Deep Neural Network Regression for Short-Term Load Forecasting of Natural Gas

Gregory D. Merkel
Department of Electrical and Computer Engineering
Marquette University
Milwaukee, WI

Richard J. Povinelli
Department of Electrical and Computer Engineering
Marquette University
Milwaukee, WI

Ronald H. Brown
Department of Electrical and Computer Engineering
Marquette University
Milwaukee, WI

***Abstract*— This paper proposes a short-term load forecasting method for natural gas using deep learning. Deep learning has proven to be a powerful tool for many classification problems seeing significant use in machine learning fields like image recognition and speech processing. This paper explores many aspects of using deep neural networks for time series forecasting. It is determined that the proposed network outperforms traditional artificial neural networks and linear regression based forecasters.**

***Index Terms*— Deep learning, deep neural network, regression, short-term load forecasting, time-series analysis.**

I. Introduction

Short-term load forecasting is important for the day-to-day operation of natural gas utilities. Many purchasing decisions are made using these forecasts, and there can be high cost to both natural gas utilities and their customers if the short-term load forecast is inaccurate. If the forecast is low, a gas utility may have to purchase gas at a much higher price; if the forecast is high, a gas utility may have to store the excess gas.

Figure 1 illustrates a typical time series of daily load for a natural gas utility. The data, seen below, is a weighted combination of several metropolitan areas in United States.

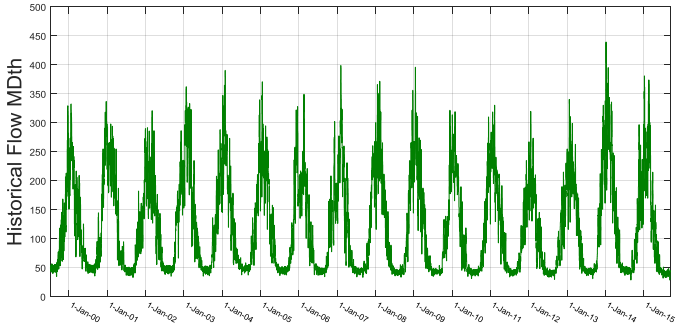

Figure 1: Natural gas load versus time for weighted combination area (1999-2015).

Traditionally, short-term load forecasting of natural gas is done using a linear regression (LR) and autoregressive integrated moving average (ARIMA) models [1]. These models perform well on linear stationary time-series, and thus have been used successfully for forecasting short-term load, which has roughly a linearly relationship with temperature [2]. This can be seen in Figure 2. Like many real-world systems, gas demand contains nonlinearities. Some of these are easy for a proficient forecaster to capture using an LR model by, for instance, using heating degree days as an input instead of temperature. However, natural gas demand also contains many nonlinearities that either cannot be easily captured with LR or ARIMA models or are impossible for forecasters to glean from the data themselves. The forecasting community's answer to this problem for many years has been using artificial neural networks (ANN) in place of, or in conjunction with, linear models, as ANNs are able to capture nonlinearities [1], [3], [4].

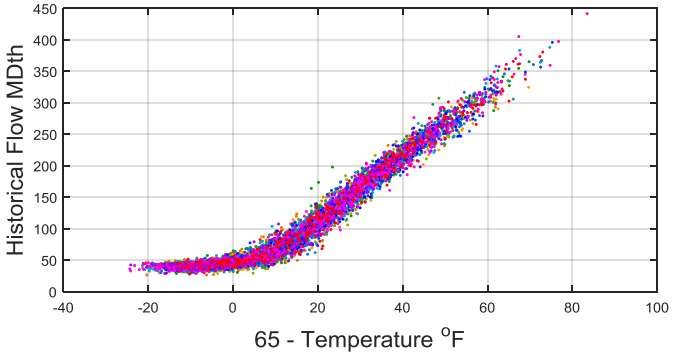

Figure 2: Natural gas load versus temperature for weighted combination area (1999-2015).

Recently, the machine learning community have been successful in replacing ANNs and other nonlinear models with deep neural networks (DNN) [5]. Längkvist discusses the use of DNNs for problems ranging from video analysis and motion capture to speech and music recognition [5]. Functionally, DNNs are just large ANNs; the main difference is in the training algorithms. ANNs are trained using gradient descent, which is computationally intensive. Large neural networks trained by gradient descent also are prone to overfitting data sets. DNNs avoid both problems by using a restricted Boltzmann machine training algorithm to "pre-train" the model, followed by a few epochs of gradient descent [6].

We will provide a short overview of deep learning and DNNs, focusing on aspects of DNNs that are important for regression problems. Then we will discuss the inputs and architecture of the proposed model. And finally, we will show

This work is supported by the GasDay laboratory at Marquette University

the results of experiments using the proposed model on 176 real natural gas operational areas. The DNNs using sigmoidal transfer functions will be compared to the performance of an ANN forecaster and an LR model. The DNN performs better than the other two models.

II. Restricted Boltzmann Machines

Fundamental to understanding DNNs are restricted Boltzmann Machines (RBM). This section will describe how they work and how they relate to DNNs.

A. Energy Based Models

RBMs are energy-based models. This means that for any input vector x they will have an associated scalar energy based on an energy function $E(x)$. A trained energy-based model will have lower energy when given inputs that are expected, while having high energy for those that are not. For example, in a short-term-load-forecasting system for natural gas, if the temperature were given as some extreme value such as 150˚F we would expect a trained energy-based model to have high energy. For a simple energy-based model the probability distribution is given as

$$p(x) = \frac{e^{-E(x)}}{Z},$$

where

$$Z = \sum_k e^{-E(k)}$$

and k represents the set of all possible inputs to the energy-based model. This simply means that the probability of vector x is equal to the exponential of the energy function divided by the sum of the exponentials of each possible vector. The goal in training the energy-based model is to have the probability distribution $p(x)$ be as close as possible to the actual probability distribution of the inputs.

B. Energy Based Models with Hidden Layers

For more complicated energy-based models like RBMs we may have hidden units as in Figure 3. For these models, the calculation becomes slightly more complicated as we must calculate the energy associated with a visible input v for each of the hidden units h. This probability distribution is given as

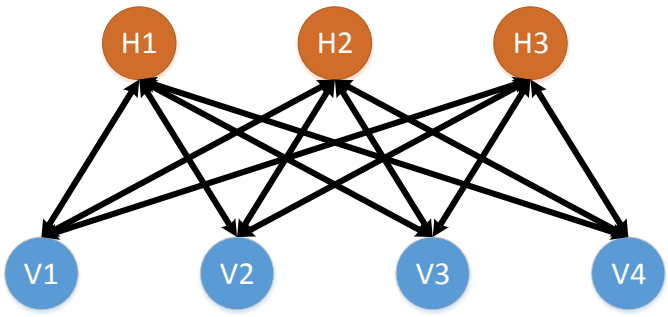

Figure 3: A restricted Boltzmann machine with four visible units and three hidden units. Note the similarity with a single layer of a neural network.

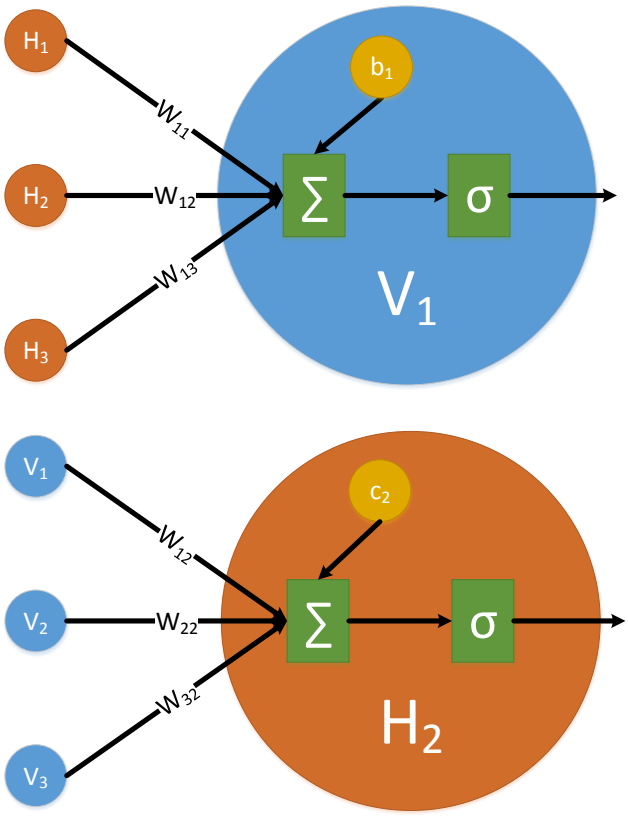

Figure 4: Visual representation of hidden and visible layer calculations. Note the similarity between these and the neurons of an artificial neural network.

$$p(v) = \sum_h p(v,h) = \sum_h \frac{e^{-E(v,h)}}{Z}$$

where

$$Z = \sum_k e^{-E(k,h)}.$$

For the sake of simplicity in notation in later equations, this can instead be written as

$$p(v) = \frac{e^{-\mathcal{F}(v)}}{Z}$$

where

$$\mathcal{F}(v) = -log \sum_h e^{-E(x,h)}.$$

$\mathcal{F}(v)$ will be referred to as the free energy function.

C. Restricted Boltzmann Machines

As stated before, the energy-based models of interest are restricted Boltzmann machines. In Figure 4, we see that RBMs have bias vectors b and c, which are related to the visible and hidden layers, respectively, as well as a weight matrix W, which relates the hidden vector to the visible vector. Assuming the RBM is using binary units, which will be used throughout this paper, the transfer function at the nodes will be sigmoidal. This means that the visible vector and hidden vector are calculated from one another with $v = sigmoid(b + W'h)$ and

$h = sigmoid(c + Wv)$ where the sigmoid function is defined as

$$sigmoid(t) = \frac{1}{1+e^{-t}}.$$

Of note here, the visible nodes are not dependent on one another nor are the hidden nodes. This makes it simple to calculate the probability of any h for any given v and vise-versa. These probabilities are given as:

$$p(h|v) = \prod_i p(h_i|v)$$

and

$$p(v|h) = \prod_j p(v_j|h).$$

Given this information, the energy function of the RBM is defined as

$$E(v,h) = -b'v - c'h - h'Wv.$$

From this the free energy function, can be derived and is defined as

$$\mathcal{F}(v) = -b'v - \sum_i log\,(1 + e^{c_i + W_i v}).$$

More information on RBMs and how to train them can be found in [6] and [7].

III. Deep Learning

As can be seen in figures 3 and 4, a trained RBM closely resembles a single layer of an artificial neural network. This allows us to stack RBMs to form a neural network. We train an RBM, which we'll call RBM1, based on our input data. Then we take all the values at the hidden layer of RBM1 and use them as the inputs to train RBM2. Then the outputs of RBM2 can be used as inputs to RBM3, and so on. This process is shown in Figure 5. This training is unsupervised, meaning that no targets outputs are given to the model. It will have information about the inputs and how they are related to one another, but they will not be able to solve any real problems.

The next step in training a deep neural network, often called "fine tuning", involves using gradient descent to train the neural network to solve a particular problem. Our problem is short-term load forecasting, so actual natural gas load values are used as target outputs and a set of features such as temperature, wind speed, day of the week, and previous loads are used as the inputs. After the supervised training step, the DNN function similarly to a large ANN.

IV. Method

In this section, some of the details of the proposed model, such as the architecture of the model and the inputs, are discussed.

A. Architecture of the DNN

Between varying the number of layers of the deep neural network and varying the number of neurons in each of those layers, the design space of a neural network is practically infinite. This is because the size of a neural network is limited only by computational power. The architecture used for both the ANN and DNN in this paper has four hidden layers with 60, 60, 60, and 12 neurons respectively.

B. Inputs to the DNN

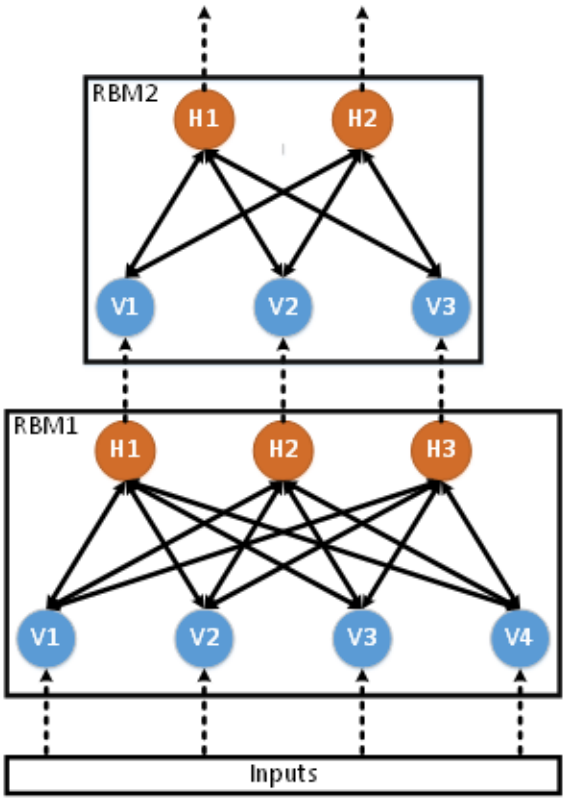

Figure 5: Graphical representation of how RBMs are trained and stacked to function as a neural network.

The inputs to the model include temperature (T) and wind speed (ws) as well as industry standards like heating degree day:

$$HDD = \max(0, T_{reference} - T),$$

And effective degree day:

$$EDD = \text{HDD}\,\frac{100+ws}{100}.$$

Multiples of each of these are created using different reference temperatures. Time based variables such as day of the week (dow) and day of the year (doy) are also used as well as more complicated time related variables such as:

$$\text{input} = \cos\left(\frac{2\pi dow}{7}\right),$$

and

$$\text{input} = \text{HDD} * \sin\left(\frac{2\pi doy}{365}\right).$$

Another important set of inputs are loads, temperatures, and HDDs from previous days. These "lagged" terms are used because in gas systems prior day temperatures and loads can have a large effect.

V. Results

This section discusses the performance of the DNN on the short-term load forecasting problems.

A. Metrics

We will use several metrics to evaluate the performance of the models. First, the models will be evaluated using root mean square error (RMSE):

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left[\hat{S}(n) - S(n)\right]^2},$$

and mean absolute percent error (MAPE):

$$MAPE = \frac{1}{N}\sum_{n=1}^{N}\frac{\left|\hat{S}(n) - S(n)\right|}{S(n)}.$$

These metrics were chosen because of the value they provide together. RMSE is a powerful metric for short term natural gas load forecasting, because it naturally places more value on days with higher load, which are more important to natural gas utilities. Unfortunately, RMSE is dependent on the magnitude of the system and cannot be used to compare the performance of the technique between different systems. MAPE, on the other hand, is not magnitude dependent and can be used to compare performance between models of different systems. Conversely, MAPE naturally places higher value on days with lower load, which are less important to natural gas utilities. These two metrics together give a fuller picture of the performance of the model than either could alone.

Another important criteria when evaluating a short term natural gas demand forecasting model is its performance on high demand and traditionally hard to forecast days. The highest demand days for natural gas are the coldest days when the heating load is the greatest. Prices are higher when the weather is colder so it's more important that a forecast be correct on these days. Other important days for forecasting natural gas are days much colder or warmer than the previous day, the first warm days after a heating season, and the first cold days leading into a heating season. A forecasting model that performs well on these days can help a local distribution company avoid penalties and having to buy gas on the spot market. As such, all the models will be evaluated on these day types individually as well as on all days.

B. The data set

The data set used in this paper come from 176 natural gas operation areas around the United States. These operational areas come from many different geographical regions including the Southwest, the Midwest, West Coast, Northeast, and Southeast and thus represent a variety of climates. The data sets also include a variety of urban, suburban, and rural areas. This diverse data set allows for more broad conclusions to be made about the performance of the models.

C. Comparing DNNs to ANNs and LR models

In this section, the DNN is compared to other common forecasters: the feedforward ANN using a Levenberg-Marquardt training algorithm and the LR model. Both the ANN and DNN outperform the LR model on most of the 176 areas. This can be seen in Figure 6, which shows the histogram of the differences in MAPE between the LR model and DNN. In this figure, positive values reflect operating areas where the DNN is better than the LR model where negative values reflect the couple of areas where the LR model is better. This graph shows that the DNN outperforms the LR model on most of the operational areas. We performed a right-tailed t-test and the p-value was found to be 4.64×10^{-35}, which supports these findings.

Likewise, the DNN outperforms the ANN on about half of the areas using MAPE and RMSE. Shown in Figure 6 are histograms of the difference between the ANN and DNN in MAPE. Although it appears that the center of the distribution is approximately zero, the right-tailed t-test shows that the mean of the distribution is greater than zero with a p-value of 1.08×10^{-4}. This means that in general, the DNN outperforms the ANN.

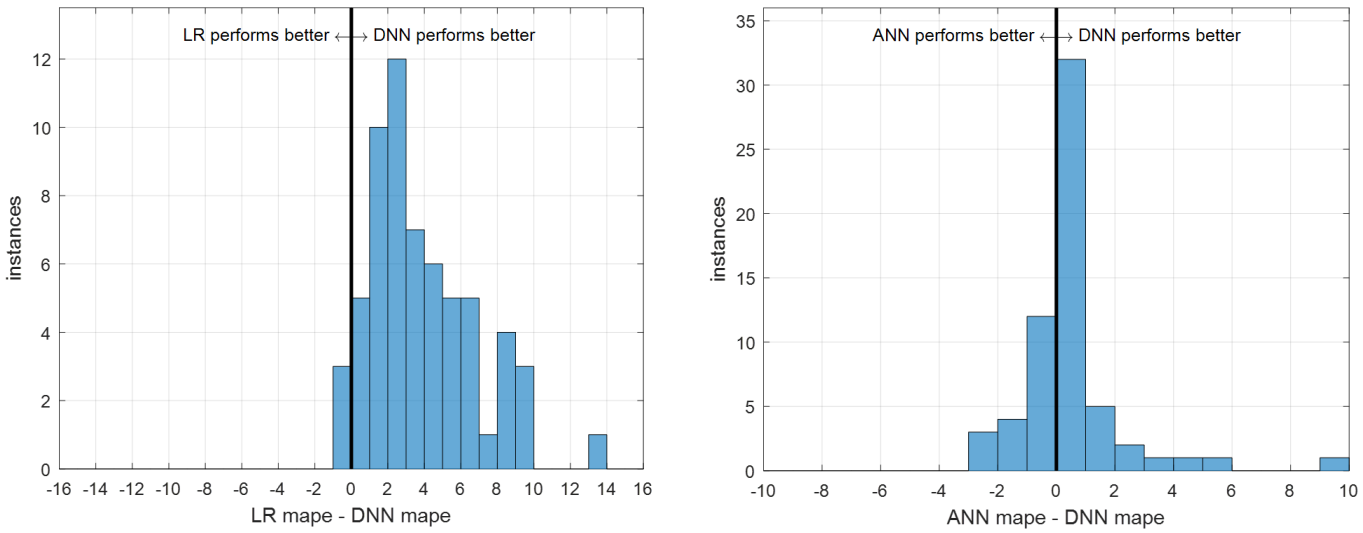

Figure 6: On the left is the Histogram of the differences in MAPEs between DNN and the LR models. Visually, we can see that DNN generally outperforms the LR model. This is reflected in the p-value from the right-tailed t-test. On the right is the Histogram of the differences in MAPEs between the DNN and ANN models. We can see that the DNN and ANN models perform similarly with some bias towards the DNN. This is reflected in a larger p-value from the right-tailed t-test.

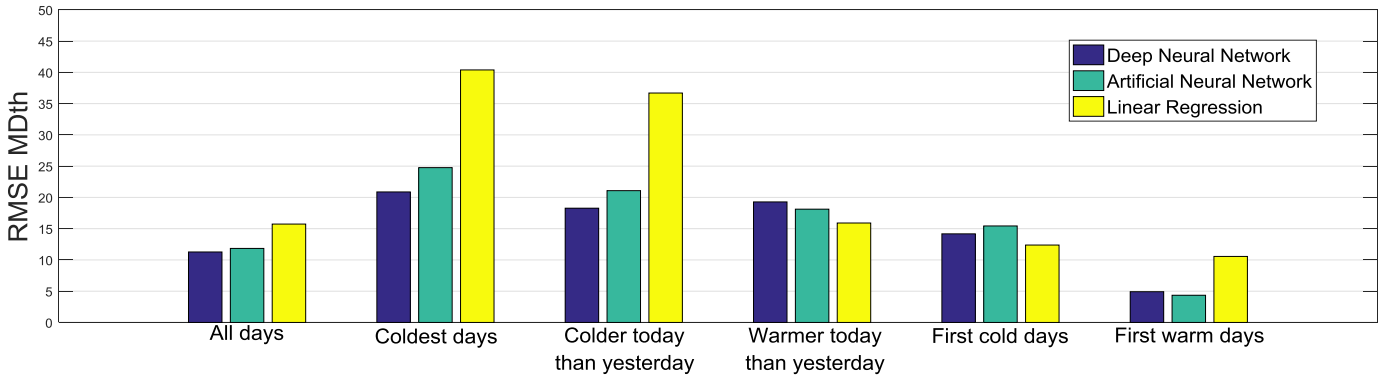

Figure 7: Shows the performance of the median operational area when comparing DNNs to ANNs on several hard to forecast days. This includes the 5% coldest days, days with the 5% greatest drop in temperature between yesterday and today, days with the 5% greatest increase in temperature between yesterday and today, and the first heating and first non-heating days of the year. RMSE values are scaled to protect the anonymity of the operational area.

We also want to compare the performance of the two models on hard to forecast days. Figure 7 shows that the DNN outperforms the ANN and LR model on all days and performs better on the days with the highest flow for an area. However, the DNN does not perform better on some of the other types of days. This tells us that despite the DNN being an effective forecaster, there are still improvements to be made in future work.

VI. Conclusion

Deep neural networks are a powerful tool for solving many machine learning problems such as speech processing and image classification. We find that, in general, the DNN outperforms other common forecasters given similar inputs. This shows that they are a viable tool to be used for time series forecasting; short-term natural gas forecasting in particular. This work is continued in [8] and [9].

VII. References

[1] S. R. Vitullo *et al*, "Mathematical models for natural gas forecasting," *Canadian Applied Mathematics Quarterly*, vol. 17, *(7)*, pp. 807-827, 2009.

[2] T. Haida and S. Muto, "Regression based peak load forecasting using a transformation technique," *IEEE Trans. Power Syst.*, vol. 9, *(4)*, pp. 1788-1794, 1994.

[3] K. Hornik, M. Stinchcombe and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, *(5)*, pp. 359-366, 1989.

[4] D. C. Park *et al*, "Electric load forecasting using an artificial neural network," *IEEE Trans. Power Syst.*, vol. 6, *(2)*, pp. 442-449, 1991.

[5] M. Längkvist, L. Karlsson and A. Loutf, "A review of unsupervised feature learning and deep learning for time-series modeling," *Pattern Recognition Letters*, vol. 42, pp. 11-24, 2014.

[6] Geoffrey E. Hinton, Simon Osindero and Yee-Whye Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, *(7)*, pp. 1527-1554, 2006.

[7] G. Hinton, "A practical guide to training restricted Boltzmann machines," *Momentum*, vol. 9, *(1)*, pp. 926, 2010.

[8] G. Merkel, R. Povinelli and R. Brown, "Deep Neural Network Regression as a Component of a Forecast Ensemble," *37th Annual International Symposium on Forecasting*, 2017.

[9] G. Merkel, "Deep Neural Networks as Time Series Forecasters of Energy Demand," *ProQuest Dissertations Publishing*, 2017.