, such as zone 9 and 10, as can be seen in Figure 4. Additionally, our current approaches ensembles with the same constant regardless of load zones, season etcetera. It would be interesting to allow for ensembling to vary for each load zone, as models may perform well for some load zones, but weakly for others.

8 Appendix

Parametric Regression methodology

As mentioned in section 4, we disaggregate the data by season, weather zone, hour and weekday. Our parametric regression is of the form

$$L = \alpha_1 + \alpha_2 d + \alpha_3 T + \alpha_4 T d + \alpha_5 T^2 + \alpha_6 T^2 d \tag{4}$$

Where T is the weather from the load zone that minimizes the training mean squared error, and d is the total number of elapsed days from January 1st 2004.

Wavelet Decomposition methodology

The entire training set was decomposed using the equations mentioned in section 4.2.3, we handled the edge cases by 'bouncing back', that is:

$$\begin{aligned} lf_{\frac{n}{2}-1} &= h_0 s_{n-1} + h_1 s_n + h_2 s_n + h_3 s_{n-1} \\ lf_{\frac{n}{2}} &= h_0 s_n + h_1 s_n + h_2 s_{n-1} + h_3 s_{n-2} \end{aligned} \tag{5}$$

This generated one low frequency and one high frequency vector of size $\frac{n}{2}$. Each entry is repeated once to reconstruct the length of the original load. For the low frequency pattern, the entries are repeated consecutively, e.g. $lf_1, lf_1, lf_2, lf_2, ...$, for the high frequency load, the ith value is followed by the $(i+2)th$ value, e.g. $hf_1, hf_3, hf_2, hf_4, hf_3, hf_5, ...$

Next, we fit two separate Neural Networks for the long term and short term behaviors, with the same inputs except for the short term we did not input the month of year. The predictions of those networks on new data was then added to make the prediction.

9 References

[1] Aliasghar Baziar and Abdollah Kavousi-Fard. Short term load forecasting using a hybrid model based on support vector regression. 2015.

[2] Nathaniel Charlton and Colin Singleton. A refined parametric model for short term load forecasting. *International Journal of Forecasting*, 30(2):364 – 368, 2014.

[3] Ying Chen, P. B. Luh, and S. J. Rourke. Short-term load forecasting: Similar day-based wavelet neural networks. In *2008 7th World Congress on Intelligent Control and Automation*, pages 3353–3358, June 2008.

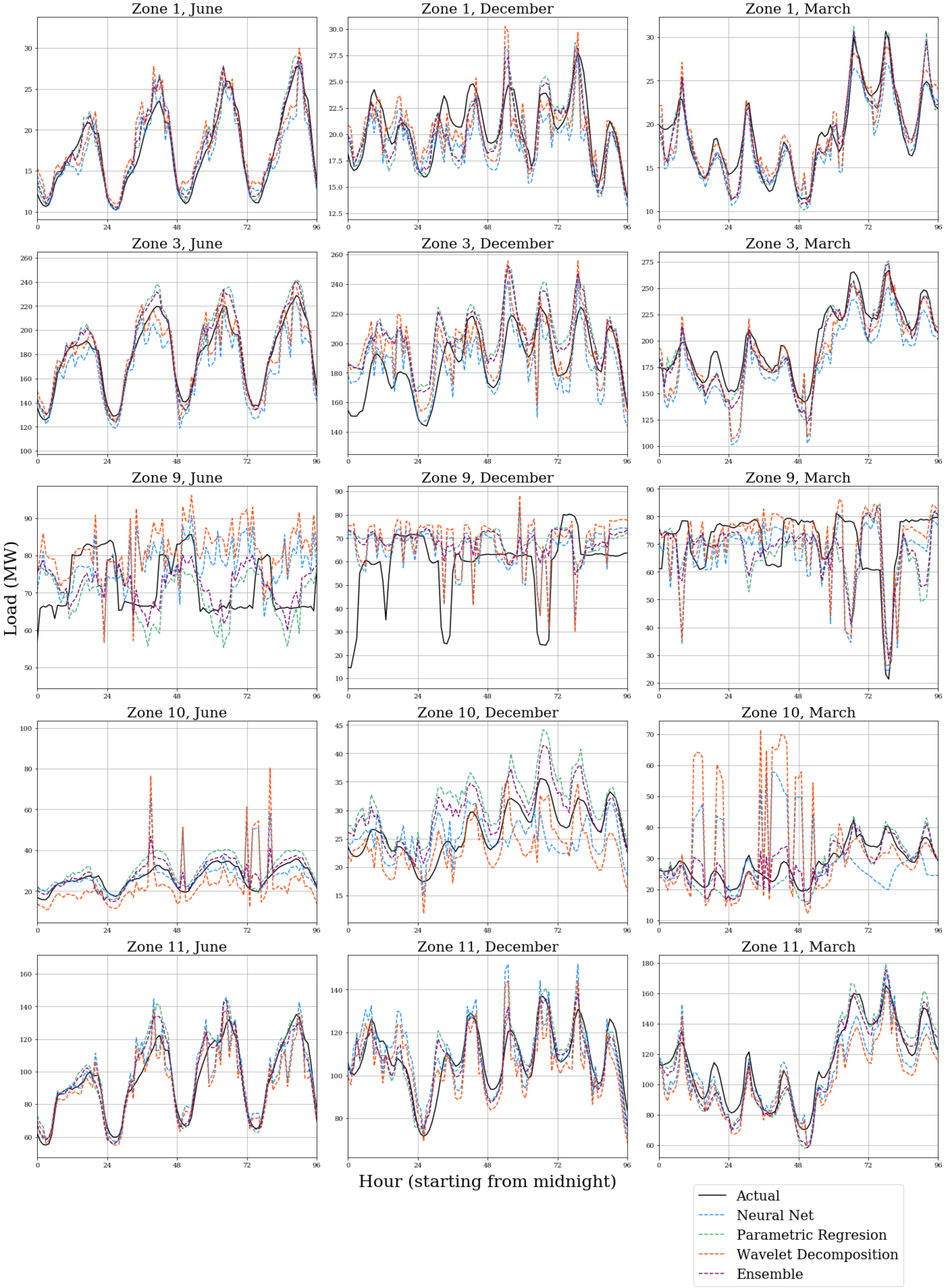
[4] Tao Hong, Pierre Pinson, and Shu Fan. Global energy forecasting competition 2012. *International Journal of Forecasting*, 30(2):357 – 363, 2014.

[5] Daniel L. Marino, Kasun Amarasinghe, and Milos Manic. Building energy load forecasting using deep neural networks. *CoRR*, abs/1610.09460, 2016.

[6] Yves Meyer. Daubechies wavelets. 1999.

[7] Nataraja.C, Gorawar, and Shri Harsha.J. Short term load forecasting using time series analysis: A case study for karnataka, india. 2012.

Figure 4: Model Performance Across Five Zones and Three Seasons.


Zone 1, June
Zone 1, December
Zone 1, March
Zone 3, June
Zone 3, December
Zone 3, March
Zone 9, June
Zone 9, December
Zone 9, March
Zone 10, June
Zone 10, December
Zone 10, March
Zone 11, June
Zone 11, December
Zone 11, March
Load (MW)
Hour (starting from midnight)
Actual
Neural Net
Parametric Regresion
Wavelet Decomposition
Ensemble