Deep Neural Networks As Time Series Forecasters of Energy Demand

Gregory Merkel
Marquette University

DEEP NEURAL NETWORKS AS TIME SERIES FORECASTERS OF ENERGY DEMAND

By

Gregory D. Merkel, B.S.

A Thesis Submitted to the Faculty of the Graduate School,
Marquette University,
In Partial Fulfillment of the Requirements for
The Degree of Master of Science

Milwaukee, Wisconsin

August 2017

ABSTRACT
DEEP NEURAL NETWORKS AS TIME SERIES FORECASTERS OF ENERGY DEMAND

Gregory D. Merkel, B.S.

Marquette University, 2017

Short-term load forecasting is important for the day-to-day operation of natural gas utilities. Traditionally, short-term load forecasting of natural gas is done using linear regression, autoregressive integrated moving average models, and artificial neural networks. Many purchasing and operating 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. Therefore, the GasDay lab continues to explore new ways to make better forecasts.

Recently, deep neural networks (DNNs) have emerged as a powerful tool in machine learning problems. DNNs have been shown to greatly outperform traditional methods in many applications, and they have completely revolutionized some fields. Given their success in other machine learning problems, DNNs are evaluated in energy forecasting.

This thesis examines many DNN parameters in the context of the short-term load forecasting problem including architecture, input features, and use of synthetic data. The performance of the model is compared against several traditional forecast strategies, including artificial neural networks and linear regression short-term load forecasting strategies. Additionally, the DNN forecaster is evaluated as part of the GasDay ensemble.

The DNN forecaster proposed in this thesis offers an average 6.98% improvement in terms of weighted mean absolute percent error (WMAPE) when included as part of the GasDay ensemble. Finally, ideas for future work are discussed.

ACKNOWLEDGEMENTS

Gregory D. Merkel, B.S.

I would like to thank everyone who helped me reach this point. First and foremost of these is my parents Stuart and Lynn as well as my brother Peter, whose support has been crucial in my success.

Secondly, I would like to acknowledge the members of my committee, Drs. Richard Povinelli, Ronald Brown, and Cris Ababei. Their mentorship and advice helped shape this work and have been invaluable to my development as a researcher. I also thank my other mentors Thomas Quinn and Dr. George Corliss for their regular discussions, and Dr. Xin Feng for his role in convincing me to pursue a Master of Science.

Finally, I would like to acknowledge the opportunities and friendship provided to me by the GasDay lab and its participants. It has been an incredible experience working with and learning from all of them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Natural Gas Demand Forecasting

This section is an introduction to the natural gas industry, the GasDay lab at Marquette University, and the short-term load forecasting problem. It also discusses the current forecasting techniques employed by the GasDay lab including inputs and forecasting models.

1.1 Natural gas industry

Much of the information in this section can be found on the United States Energy Information Administration's web site [1]. The natural gas industry consists of three main parts; production and processing, transmission and storage, and distribution. Like many fossil fuels, natural gas (methane) is found underground usually near or with pockets of petroleum. As such, it is a common byproduct of drilling for petroleum. When natural gas is captured, it often is processed to remove higher alkanes such as propane and butane, which produce more energy when burned and can be sold at a higher price. After the natural gas has been processed, it is transported via pipelines around the country and stored either as liquid natural gas in tanks or back underground in aquifers, salt caverns, and other underground spaces. This gas is purchased by local distribution companies (LDCs) who provide natural gas to residential, commercial, and industrial consumers of natural gas. This thesis focuses on the natural gas consumed by their customers of these LDCs. Subsets of the customers of LDCs separated by geography or by municipality are referred to as operating areas. Operating areas are defined by the individual LDCs and can be as large as a state or as small as a few towns. The amount of natural gas used is

often referred to as the load and is measured in dekatherms (Dth), which is approximately the amount of energy in 1000 cubic feet of natural gas.

For LDCs, there are several uses of natural gas, but the primary use is for heating homes and business buildings. This is referred to as heatload. Heatload changes based on the outside temperature. During the winter, when outside temperatures are low, the heatload is high. When the outside temperature is high during the summer, the heatload is approximately zero. Other uses of natural gas, such as cooking, drying clothes, heating water, and other household appliances, are called baseload. Baseload is not effected by weather and generally remains constant throughout the year. However, baseload may change due to changes in the number of customers.

1.2 Marquette University's GasDay lab

GasDay at Marquette University operates as both a small business and a research laboratory. As a small business, GasDay works with 34 local distribution companies and forecasts approximately 20% of the United States' residential, commercial, and industrial natural gas consumption. As a research laboratory, GasDay develops techniques for forecasting, data cleaning, machine learning, and data science in an effort to improve the value provided to their customers. GasDay provides daily, hourly, and monthly forecasts and many other services to its customers. The main service provided by GasDay is daily forecasts for the demand of natural gas, which takes places from 10 A.M. one day to 10 A.M. Eastern time the next day. This thesis focuses on this daily short-term load forecasting problem.

1.3 Why is natural gas forecasting important?

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 or pay a penalty [2]. Given the monetary importance of quality forecasts to natural gas utilities, it is critical that the GasDay lab explore new ways to make better forecasts.

1.4 Factors in natural gas demand

As mentioned earlier, the baseload of natural gas consumption for an operating area typically changes slowly as the number of customers, or their behavior, change. Given the steady nature of baseload, most of the effort in forecasting natural gas focuses on predicting the heatload. Hence, the most important factor effecting the natural gas consumption is the weather.

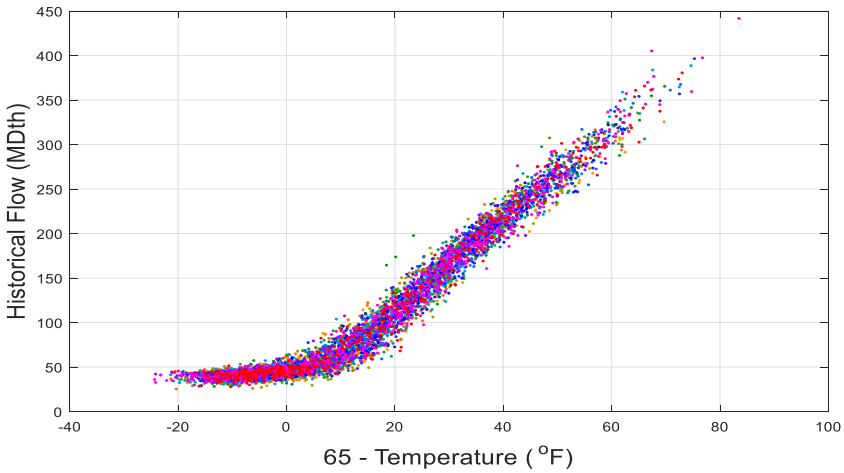

Figure 1-1: Weighted combination of several northern U.S. metropolitan operating areas.

Seen in Figure 1-1, the temperature has roughly a linear relationship with load. However, there is however a kink in the line around 65 ˚F. This occurs because at temperatures greater than 65 ˚F, home and business owners start using electricity to cool their buildings rather than use natural gas to heat them. This makes the heat load zero and leaves only the base load at temperatures greater than 65 ˚F. To handle this nonlinearity, heating degree days (HDD) are used instead of temperature,

$$HDD = \max(0, T_{ref} - T), \tag{1-1}$$

where T is the temperature and T_{ref} is the reference temperature [3]. Throughout this thesis, HDDs are written followed by their reference temperature. For instance, if the reference temperature is 65 ˚F, the heating degree day variable is written as *HDD65*.

In addition, a variant that accounts for wind called wind-adjusted heating degree day (HDDW) is used,

$$HDDW = \begin{Bmatrix} HDD\dfrac{72+ws}{80} & ws > 8 \\ HDD\dfrac{152+ws}{160} & ws < 8 \end{Bmatrix}, \tag{1-2}$$

where *ws* is the wind speed in miles per hour.

Besides *HDDW*, there are several other weather-based inputs that can be used in forecasting natural gas. One such input is cooling degree days (CDD), defined as

$$CDD = \max\left(0, T - T_{ref}\right), \tag{1-3}$$

which accounts for temperature related effects when the temperature is above the reference. As seen in Figure 1-1, these effects are not as pronounced as those when the temperature is below the reference, but they are still present. Finally, the dew point temperature (DPT) is another effective input, as it captures humidity.

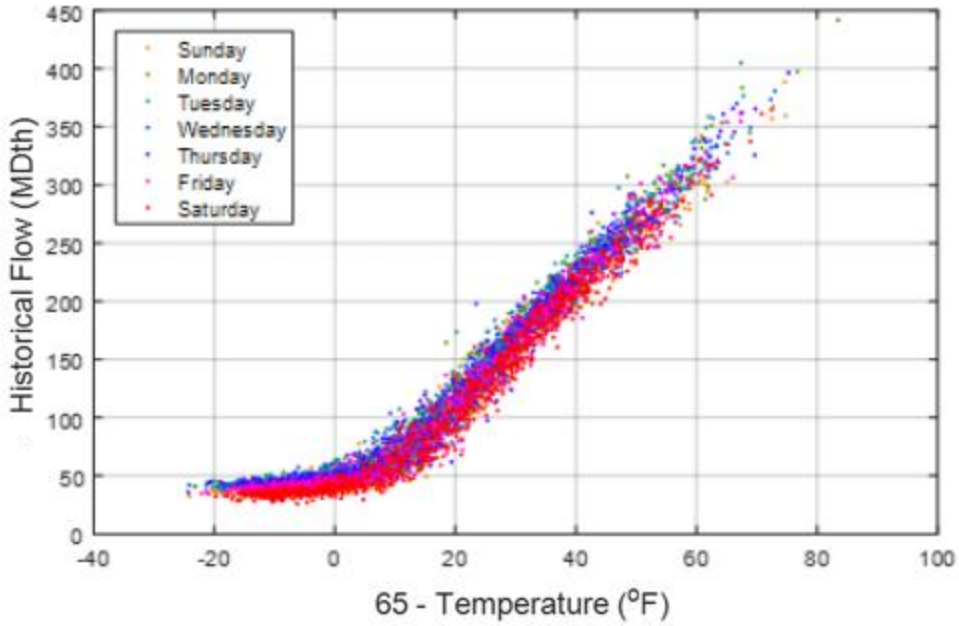

Figure 1-2: Weighted combination of several northern U.S. metropolitan operating areas colored by day of the week. This is the same data as in Figure 1-1.

In addition to weather inputs, time-related inputs also play a role in gas demand. As can be seen in Figure 1-2, the day of the week (DOW) plays a role in natural gas demand. The demand for natural gas is less on weekends (Friday-Sunday) than on weekdays (Monday-Thursday), with Wednesday generally having the highest demand, and Saturday generally having the lowest demand. Day of the year (DOY) plays a role in determining demand as well, due to changes in homeowner behaviors between seasons. For instance, 50ºF may not result in everyone turning on their furnaces in early fall, but it is likely that furnaces will be on during the winter and early spring at 50ºF.

1.5 Modeling techniques

This section gives an overview of linear models and artificial neural networks. These are two common modeling techniques available to natural gas demand forecasters. The strengths and weakness of both models also are discussed.

1.5.1 Linear models

Traditionally, short-term load forecasting of natural gas is done using multiple linear regression (LR) or autoregressive integrated moving average (ARIMA) models [4]. For customer demand *s*, forecast point *k*, and a set of *m* independent inputs like the ones discussed above, the linear regression model is defined as:

$$s_k \approx \hat{s}_k = \beta_0 + \sum_{j=1}^{m} \beta_j x_{kj} \,, \tag{1-4}$$

where β_0 through β_m, are the coefficients that represent the effect that each input has on the demand [5]. Several models can be defined using this notation. The GasDay linear regression model uses many inputs, but for the sake of explanation, a five-parameter model is used,

$$\hat{s}_k = \beta_0 + \beta_1 HDD65_k + \beta_2 HDD55_k + \beta_3 \Delta HDD_k + \beta_4 CDD65_k \,. \tag{1-5}$$

For this model, and most linear regression models for forecasting natural gas, β_0 is the base load. Similarly, the sum of β_1 and β_2 represents the heat load. Two reference temperatures are used to better model the transition between heating and non-heating days. β_3 accounts for the effect that the change in temperature between the previous day and the current day (ΔHDD) has on the current day's natural gas demand. This effect is discussed at length in [6]. Finally, β_4 allows the model to adjust to any temperature effects on demand during non-heating days. This coefficient is usually small, but not insignificant.

The five-parameter linear regression model and other linear 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 [7]. Unfortunately, gas demand contains nonlinearities. Some of these nonlinearities are easy

for a proficient forecaster to capture using an LR model. For instance, by using heating degree days as an input instead of temperature, the major nonlinearity that occurs around 65 ˚F can be accommodated. However, natural gas demand also contains many smaller nonlinearities that either cannot be captured easily with LR or ARIMA models or are difficult for forecasters to detect from the data.

1.5.2 Artificial neural network

The forecasting community's answer to the problem of nonlinearities has been to use artificial neural networks (ANNs) in place of, or in conjunction with, linear models [4], [8], [9]. Hornik et al. described them as "universal approximators," meaning that they can be used to solve almost any regression problem [8]. Artificial neural networks are based on a simplified model of neurons in the human brain.

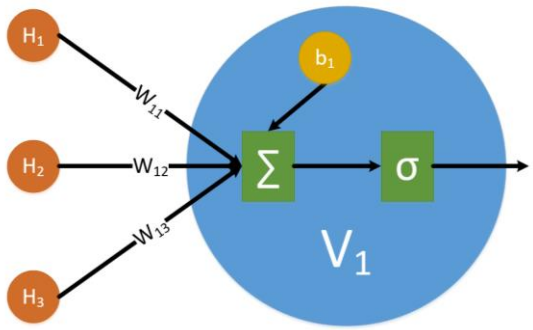

Figure 1-3: Diagram of a single node of an ANN.

Figure 1-3 shows a single node of an ANN, often called a neuron. Like the neurons in the human brain, the ANN neuron takes in information from other nodes, processes it, and sends an output based on that information. The calculation of this output Y is given as:

$$Y = \sigma(x_1 W_{11} + x_2 W_{12} + x_3 W_{13} + b_1), \tag{1-6}$$

where x is the set of inputs, w is the weights on each input, and b is a constant bias. The σ represents the transfer function. There are a variety of different transfer functions that can be used with neural networks. A collection of these nodes makes up a neural network.

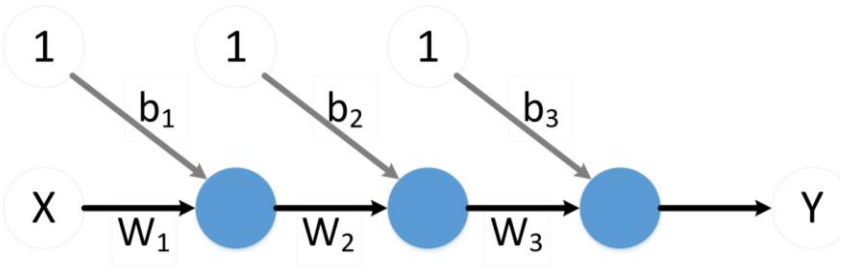

Figure 1-4: Three sequential neurons in a neural network.

Figure 1-4 shows three sequential neurons forming a simple artificial neural network. In the case of a neural network used to forecast natural gas X, on the left is a vector of the factors discussed in Section 1.4, while Y on the right is the forecast $\hat{s}$. In this case, the calculation of the forecast is:

$$\hat{s} = Y = \sigma(\sigma(\sigma(XW_1 + b_1)W_2 + b_2)W_3 + b_3). \tag{1-7}$$

For a neural network to perform well, there is probably more than one node in each layer, but Figure 1-4 is an easy way to visualize multiple layers. There are many ways to calculate the weight matrices and the biases, but the most common of these is backpropagation [10]. The training algorithm used to train the GasDay ANN is a neuron decoupled extended Kalman filter [11].

1.5.3 Ensemble forecasting

Another common technique in modeling natural gas demand, and modeling in general, is use of ensemble models. An ensemble model is used to describe any technique that combines the results of multiple forecasters to make a final forecast. For instance, the simplest ensemble is an average of the several forecasts. Even using this simple ensemble

modeling technique, a researcher is guaranteed to have a more accurate ensemble forecast than the least accurate of their individual forecasts on any given day [12]. A slightly more complicated ensemble may consist of weighting the models so that the final forecast is weighted average. For example, if a researcher were to ensemble two models, they might use weights of 0.35 and 0.65 if they know that one model generally performs better.

The GasDay ensemble is called the Dynamic Post Processor (DPP), which is an ensemble of the GasDay LR model and the GasDay ANN [13]. The DPP is useful because, in addition to selecting initial weights, the DPP adjusts those weights depending on how the two models are performing. The DPP also has an advantage over other ensemblers when forecasting natural gas demand because it can adjust to changing demand. For instance, if an operating area sees a significant increase in the number of natural gas customers, the DPP automatically adjusts the forecast upward to compensate. More information about the DPP can be found in [13] and later in this thesis.

1.6 Problem statement

Recently, the machine learning community have been successful in replacing ANNs and other nonlinear models with deep neural networks (DNN) [14]. Längkvist discusses the use of DNNs for problems ranging from video analysis and motion capture to speech and music recognition [14]. DNNs also have led to unprecedented advances in many fields such as image pattern detection [15].

As it will be described in depth later in Chapter 2, 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 of these

problems by using a restricted Boltzmann machine training algorithm to “pre-train” the model, followed by a few epochs of gradient descent [16].

The goal of this thesis is to adapt the DNN technology to short-term load forecasting of natural gas demand and to evaluate the DNNs performance as a forecaster. Little work has been done in the field of time series regression using DNNs, and almost no work has been done in the field of energy forecasting with DNNs. One notable example of literature on these subjects is Xueheng Qui et al., who claim to be the first to use DNNs for regression and time series forecasting [17]. They show promising results on three electric load demand time series and several other time series using twenty DNNs ensembled with support vector regression. The major problem with their work is that the DNNs used are quite small; the largest architecture consists of two hidden layers of 20 neurons each. Because of their small networks, Qui et al. do not take full advantage of the DNN technology.

Another example of work in this field is Busseti et al. [18], who found that deep recurrent NNs significantly outperform the other deep architectures they used for forecasting energy demand. These results are interesting but demonstrated poor performance when compared to the industry standard in energy forecasting, and they are nearly impossible to reproduce, given the information in the paper.

Some good examples of time series forecasting using DNNs include Dalto, who used them for ultra-short-term wind forecasting [19], and Kuremoto et al. [20], who used DNNs on the Competition on Artificial Time Series (CATS) benchmark. In both of these applications, DNNs outperformed neural networks trained by backpropagation. Dalto capitalized on the work of Glorot and Bengio when designing his network and showed

promising results [21]. Meanwhile, Kuremoto successfully used Kennedy's particle swarm optimization in selecting their model parameters [22]. The work most similar to this thesis is Ryu et al., who found that two different types of DNNs examined performed better on short-term load forecasting of electricity than shallow neural networks and what they called a double seasonal Holt-Winters model [23].

Given the results of these papers, DNNs should surpass ANNs in most regression problems including the short-term load forecasting of natural gas problem. This thesis explores the use of DNNs to model a natural gas system. This is done by comparing the performance of the DNN to various benchmark models and the current GasDay model. Furthermore, this thesis discusses promising methods for applying DNNs to energy demand forecasting and exploring inputs, model parameters, and transfer functions. Finally, it discusses the value of adding a DNN component to the GasDay dynamic post processor.

CHAPTER 2

Overview of Restricted Boltzmann Machines and Deep Neural Networks

This chapter discusses how deep neural networks (DNNs) work and how to train them to solve regression problems.

2.1 Restricted Boltzmann machines

Fundamental to understanding DNNs are restricted Boltzmann Machines (RBM). This section describes how they work and how they relate to DNNs. Most of the information is based on [24] and [25].

2.1.1 Energy based models

RBMs are energy-based models. This means that for any input vector x, they have an associated scalar energy based on an energy function $E(x)$. A trained energy-based model has lower energy when given inputs that are expected and high energy for inputs that are not expected [26]. For example, in a short-term load forecasting system for natural gas, if the input reserved for temperature is given some high value such as 250˚F, it is expected that a trained energy-based model would have high energy. For a simple energy-based model, the probability distribution is given as

$$p(x) = \frac{e^{-E(x)}}{Z}, \tag{2-1}$$

where

$$Z = \sum_{k} e^{-E(k)}, \tag{2-2}$$

and k represents the set of all possible inputs to the energy-based model [24]. In other words, 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 [26].

2.1.2 Energy based models with hidden layers

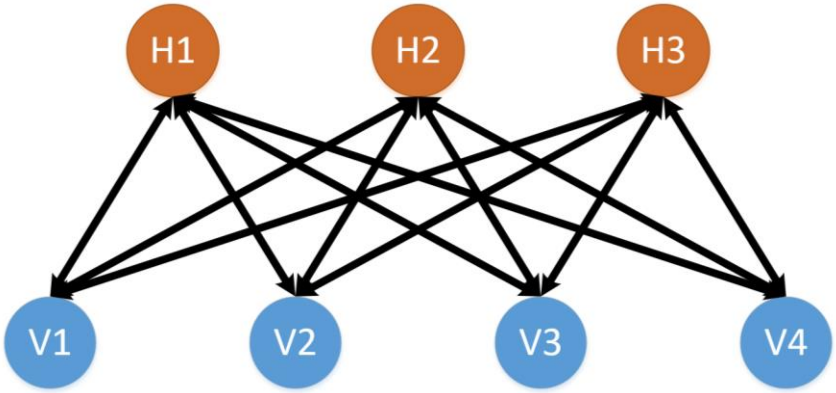

Figure 2-1: A restricted Boltzmann machine with four visible units and three hidden units. Note the similarity with a single layer of a neural network.

For more complex energy-based models like RBMs, the hidden units may be arranged as in Figure 2-1. For these models, the calculation becomes slightly more complicated as the energy associated with a visible input v must be calculated for each of the hidden units h. This probability distribution is given as [24], [25]:

$$p(v) = \sum_k p(v,h) = \sum_k \frac{e^{-E(v,h)}}{Z}, \tag{2-3}$$

where

$$Z = \sum_k e^{-E(k,h)}. \tag{2-4}$$

For the sake of simplicity in notation in later equations, this can instead be written as [24]

$$p(v) = \frac{e^{-F(v)}}{Z}, \tag{2-5}$$

where

$$F(v) = -\log \sum_{h} e^{-E(x,h)}. \tag{2-6}$$

$F(v)$ is hence referred to as the free energy function.

2.1.3 Restricted Boltzmann machines

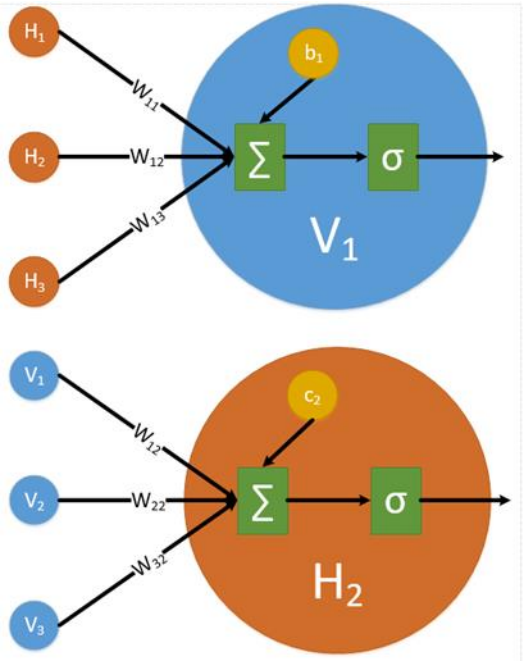

Figure 2-2: Visual representation of hidden and visible layer calculations. Note the similarity between these and the neurons of an artificial neural network.

As stated before, the energy-based models of interest are restricted Boltzmann machines (RBMs). Figure 2-2 shows the RBMs have bias vectors b and c, that are related to the visible and hidden layers, respectively, a weight matrix W which relates the hidden vector to the visible vector. Assuming that the RBM is using binary units, which is true

throughout this thesis, the transfer function at the nodes is sigmoidal. This means that the visible vector and hidden vector can be calculated from one another with $v = sigmoid(b + W'h)$ and $h = sigmoid(c + Wv)$, where the sigmoid function is [25]

$$sigmoid(t) = \frac{1}{1+e^{-t}}. \tag{2-7}$$

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 [24]

$$p(h \mid v) = \prod_i p(h_i \mid v) \tag{2-8}$$

and

$$p(v \mid h) = \prod_j p(v_j \mid h). \tag{2-9}$$

Given this information, the energy function of the RBM is [24]

$$E(v,h) = -b'v - c'h - h'Wv, \tag{2-10}$$

and the free energy function is [24]

$$F(v) = -b'v - \sum_i \log(1 + e^{c_i + W_i v}). \tag{2-11}$$

2.1.4 Training restricted Boltzmann machines

This section describes how to train a restricted Boltzmann machine for binary inputs, those scaled to be between 0 and 1, and a sigmoidal transfer function as described in Section 2.1.3. First, in a step known as the positive phase, the probability that each value in the hidden vector h is equal to 1 for a given v is calculated. This probability is [24]

$$P(h = 1 \mid v) = sigmoid(c + Wv). \tag{2-12}$$

Then, a random sample is taken from a uniform distribution from 0 to 1 for each probability, to define a vector h_p. That ends the positive phase.

In the next step, known as the negative phase, the vector h_p is used to calculate a probability that v is equal to 1, [24]

$$P(v=1|h_p)=sigmoid(b+W'h_p). \tag{2-13}$$

Again, a random sample is taken from a uniform distribution from 0 to 1 for each probability, this time to define a vector v_n. This ends the negative phase.

After this, an output is calculated from the restricted Boltzmann machine using v_n. This output is [24]

$$h_{out}=sigmoid(c+Wv_n). \tag{2-14}$$

In the final step of training, the weights and biases are updated. These are defined for some learning rate η as [24]

$$\begin{aligned} W &\leftarrow W+\eta(h_p v'-h_{out}v_n') \\ b &\leftarrow b+\eta(v-v_n) \\ c &\leftarrow c+\eta(h_p-h_{out}) \end{aligned} \quad . \tag{2-15}$$

Using this algorithm, a restricted Boltzmann machine can be trained either using a vector to train individual training points as discussed above or in batches using matrices for h and v.

2.2 Stacking restricted Boltzmann machines to make neural networks

As can be seen in Figure 2-1 and Figure 2-2, a trained RBM closely resembles a single layer of an artificial neural network. This allows us to stack RBMs to form a neural network. First, RBM1, is trained based on our input data. Then, after RBM1 is fully trained the entire input set is fed into the visible layer of RBM1 and the outputs at the

hidden layer are collected. These outputs are used as the inputs to train RBM2. This process is repeated after RBM2 is fully trained to get the inputs for RBM3, and so on. This process is shown in Figure 2-3. This training is unsupervised, meaning that no targets outputs are given to the model. It has information about the inputs and how they are related to one another, but the network is not able to solve any real problems yet.

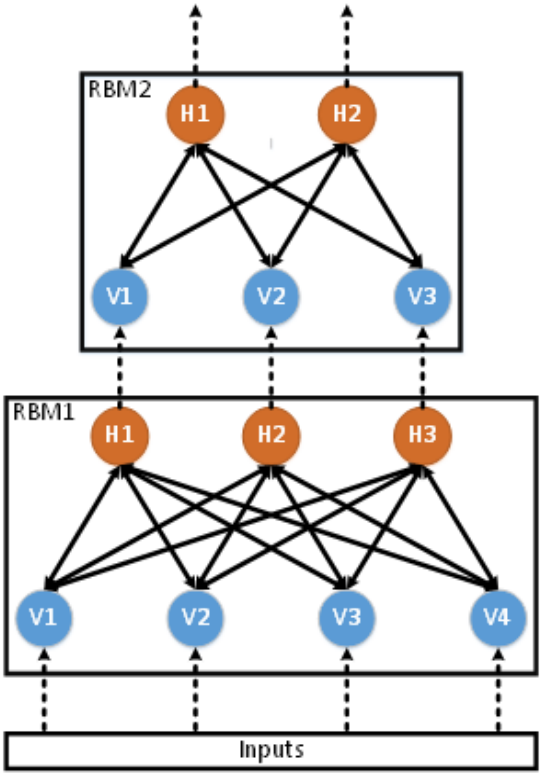

Figure 2-3: Graphical representation of how RBMs are trained and stacked to function as a neural network.

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 artificial neural network.

CHAPTER 3

Comparing Neural Network Training Algorithms

This chapter discusses the metrics, models, data, and experimental methods that are used throughout this thesis. Then, a small neural network is trained using restricted Boltzmann machine (RBM) pretraining on each of 88 operating areas and is compared with several other models. The purpose of this experiment is to give the GasDay ANN and MATLAB ANN a fair comparison by using the same relatively small architecture and set of input features. It is concluded that the small RBM neural networks do not perform as well as the GasDay ensemble. However, they do perform better than all other models examined. Finally, this chapter introduces some of the graphs and tables that are used throughout this thesis to display the results.

3.1 Metrics

This thesis uses several metrics to evaluate the performance of each model. The first of these is the root mean squared error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left[\hat{s}(n) - s(n)\right]^2}\,, \tag{3-1}$$

for a testing vector of length N, actual demand $s(n)$ and forecasted demand $\hat{s}(n)$. RMSE is a powerful metric for short-term load forecasting of natural gas because it naturally places more value on the days with higher load. These days are important, as they are when natural gas is the most expensive, which means that purchasing gas at the last minute or having bought too much gas can be costly. Unfortunately, RMSE is magnitude dependent, meaning that larger systems have larger RMSE if the percent error is constant,

which makes it a poor metric for comparing the performance of a model across different systems.

To account for the weaknesses of RMSE, this thesis also uses mean absolute percent error (MAPE):

$$\text{MAPE} = 100 \times \frac{1}{N} \sum_{n=1}^{N} \frac{\left|\hat{s}(n) - s(n)\right|}{s(n)}. \tag{3-2}$$

Unlike RMSE, MAPE is not dependent on the magnitude of the system. This means that it is more useful for comparing the performance of a method between operating areas. It does, however, put some emphasis on the lowest flow days, which, on top of being the least important days to forecast correctly, are often the easiest days to forecast. As such, MAPE is not the best metric for looking at the performance of the model across all the days in a year, but can be used to describe the performance on a particular day type.

The final error metric used in this thesis is weighted MAPE (WMAPE):

$$\text{WMAPE} = 100 \times \frac{\sum_{n=1}^{N} \left|\hat{s}(n) - s(n)\right|}{\sum_{n=1}^{N} s(n)}. \tag{3-3}$$

This error metric does not emphasize the low flow and less important days while being independent of the magnitude of the system. This means that it is the most effective error metric for comparing the performance of our methods over the course of a full year.

In addition to the error metrics discussed above, the metric of training time is evaluated for each model. This is important for the business use case. Every year the GasDay business trains and delivers approximately 6000 artificial neural networks and linear regression models to LDCs across the country. Hence, a model that takes an excessively long time to train may not be useful to GasDay. In other words, training time

is simply used to distinguish between models that can be trained in a reasonable time and those that cannot.

3.2 Training and testing data

One common problem with training any type of neural network is that there is always some amount of randomness in the results [27]. This means that it is difficult to ascertain whether a single trained model is performing well because the model parameters are good or because of probability. Hanson and Salamon mitigated this problem using cross validation [27]. This means that they trained many models on the different parts of the same set of data so that they could test their models on multiple parts of the data.

In this thesis, the problem of randomness is mitigated by having training and testing data from 88 operating areas around the United States. These operating 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 broader conclusions to be made about the performance of the models.

For each of the 88 operating areas, several models are trained using at least 10 years of data for training and 1 year for testing. The inputs to these models are the GasDay standard inputs discussed in Section 1.4. All the weather inputs in this experiment are observed weather as opposed to forecasted weather for the sake of simplicity.

3.3 Small restricted Boltzmann machine neural network

The neural network that is the focus of this chapter is a shallow neural network with two hidden layers of 12 and 4 nodes pretrained using RBMs. Each RBM is trained for 1000 epochs, and 1000 epochs of backpropagation are performed. The size and number of these layers is the same as the other neural networks to which it is compared to. Additionally, this network and all other forecasters discussed in this section are given the same inputs to ensure that a fair comparison is done between the various forecasters. Despite its small size, the RBM trained neural network is referred to as a DNN throughout this chapter to simplify notation.

3.4 Models for comparison

In this preliminary experiment, this thesis compares the performance of deep neural networks to five different models. The primary of these models is the GasDay dynamic post processor and component models discussed in Section 1.5. For the remainder of this thesis, the GasDay dynamic post processor is referred to as GDDPP. The GasDay linear regression and artificial neural network models are referred to as GDLR and GDANN, respectively. The GDLR model is tuned specifically to perform better on harder to forecast days [4]. On the other hand, the GDANN is trained using a Kalman-filter based algorithm [11] and has two hidden layers of size 12 and 4. The purpose of using these models in this experiment is to determine if the small DNN performs comparably to the current GasDay models. In addition to the models used by GasDay, this thesis also compares the DNN to models built using MATLAB tools. The first is a model built using the MATLAB neural network toolbox. This model is referred to as MLANN. This network is trained using the Levenberg-Marquardt training algorithm

and two hidden layers of sizes 12 and 4. The maximum epochs is set to 1000, but it is unlikely that this is reached because of how the Levenberg-Marquardt algorithm avoids overfitting. Similarly, this experiment also uses MATLABs built-in linear regression model. This model is referred to as MLLR. The purpose of including MLANN and MLLR is for repeatability of these experiments outside the GasDay lab, as the current GasDay models are proprietary and cannot be fully disclosed.

3.5 Results

This section gives an overview of the results of comparing the models discussed in Section 3.4. It compares the DNN to the GDDPP, each of its components, and the MATLAB built-in ANN and LR models on all 88 areas. The small DNNs perform as well as the GDDPP and better than all the other models. Then, the GDDPP and DNN are compared across unusual days, which are defined in Appendix B, for all of the areas. They perform similarly. Finally, three areas are anonymized and examined individually. One area is an example where the DNN performed better overall, one area is an example where the GDDPP performed better overall, and on the final area they performed about the same.

3.5.1 "All days" comparison

This section compares the small DNN to the GDDPP and to all of the other models.

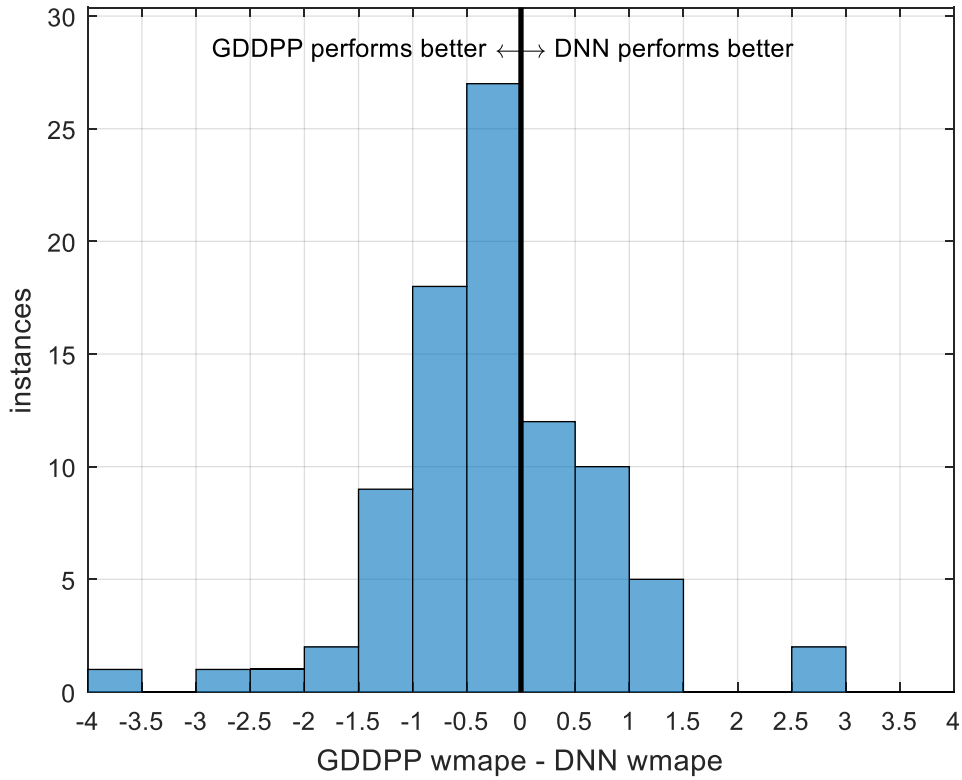

Figure 3-1: This is a histogram of the differences in WMAPE between the GDDPP and the DNN. Values on the left of the thick line at 0 indicate areas where the GDDPP performs better. Those on the right indicate areas where the DNN performs better.

Figure 3-1 shows a histogram of the differences between the weighted MAPE of the DNN forecaster over the course of a year and the weighted MAPE of the GDDPP over the course of the same year. Each instance represents one of the 88 operating areas on which the models were build. Every instance right of the center line is an example of an area where the DNN had a lower weighted MAPE than the GDDPP, and each instance to the left of the center line represents an area where the GDDPP has a lower weighted MAPE. It appears in Figure 3-1 that on average the GDDPP performs better than the

DNN. This difference is statistically significant, as a left-tailed t-test has a p-value of 0.0072. These results are relatively unsurprising, as this is a comparison between a single model and an ensemble of models.

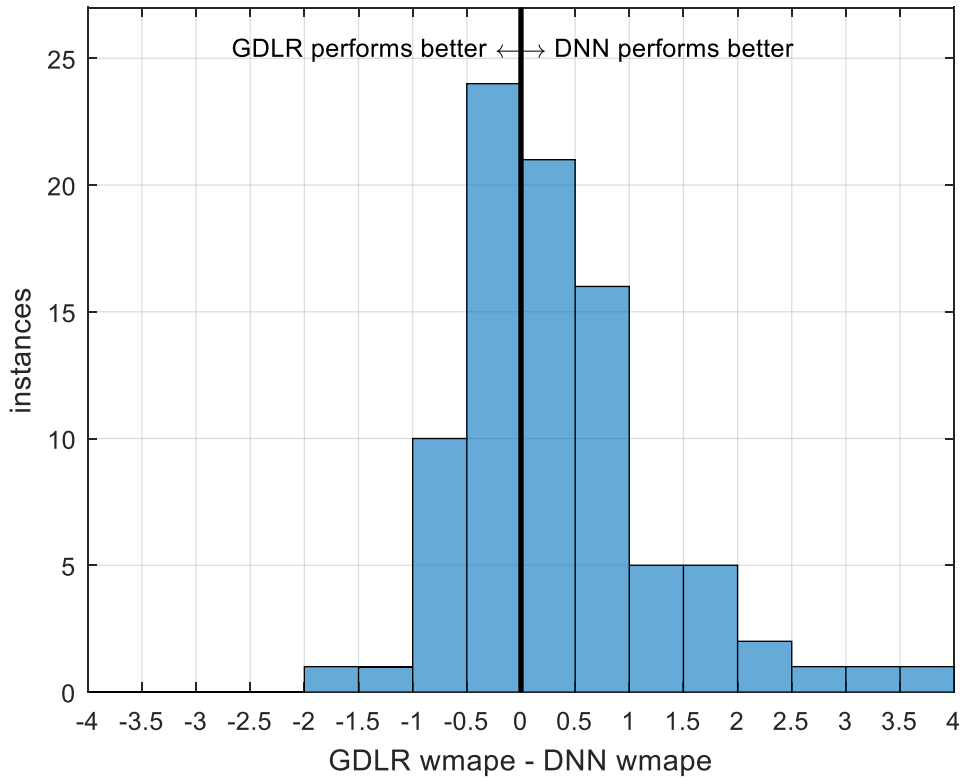

Figure 3-2: This is a histogram of the differences in WMAPE between the GDLR and the DNN. Values on the left of the thick line at 0 indicate areas where the GDLR performs better. Those on the right indicate areas where the DNN performs better.

Next, the DNN model is compared to the component models of the GDDPP. First is the GDLR. Figure 3-2 shows that the DNN performs much better than the GDLR over a majority of the areas. The majority of the areas represented on the right side of the center line and only two of those that are on the left are outside of one point of weighted MAPE. This difference is supported by a p-value of 3.24×10^{-4}. This is to be expected, as the GDLR can only capture linear trends, while the neural network can capture both linear and nonlinear trends.

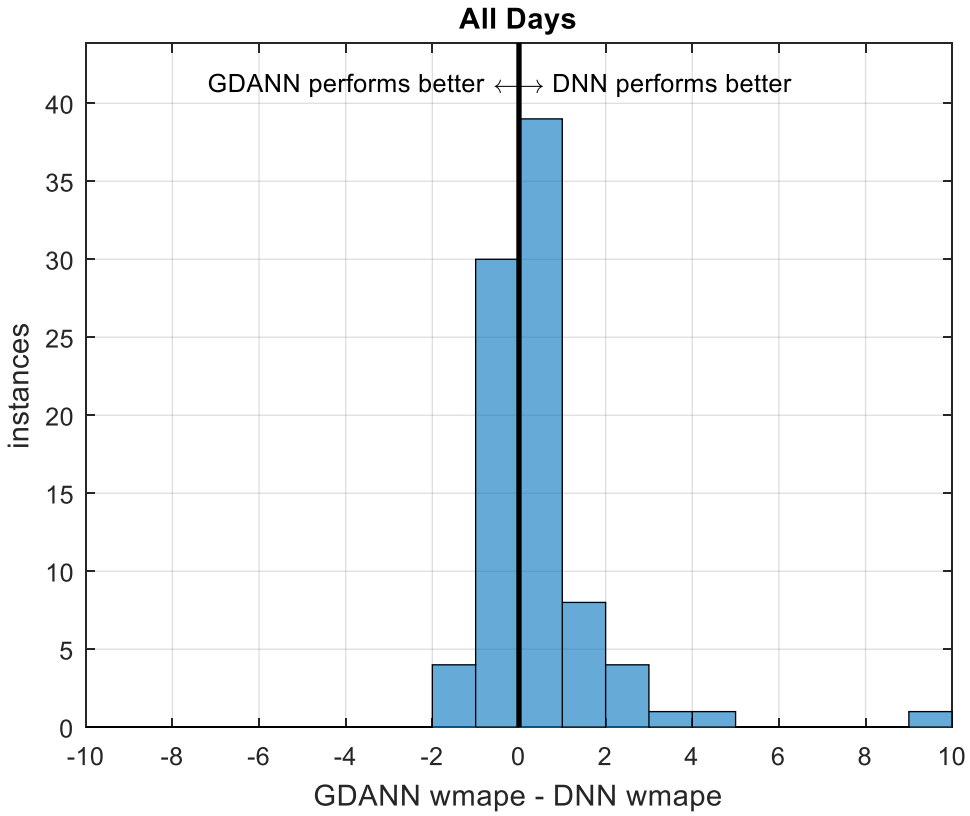

Figure 3-3: This is a histogram of the differences in WMAPE between the GDANN and the DNN. Values on the left of the thick line at 0 indicate areas where the GDANN performs better. Those on the right indicate areas where the DNN performs better.

Of greater interest is the comparison between the DNN and the GDANN. In this case, the models have identical architectures; only the training algorithm differs. These two models perform similarly, with only 19 of the 88 areas having a difference in performance greater than one point of weighted MAPE. Still, both visually in Figure 3-3 and mathematically with a p-value of 0.0018, it is apparent that the DNN performs better than the GDANN.

Finally, the DNN is compared to the MLANN and MLLR models. Figure 3-4 shows both comparisons. The MATLAB models are not as good as the DNN. This is supported by p-values, which are essentially zero and visually, as most of the instances

appear on the right side of the graphs. In particular, there is only one area on which the MLLR model performed better than the DNN forecaster.

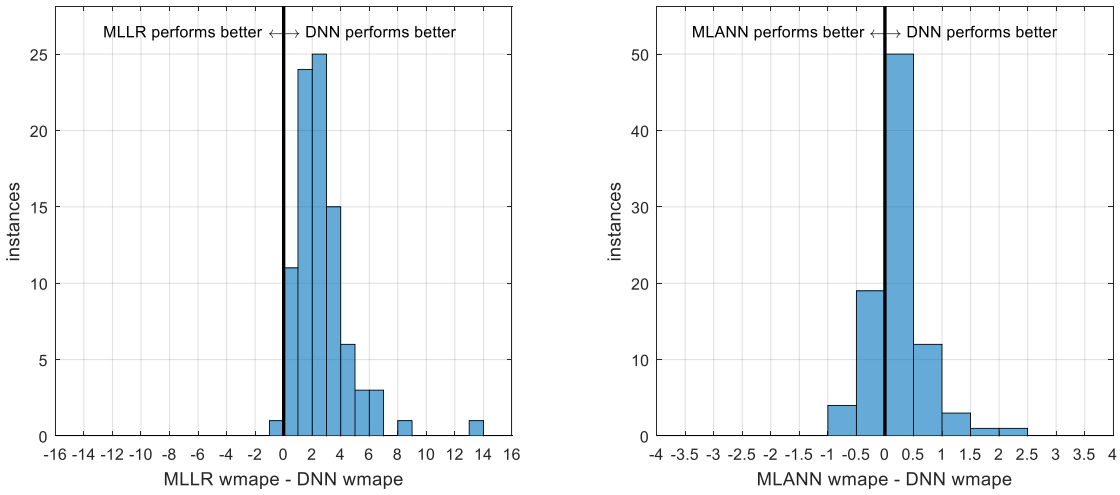

Figure 3-4: This is a histogram of the differences in WMAPE between the MLLR and the DNN and between the MLANN and DNN. Values on the left of the thick line at 0 indicate areas where the MATLAB model performs better. Those on the right indicate areas where the DNN performs better.

3.5.2 "Unusual days" comparison

Given the similar performance between the GDDPP and the small DNN on all days, it becomes important to analyze the performance of both on unusual days. Unusual days are days that tend to be harder or more important to forecast. For instance, the first heating days of a heating season or the first non-heating days after the heating season are typically hard days to forecast. Meanwhile, the coldest days of the year are not typically the most difficult days to forecast, but they tend to be important days to forecast well. More information on unusual days and how they are calculated is found in Appendix B.

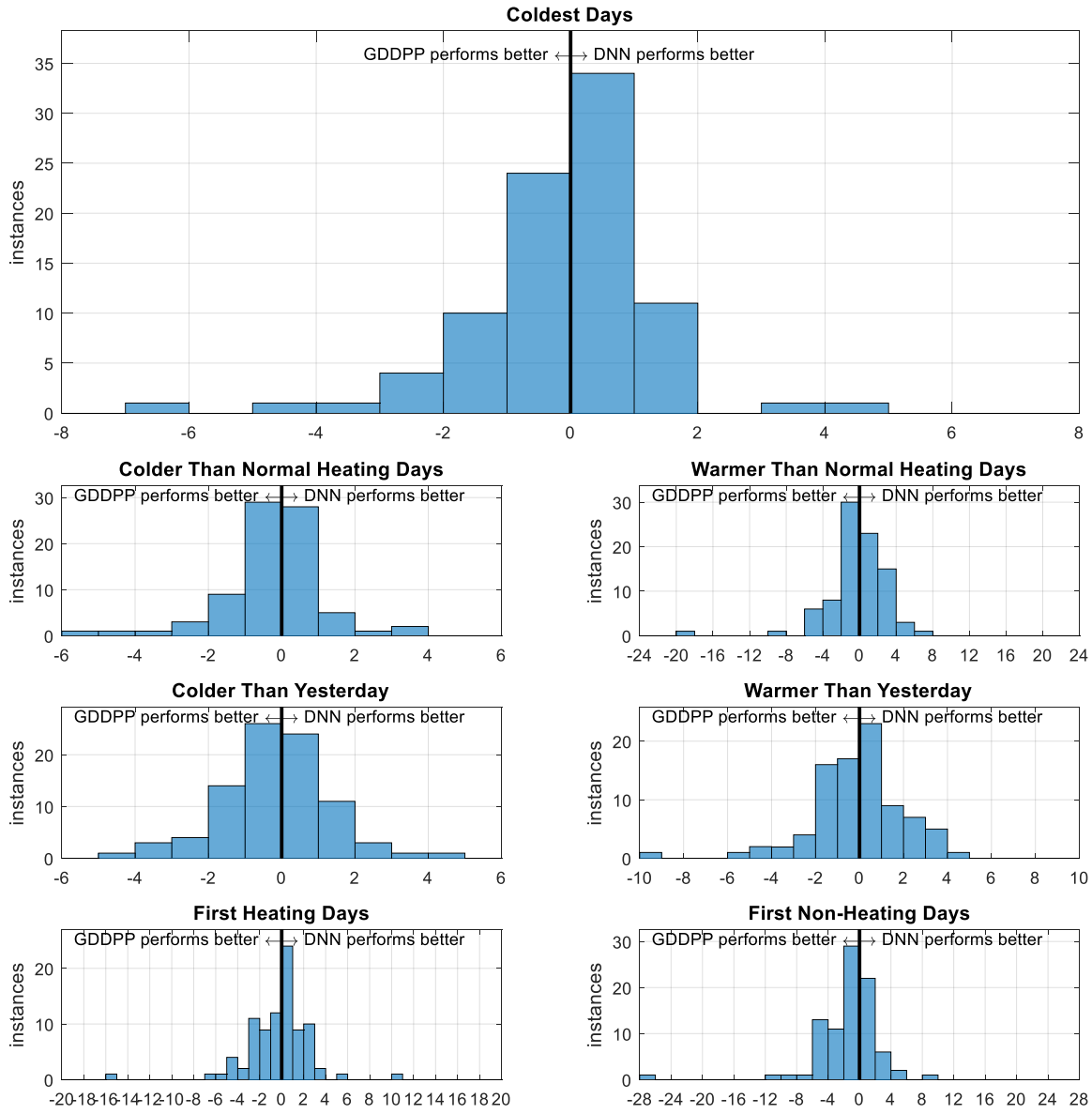

Figure 3-5: This is a histogram of the differences in WMAPE between the GDDPP and the DNN for various unusual day types. Values on the left of the thick line at 0 indicate areas where the GDDPP performs better. Those on the right indicate areas where the DNN performs better. The results of a left-tailed t-test on each of these distributions are included in Table 3-1.

Figure 3-5 shows that the GDDPP generally performs better than the DNN on all of the unusual day types, but Table 3-1 shows that the only statistically significant differences are on colder than normal heating days and the first non-heating days. This is despite the fact that when compared across all days there is a statistically significant

difference. This is a promising sign for the DNN as it performs better on the unusual days than it does on all days.

Table 3-1: Left-tailed t-test comparing the GDDPP and the DNN on each unusual day type.

Unusual Day Type	**p-value**
All Days	**0.0072**
Coldest Days	0.1668
Colder Than Normal Heating Days	**0.0427**
Warmer Than Normal Heating Days	0.2080
Colder Than Yesterday	0.1488
Warmer Than Yesterday	0.3480
First Heating Days	0.2229
First Non-Heating Days	**0.0018**

3.5.3 Individual models

In this section, a further inspection is done on some individual operating areas. These areas are chosen based on the difference between the performance of the DNN and the GDDPP. The first is the area with the greatest difference in weighted MAPE in favor of the DNN, the second is the area with the greatest difference in weighted MAPE in favor of the GDDPP, and the final area is the median area which, in this case, results in a 0.246 difference in weighted MAPE in favor of the GDDPP.

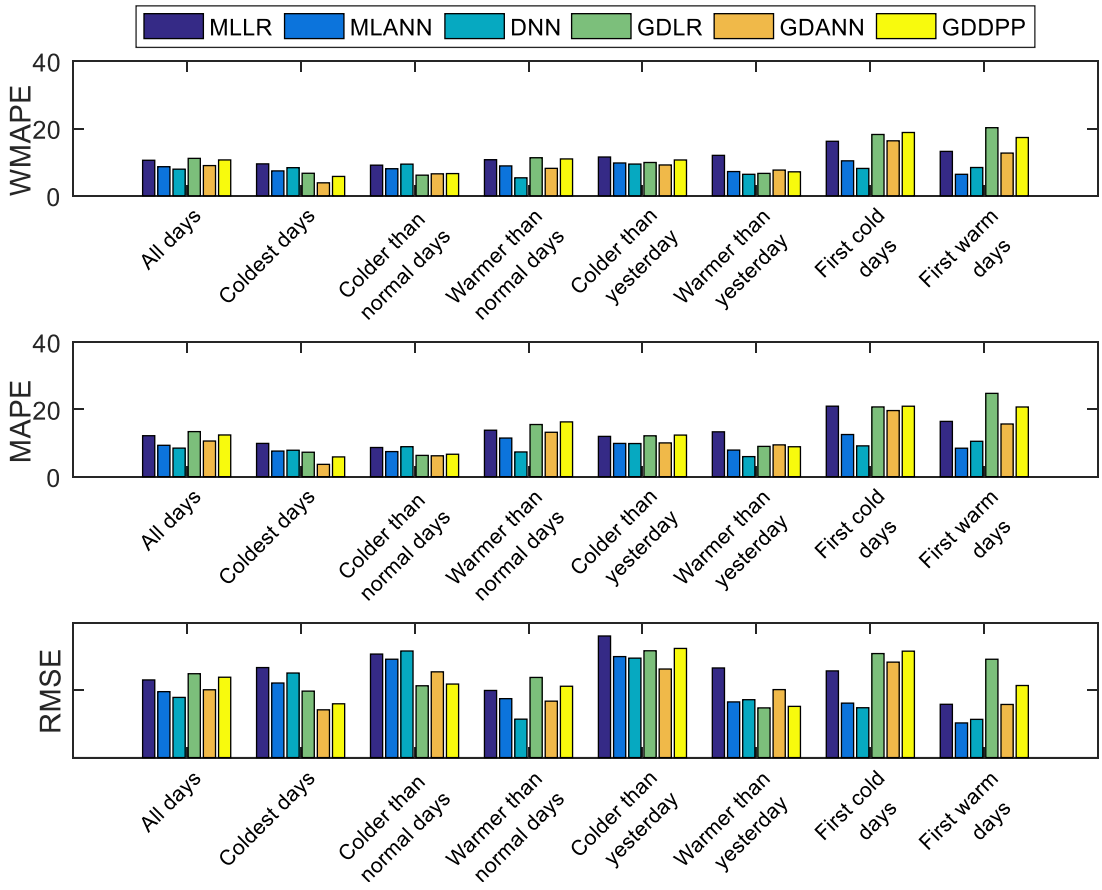

Figure 3-6: The best performing DNN when compared to GDDPP. All models are included for reference. RMSE magnitudes are removed to ensure customer anonymity.

The results for the first area, shown in Figure 3-6, illustrates a few key points, which are reiterated with each of the areas discussed in this section. First, a model which performs better when measured on all days may not perform better when evaluated on a particular day type. The example here is that the GDDPP significantly outperformed the DNN on the coldest days and on colder than normal heating days, despite the fact that the DNN performs better on almost every other metric. The other interesting thing is that the GDDPP does not always perform as well as its best component model. In this case, the GDANN performs better than the ensemble on almost every metric.

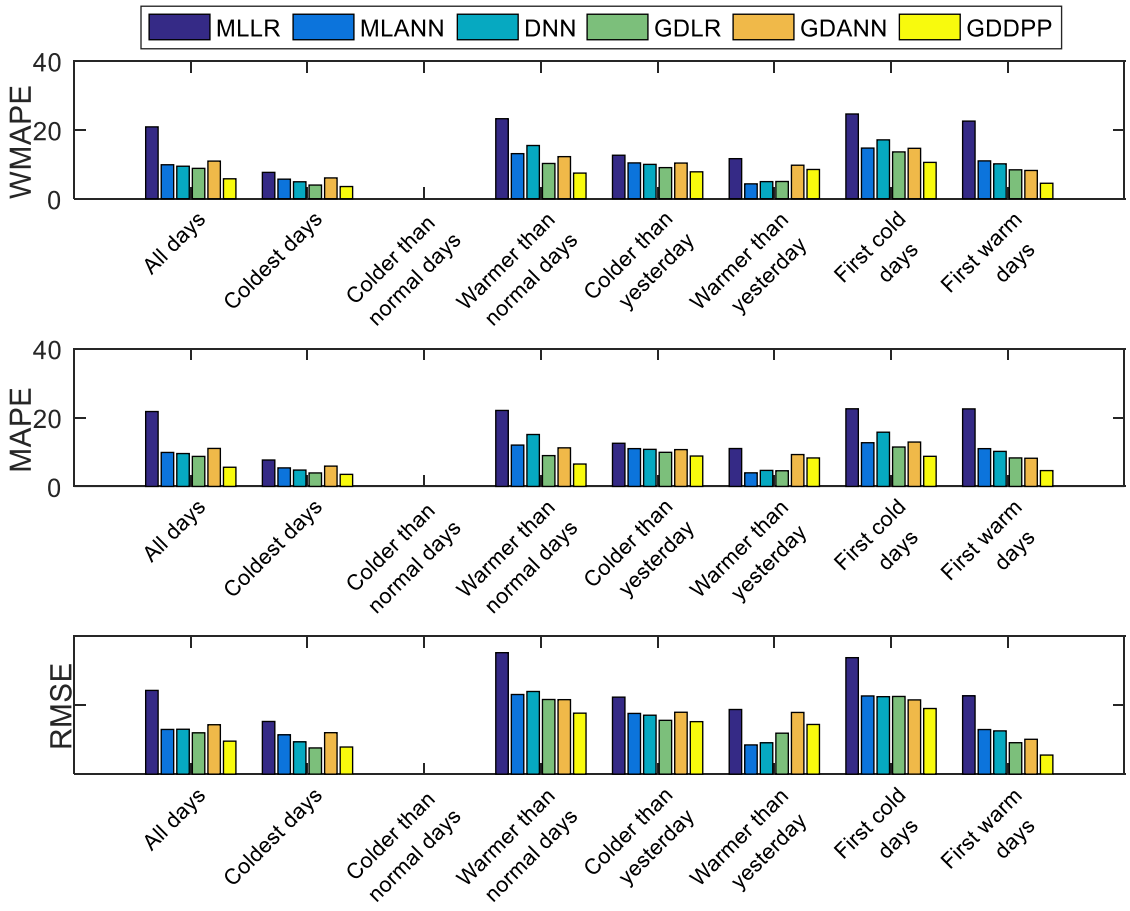

Figure 3-7: The worst performing DNN when compared to GDDPP. All models are included for reference. RMSE magnitudes are removed to ensure customer anonymity. No values are included for the colder than normal heating days as the heating season on which this area is analyzed was particularly mild so there were almost no colder than normal heating days that year. More information on how the unusual days are calculated is found in Appendix B.

As seen in Figure 3-7, despite the fact that the GDDPP performs significantly better on most metrics including all days, the DNN, as well as several other models, performs better on days which are significantly colder than the previous day. Also interesting to note are the relationships between the different neural networks. All three have certain day types on which they perform the best, although they perform relatively the same evaluated over all days. This means that there may be some benefit to ensembling multiples of these networks together.

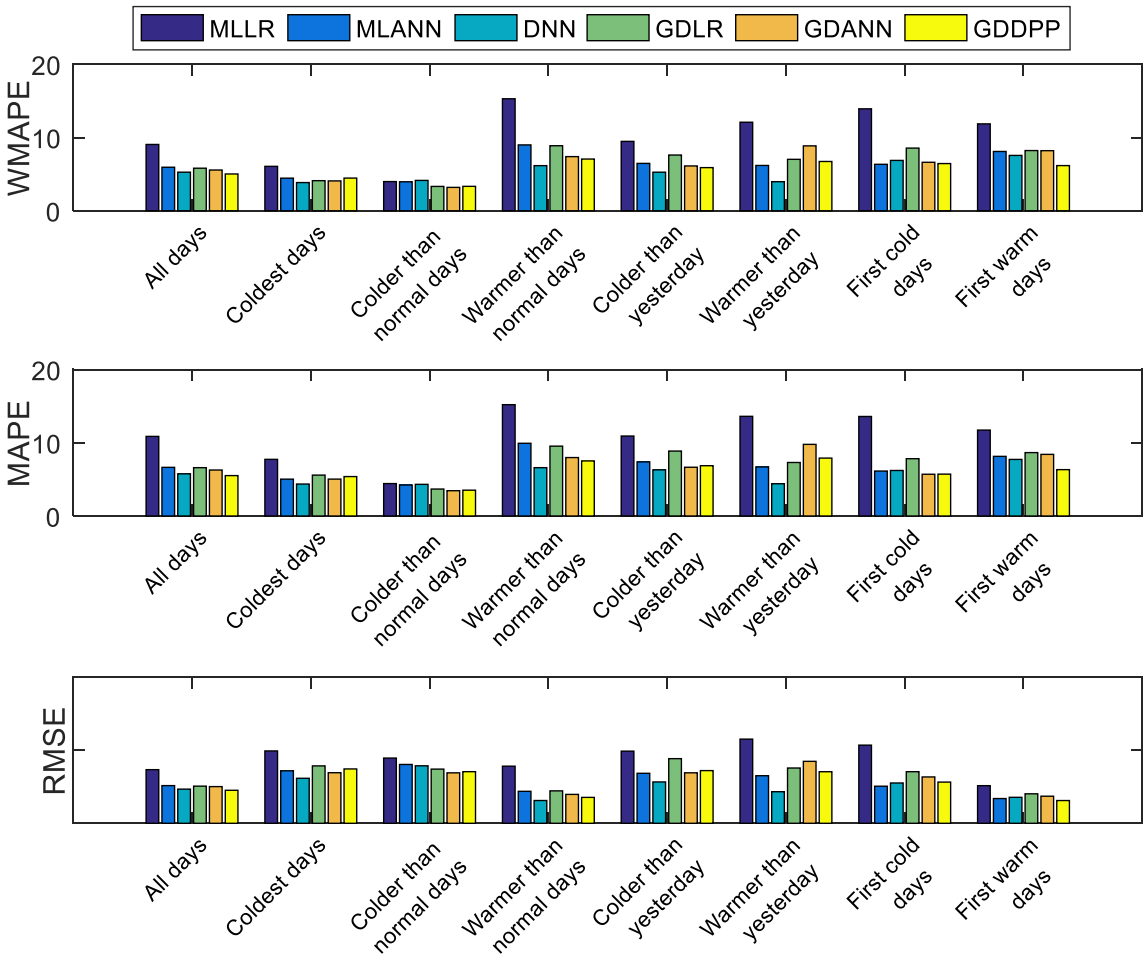

Figure 3-8: The median performing DNN when compared to GDDPP. All models are included for reference. RMSE magnitudes are removed to ensure customer anonymity.

The results on the final individual area are shown in Figure 3-8. For this area, the GDDPP performs slightly better than the DNN overall, but the only unusual day type on which there is a significant difference between their performances in favor of the GDDPP is the first non-heating days after the heating season. Important to note is that there are areas on which the opposite is true; the DNN performs better overall, but the GDDPP does better on unusual days.

3.5.4 Training time

The final metric to consider is training time. This metric is important to determine whether the models can be reasonably trained to deliver to GasDay customers. If a model is able to be trained in an amount of time less than or within the same order of magnitude as the GDANN, about 2 hours, then they are viable option as far as this metric is concerned. If a model cannot be trained in less than 2 hours, then the model may not fulfill the needs of the GasDay lab. In this case, the small DNNs can be trained in an average of 71 seconds, which is more than fast enough to meet this requirement.

3.6 Conclusions

The first and most important conclusion of Chapter 3 is that the small RBM trained neural network, in general, performs better than the other individual models discussed in this section. All of the models were trained using the same input features, data sets, and architecture and tested on the same year. This shows that it is the RBM training algorithm that is resulting in this improvement. Now analysis can be done to see if making larger DNNs, using a greater number of features, and augmenting the RBM training with surrogate data points, result in any amount of improvement over this model. The results of these experiments are in Chapter 4.

The second conclusion is that the DNN and GDDPP often performed differently on different unusual day types. This means that it is worth analyzing the performance of the GDDPP with the DNN as a component model. Additionally, it is important to do this analysis because in order for a DNN to be included in the GasDay product, it will probably be included as component of the GDDPP. This experiment is conducted in Chapter 5.

CHAPTER 4

Building a Better Deep Neural Network Forecaster

Chapter 3 showed that the deep neural network training algorithm outperformed other neural network training algorithms when the network architecture was kept the same. In this chapter, different methods for improving upon the deep neural network (DNN) model from Chapter 3 are presented. First, this chapter discusses how using additional features improves the model. Then, this chapter discusses how the number of layers affects the model. Finally, this chapter uses surrogate data, which is defined in [28], during the pretraining step to create a larger training set and to see how training on this larger data set affects model performance.

It is shown in this chapter that increasing the number of inputs has a significant positive impact on model performance, increasing the number of layers only provides additional value until around 3 or 4 layers, and that the use of a relatively small amount of surrogate data points is good but many surrogate data points are not. Finally, this chapter compares the DNN model to the GasDay ensemble and finds that the proposed model performs as well as the GasDay ensemble model.

4.1 Number of input features

In this section, networks are trained with 73 inputs, as opposed to the 26 inputs from Chapter 3. Additionally, the 73-input neural networks have two hidden layers of 60 and 12 neurons to support the increase in the number of inputs. The purpose of this experiment and the following experiment is to make deeper neural networks. As discussed extensively in Chapter 3, the “deep neural networks” used there are in fact

quite shallow. Their purpose was to directly compare the training algorithms. The 73-input networks used in this experiment are still not likely considered deep neural networks. Like the 26-input networks, they are simply shallow restricted Boltzmann machine trained networks.

The additional 47 inputs chosen for this experiment were chosen based solely on domain knowledge and what data was readily available. Further analysis can be done to find a better set of inputs for each data set, but the ability to do this is severely limited by the amount of time that it takes to train deep neural networks and the current infrastructure for making data sets. Therefore, any further analysis of inputs is beyond the scope of this thesis and is discussed in more detail in Section 6.2.

Table 4-1: Characteristics of the architectures for all of the models that are analyzed in this section. The "small 26 input' network is the same network used in Chapter 3.

Name	Number of input features	Neurons in hidden layer 1	Neurons in hidden layer 2
Small 26-input	26	12	4
Large 26-input	26	60	12
Small 73-input	73	12	4
Large 73-input	73	60	12

In this section, the four models in Table 4-1 are analyzed. The hypothesis is that both increasing the number of inputs and the numbers of neurons in each layer will improve the model. Both the small neural network with 73 inputs and the large neural network with 26 inputs are included to show the value that is gained by increasing the number of neurons and the value that is gained by increasing number of inputs separate from each other.

4.1.1 Results

First, this chapter shows a comparison between the small 26-input network in Chapter 3 and the large 73-input network. As was expected, the large 73-input network outperforms the smaller network with fewer inputs. This is supported both by Figure 4-1 and a right-tailed t-test with a p-value of 1.3×10^{-9}. The unusual day comparisons result in roughly the same information and are found in Appendix A.1.

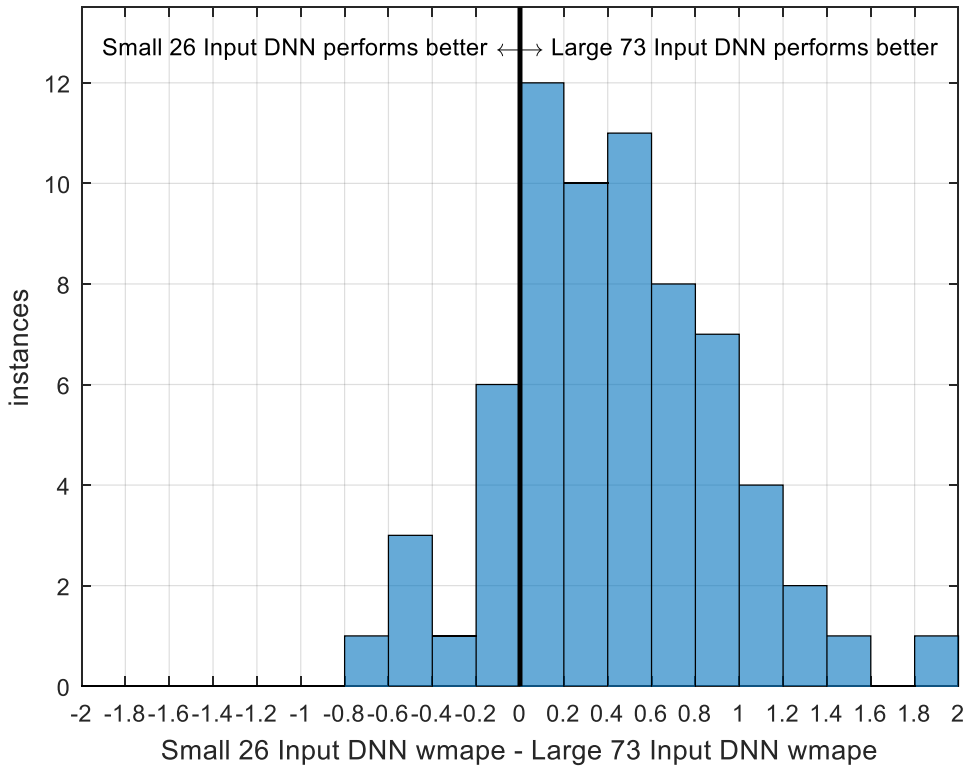

Figure 4-1: This is a histogram of the differences in WMAPE between the small 26-input DNN and the large 73-input DNN. Values on the left of the thick line at 0 indicate areas where the small 26-input DNN performs better. Those on the right indicate areas where the large 73-input DNN performs better.

More interesting results come when the two small networks are compared. The small 73-input network does not significantly outperform the small 26-input network on any measure. Shown in Figure 4-2 is a comparison between the two models. From this

figure and a right tailed p-value of 0.4328, it is shown that there is not much of a difference between the two. This is further supported by unusual day graphs in Appendix A.2.

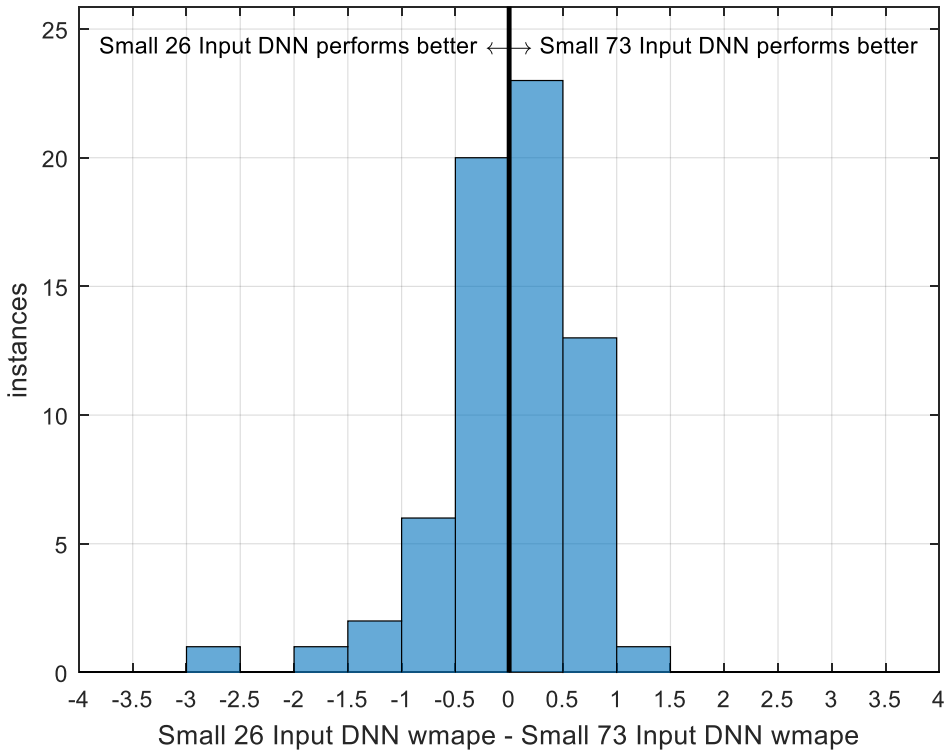

Figure 4-2: This is a histogram of the differences in WMAPE between the small 26-input DNN and the small 73-input DNN. Values on the left of the thick line at 0 indicate areas where the small 26-input DNN performs better. Those on the right indicate areas where the small 73-input DNN performs better.

This result is interesting and implies that either the additional neurons are providing the improvement and not the additional inputs or that the additional neurons are needed to take advantage of the information provided by the additional inputs. To distinguish between these possible explanations the difference between the two networks with 26 inputs must be examined. As is seen in Figure 4-3 and supported by a p-value of 2.3×10^{-2}, the large 26-input DNN performs significantly better than the small 26-input DNN, but not as much better as the large 73-input network.

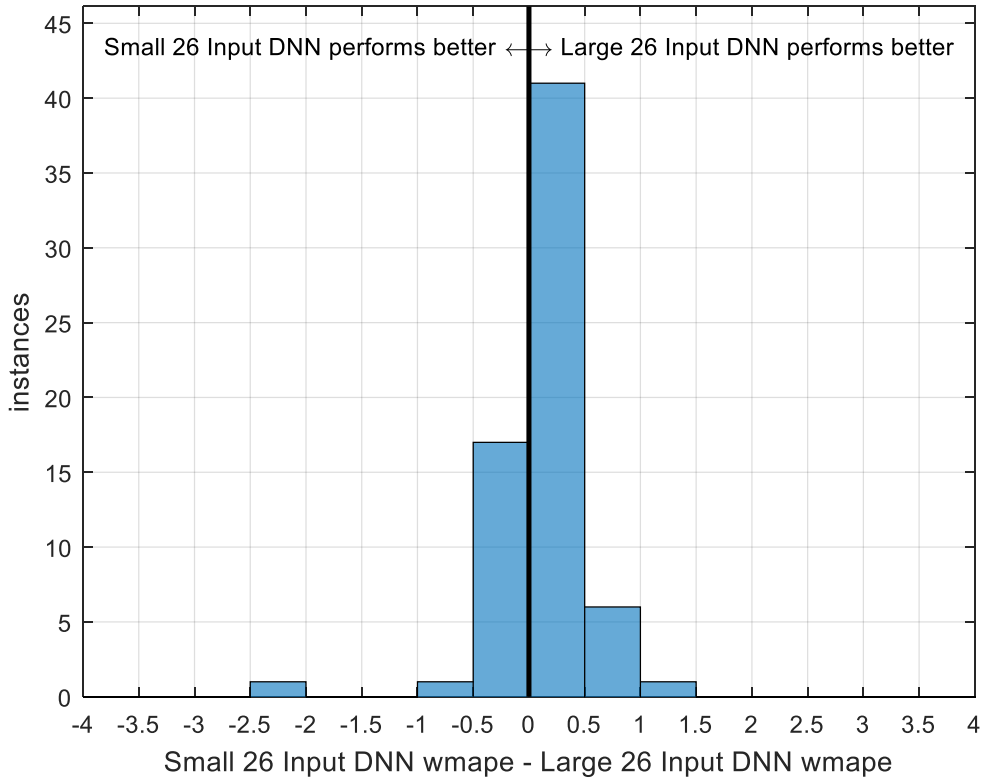

Figure 4-3: This is a histogram of the differences in WMAPE between the small 26-input DNN and the large 26-input DNN. Values on the left of the thick line at 0 indicate areas where the small 26-input DNN performs better. Those on the right indicate areas where the large 26-input DNN performs better. Similar graphs for unusual days can be found in Appendix A.3.

From this information two conclusions are drawn. First, it is concluded that adding additional inputs without increasing the size of the network does not guarantee to improve model performance. Secondly, it is concluded that increasing model width can improve the results. There is probably a limit to how wide the network can be made, but further research needs to be done to find that limit.

4.2 Network size

This section discusses how much the number of layers in the network effects the performance of the model. In theory, additional layers result in diminishing marginal returns. It is expected that eventually more layers result in a decrease in performance as

the model begins to overfit the training data set [29], [30]. In this section, the seven networks in Table 4-2 are analyzed. As can be seen, each network increments the number of 60 neuron layers to reduce any influence that layer size might have on the network performance. These seven networks are each be tested on 88 different operating areas.

Table 4-2: Characteristics of the architectures for all models in this section. The 2 Layer network is the same as the "large 73-input" network in Section 3.

Name	Number of neurons in each hidden layer	Number of inputs
1 Layer	12	73
2 Layer	60, 12	73
3 Layer	60, 60, 12	73
4 Layer	60, 60, 60, 12	73
5 Layer	60, 60, 60, 60, 12	73
6 Layer	60, 60, 60, 60, 60, 12	73
7 Layer	60, 60, 60, 60, 60, 60, 12	73

4.2.1 Results

In this results section, the histogram of differences is not used to make the comparison. The histograms are an effective way to compare the performance of two models over many areas, but are not effective for showing trends as a parameter, such as number of layers, is incremented. As such, a box plot of the WMAPEs for each of the seven networks are used. This is shown in Figure 4-4. From this it is seen that the 1, 6, and 7 layer models do not perform as well as the 2, 3, 4, and 5 layer models. This is supported by the p-values in Table 4-3. Table 4-3 also shows additional information that

cannot be gleaned from Figure 4-4. In particular, it shows that the 3 and 4 layer models perform better than the 2 and 5 layer models.

Table 4-3: Right-tailed t-test results for each of seven networks compared to each other network. Bolded values indicate that the model in the column significantly outperforms the model in the row.

Right →	1 Layer	2 Layer	3 Layer	4 Layer	5 Layer	6 Layer	7 Layer
1 Layer		**4.6×10^{-9}**	**1.4×10^{-11}**	**1.2×10^{-10}**	**3.8×10^{-9}**	**2.8×10^{-8}**	**1.5×10^{-6}**
2 Layer	1		0.061	0.11	0.70	0.87	0.93
3 Layer	1	0.94		0.56	0.99	0.99	1
4 Layer	1	0.89	0.44		0.99	0.99	1
5 Layer	1	0.30	**0.0064**	**0.0056**		0.79	0.95
6 Layer	1	0.12	**0.0045**	**0.0037**	0.21		0.82
7 Layer	1	0.066	**0.0013**	**6.9×10^{-4}**	**0.047**	0.17	

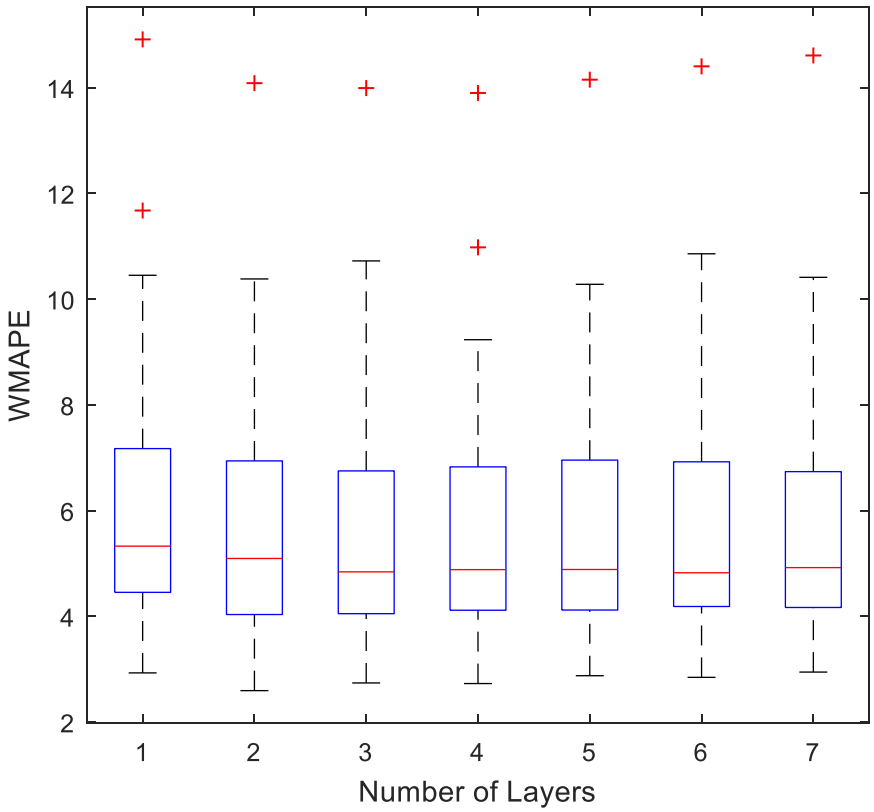

Figure 4-4: Boxplots of all the WMAPEs for each of the seven different models. The outliers at the top are small magnitude areas that are difficult to forecast for various reasons.

This trend of the 3 and 4 layer networks performing best continues for almost every metric. Given this information, it is concluded that 3 or 4 layers provide the best forecasting neural networks.

4.3 Surrogate data

One common problem for training DNNs and other complex machine learning models is a lack of data or imbalanced data, where one classification of data or region in regression problems is underrepresented. A substantial amount of work has been done in this area. Chawla et al. proposed a synthetic minority over-sampling technique (SMOTE), which has proven to effectively deal with the creating points between each of the sparse

minority sets and their k-nearest neighbors [31]. Another technique used for generating synthetic data was proposed by Goodfellow et al [32]. It involves training two neural networks against one another. One network generates synthetic data and while another tries to determine if data is synthetic or not. This creates two useful neural networks, but the relevant one here is the generator.

Fortunately, the GasDay lab has no shortage of real data, so there is no need to rely on the techniques used by [31] and [32]. The problem is that the GasDay Lab only has a couple thousand points for each area, which is sufficient but not ideal for training large networks, as exemplified by previous sections. The solution to this problem in this thesis is to generate additional training inputs by surrogate data, which is simply transforming one "donor" data set to look like another "target" data set [28]. This allows for an increase in the number of unique points in the target data set, while still using real data.

Table 4-4: Characteristics of the architectures for all models in this section. The "no surrogates" network is the same as the 5 layer network in Section 4.2.

Name	**Architecture**	**Number of inputs**	**Number of Surrogates**
No Surrogates	60, 60, 60, 60, 12	73	0
40k Surrogates	60, 60, 60, 60, 12	73	40,000
500k Surrogates	60, 60, 60, 60, 12	73	500,000

In this experiment, three networks with varying amounts of surrogate data are compared. These models are described in Table 4-4. Only three different amounts of surrogate data are used because of the long training time required. The networks trained on no surrogates took around 4.5 minutes to train, the 40k surrogate networks took

around 43 minutes to train, and the 500k surrogate networks took around 7.5 hours to train each of 79 networks. These training times have a roughly linear relationship. This make sense, as the training algorithm described in Section 2.1.4 is roughly O(n) with respect to number of training samples.

Additionally, the networks used in this section have 5 input layers as opposed to the 4 input layer networks that were determined to be best in the previous section as the experiments were conducted concurrently. The motivation for using a slightly larger than prescribed network in this section is that it may be able to better represent the additional information from the surrogate data points.

4.3.1 Results

First, the network with 40k surrogates is compared to the network with no surrogates. It is found that that the 40k surrogate model does not perform significantly better than the network with no surrogates with a right-tailed t-test resulting in a p-value of 0.20. The histogram in Figure 4-5 shows just how close the performance is, as all the differences have magnitudes less than 1.

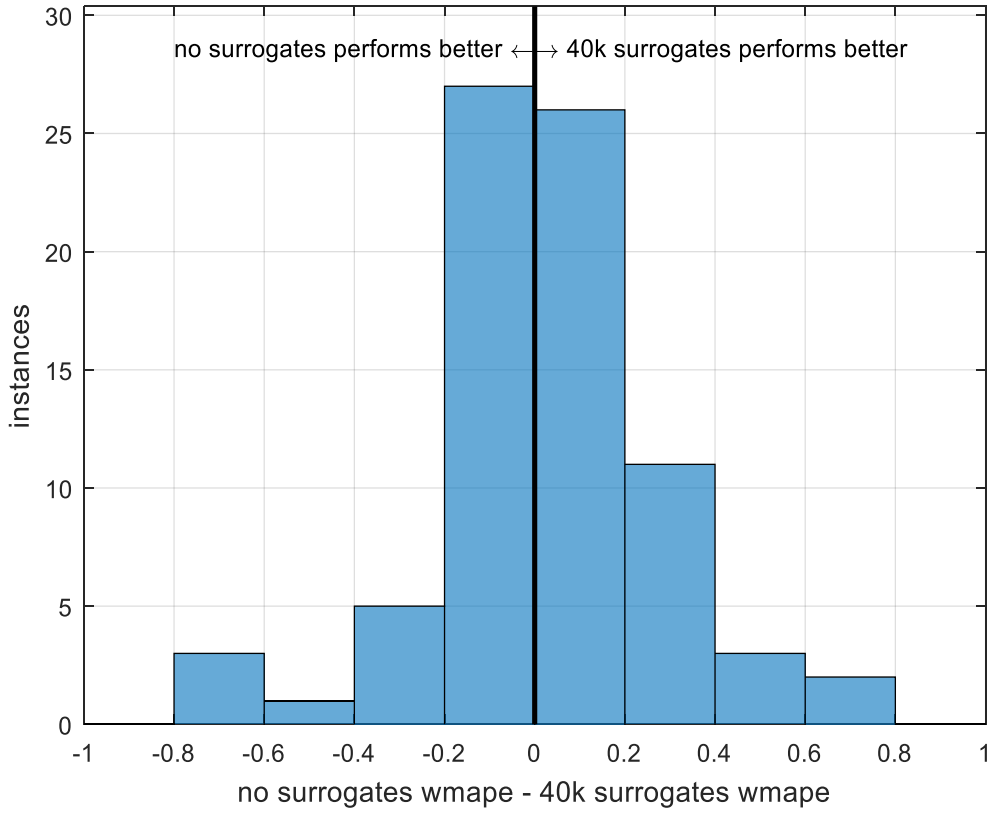

Figure 4-5: This is a histogram of the differences in WMAPE between the 40k surrogate DNN and the zero surrogate DNN. Values on the left of the thick line at 0 indicate areas where the zero surrogate DNN performs better. Those on the right indicate areas where the 40k surrogate DNN performs better.

More interesting is what happens on the unusual days. As described in [28], the surrogate data points are generated specifically to emphasize unusual days. In other words, the use of surrogate data is not expected to improve the performance of the model on all days. Instead, an improvement on the unusual day types is expected, particularly on the coldest days.

Table 4-5: Right-tailed t-test comparing the network trained on zero surrogates to the network trained on 40,000 surrogates on each unusual day type. Values less than 0.05 indicate unusual day types on which the network trained on 40,000 surrogates performs significantly better. Histograms for each of these values are included in Appendix A.4.

Unusual Day Type	**p-value**
All Days	0.1982
Coldest Days	**0.0240**
Colder Than Normal Heating Days	**0.0241**
Warmer Than Normal Heating Days	0.2884
Windiest Heating Days	0.2024
First Heating Days	0.9013
First Non-Heating Days	0.8972

Table 4-5 shows the results of the t-test comparing the two networks for each unusual day type. As expected, on the coldest days the network trained on 40,000 surrogates performs better. However, what is unexpected is that the model would perform this much worse on the first heating and non-heating days. Further research should be done on the impact of surrogate data on model performance outside of bitter cold days, but that is beyond the scope of this thesis.

Now that the network trained on 40,000 surrogates has been determined to perform better than the network trained with zero surrogates, with the exception of first heating and non-heating days, an analysis can be done to determine if increasing to 500,000 surrogates is worth the additional 7 hours of training. As can be seen in Figure 4-6, the network trained on 40,000 surrogates performs better than the network trained on

500,000 surrogates. This conclusion is supported by a right-tailed t-test that results in a p-value of 0.020. This is further supported by the results on unusual days. Table 4-6 shows that on each unusual days metric, the model trained on 40,000 surrogates performs better.

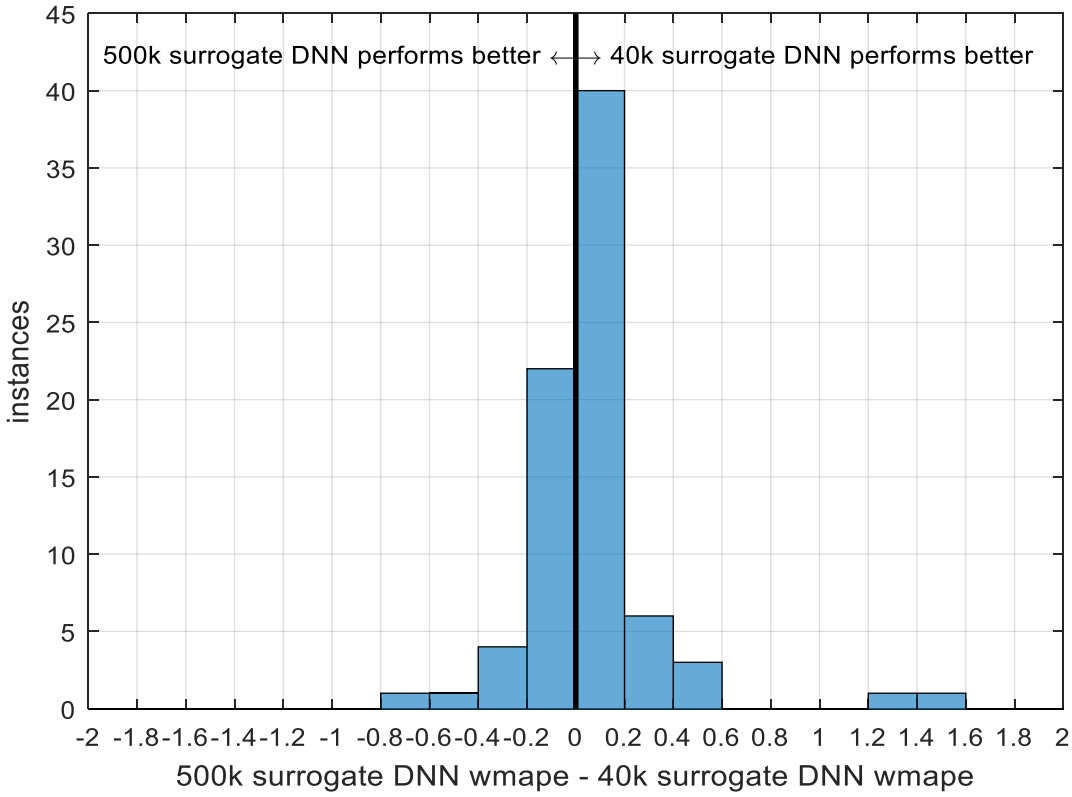

Figure 4-6: This is a histogram of the differences in WMAPE between the 40k surrogate DNN and the 500k surrogate DNN. Values on the left of the thick line at 0 indicate areas where the 500k surrogate DNN performs better. Those on the right indicate areas where the 40k surrogate DNN performs better.

Table 4-6: Right-tailed t-test comparing the network trained on 500,000 surrogates to the network trained on 40,000 surrogates on each unusual day type. Values less than 0.05 indicate unusual day types on which the network trained on 40,000 surrogates performs significantly better. Histograms for each of these values are included in Appendix A.5.

Unusual Day Type	**p-value**
All Days	**0.0196**
Coldest Days	**0.0064**
Colder Than Normal Heating Days	**0.0340**
Warmer Than Normal Heating Days	0.0770
Windiest Heating Days	0.0524
First Heating Days	**0.0275**
First Non-Heating Days	**0.0186**

The first conclusion drawn from this section is that too many surrogates take away from model performance as exemplified by the network trained on 500,000 surrogates. The second conclusion drawn is that using surrogate data results in significantly better performance on the coldest days, but also sacrifices some performance on the shoulder months. Another factor to take into account is training time. The 43 minutes that it takes to train the networks with 40,000 surrogates is still acceptable but it is still much longer than the 7 minutes that it takes without surrogates to train.

4.4 Final proposed deep neural network

In this section, a final network is proposed based on the results of this chapter. This network is shown to perform better than the GasDay ensemble.

4.4.1 Architecture of the proposed network

In Section 3, it is determined that the best input set this thesis examined was the 73-input set and that widening the network helped better use the expanded input set. Therefore, that input set is used in the final model. In Section 4.3, it is shown that using some surrogates can be beneficial, but using too many may result in loss of performance. Additionally, it was shown that the use of surrogate data probably improves performance of the model on the coldest days but also probably hurts performance on shoulder months. Given the longer training times, the additional infrastructure needed to use surrogate data with DNNs in production, and uncertainty around the tradeoff between shoulder months and coldest days, two networks are used in the final sections of this paper; one with 40,000 surrogate data points during the pretraining step and one that does not use surrogate data points. Finally, in Section 4.2, it is determined that either 3 or 4 layer networks perform the best. Thus, both of the proposed networks have 4 layers. For quick reference, both networks are described in Table 4-7.

Table 4-7: Characteristics of the architectures for all models in this section with the exception of the GasDay ensemble. The "no surrogates" network is the same as the 4 layer network in Section 4.2.

Name	Architecture	Number of inputs	Number of Surrogates
No Surrogates	60, 60, 60, 12	73	0
40k Surrogates	60, 60, 60, 12	73	40,000

4.4.2 Comparing the proposed networks to the GasDay ensemble

As can be seen in Figure 4-7, the proposed deep neural network model with no surrogate data performs similarly to the GasDay ensemble, GDDPP. Visually, it can be

seen that a slight edge is given to the DNN without surrogates. A right tailed t-test was performed on these differences and resulted in a p-value of 0.24, so any difference between the distributions is not significant. Given how close these distributions are to one another, it can be concluded that the proposed deep neural network performs well.

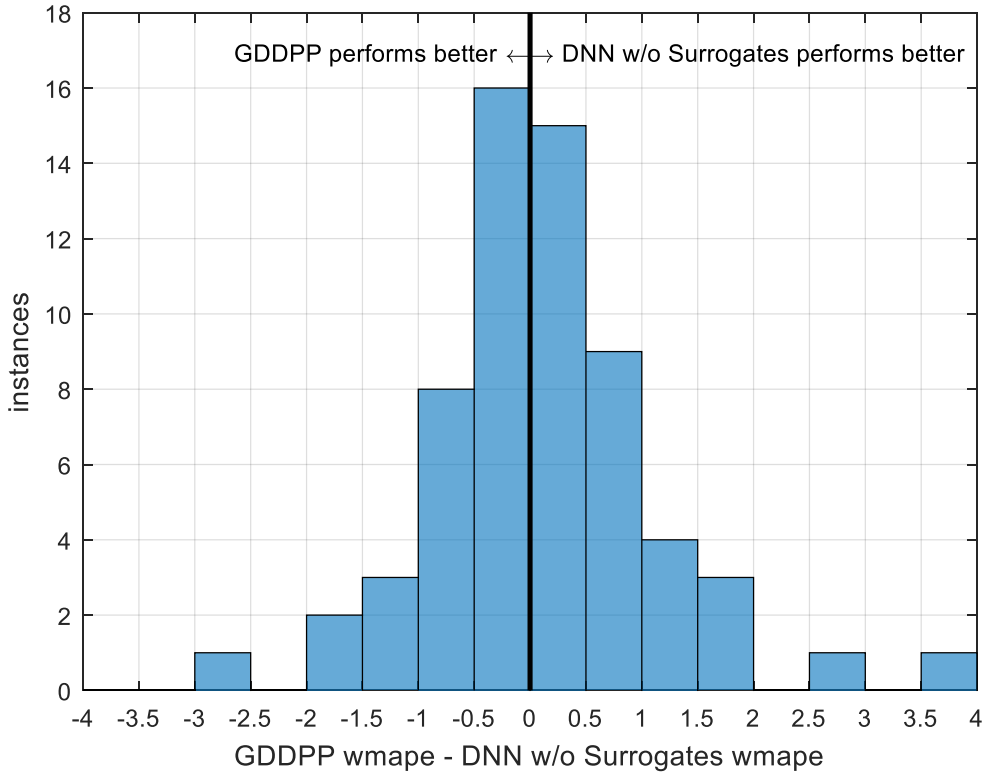

Figure 4-7: This is a histogram of the differences in WMAPE between the proposed DNN without surrogates and the GasDay ensemble. Values on the left of the thick line at 0 indicate areas where the GasDay ensemble performs better. Those on the right indicate areas where the proposed DNN without surrogates performs better.

Next, this chapter compares the DNN with surrogates to the GDDPP. This is shown in Figure 4-8. Interestingly, this DNN does not appear to perform as well compared to the GDDPP as the DNN without surrogates.

A comparison between the two DNN models is shown in Figure 4-9. Additionally, a right tailed t-test performed on the difference in performance between the

two DNNs results in a p-value of 0.0019, meaning that the DNN without surrogates performed significantly better.

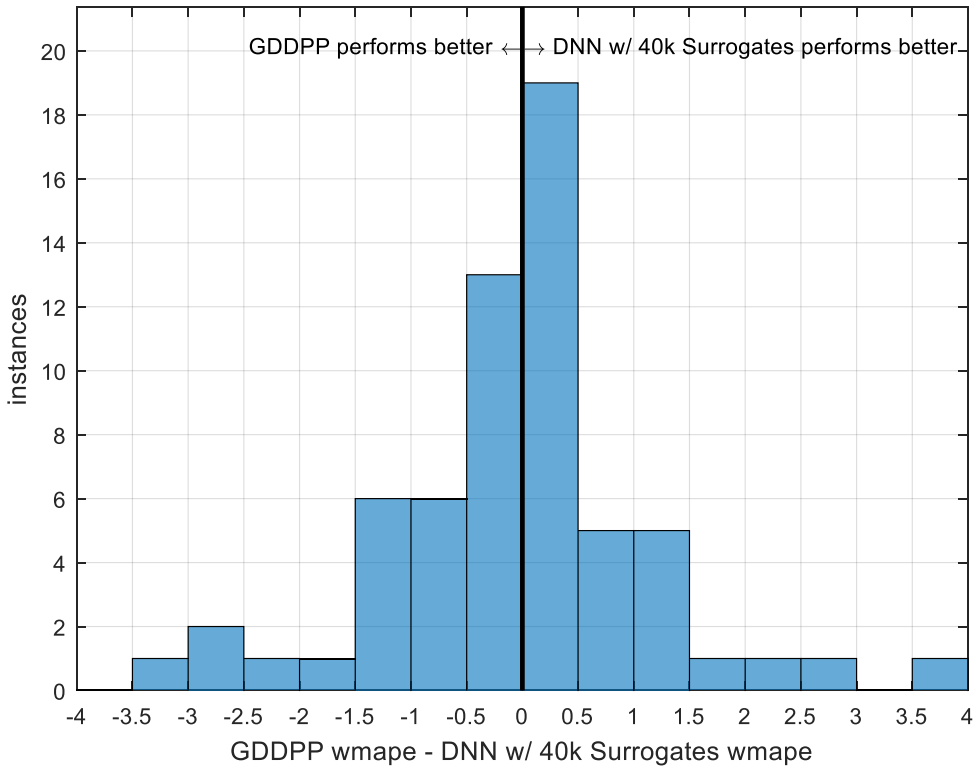

Figure 4-8: This is a histogram of the differences in WMAPE between the proposed DNN with 40,000 surrogates and the GasDay ensemble. Values on the left of the thick line at 0 indicate areas where the GasDay ensemble performs better. Those on the right indicate areas where the proposed DNN with 40,000 surrogates performs better.

This is interesting as it runs counter to the results in Section 4.3. The only difference is that the DNNs here have four layers instead of five layers. This is interesting and warrants further investigation in later work.

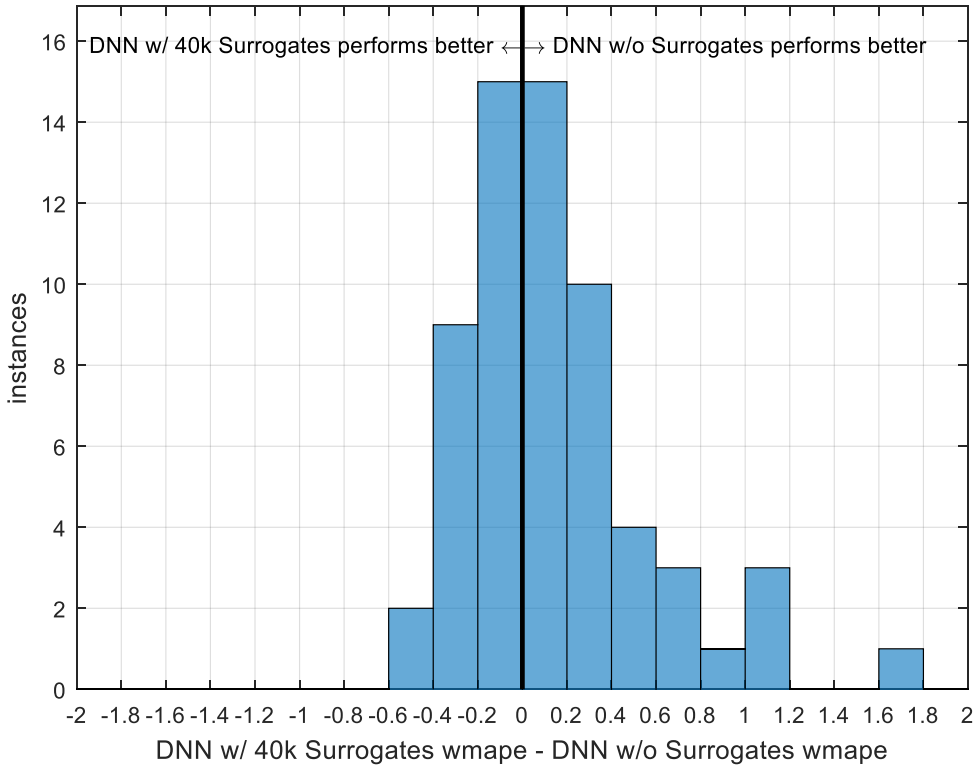

Figure 4-9: This is a histogram of the differences in WMAPE between the proposed DNN without surrogates and the proposed DNN with 40,000 surrogates. Values on the left of the thick line at 0 indicate areas where the proposed DNN with 40,000 surrogates performs better. Those on the right indicate areas where the proposed DNN without surrogates performs better.

CHAPTER 5

Deep Neural Network as a Component of a Forecast Ensemble

Chapter 3 established that the deep neural network (DNN) model can perform better than the current artificial neural network (ANN) and linear regression (LR) components of the GasDay ensemble. In Chapter 4, this thesis tried to improve this model by using more inputs, adjusting the number of layers, and pretraining the model with a large amount of surrogates. At the end of Chapter 4, it was concluded that a model with 73 inputs, five hidden layers of 60, 60, 60, 60, and 12 neurons, respectively, performs significantly better than any of the current GasDay component models and as well as the current GasDay ensemble. In this chapter, the performance of the GasDay ensemble with and without the DNN component established in Chapter 4 is analyzed. It is found that a DNN component provides enough value when using observed weather that, for many areas, it is worth it to include a DNN component in the GasDay ensemble.

5.1 The GasDay ensemble: dynamic post processor

The GasDay dynamic post processor is an ensembling method that adjusts the weights given to each component forecast based on its recent performance. This algorithm and equations come from [13]. First, the component forecasts, $\hat{c}_{j,k}$, are calculated. In this case, j refers to the component number of the n component models, and k refers to the day for which the forecast is being made. Each component also is given two tuning parameters, θ_1 and θ_0, which are adjusted daily as part of this process. First, the *a posteriori* tuned forecast for day $k-2$, two days ago, is calculated. This is

done because the actual demand will not be known for yesterday at the time the forecast for today is made. The *a posteriori* tuned forecast is

$$\tilde{c}_{j,k-2} = \begin{bmatrix} 1 & \hat{c}_{j,k-2} \end{bmatrix} \begin{bmatrix} \theta_{j,0} \\ \theta_{j,1} \end{bmatrix}. \tag{5-1}$$

Next, the *a posteriori* error is calculated using the known demand, s_{k-2},

$$\tilde{e}_{j,k-2} = \tilde{c}_{j,k-2} - s_{k-2}. \tag{5-2}$$

The *a posteriori* error, $\tilde{e}_{j,k-2}$, is bounded so that small errors are ignored and extremely large errors do not affect the model, as they are likely to be outliers. After these errors are calculated the tuning parameters are updated to reduce the error on $\tilde{c}_{j,k-2}$. Using a forgetting factor γ,

$$\theta = (1-\gamma)\theta + \gamma \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \tag{5-3}$$

After the tuning parameters have been set, the *a posteriori* error is recalculated using Equations 5-1 and 5-2.

The next step is to calculate the recent mean, $\tilde{\mu}_{j,k-2}$, and variance, $\tilde{v}_{j,k-2}$, of the two component models using Equations 5-4 and 5-5. The variable α is another forgetting factor which helps the ensemble emphasize more recent trends.

$$\tilde{\mu}_{j,k-2} = \alpha \cdot \tilde{\mu}_{j,k-3} + (1-\alpha) \cdot \tilde{e}_{j,k-2}. \tag{5-4}$$

$$\tilde{v}_{j,k-2} = \alpha \cdot \tilde{v}_{j,k-3} + (1-\alpha) \cdot (\tilde{e}_{j,k-2} - \tilde{u}_{j,k-2})^2. \tag{5-5}$$

Finally, using this mean and variance, the weight placed on each component model, w_j and the final forecast $\hat{s}_k$ are calculated using Equations 5-6 and 5-7. Note that $\sum_{j=1}^{n} w_j = 1$.

$$w_j = \frac{\frac{1}{\sqrt{\tilde{v}_j}}}{\sum_{i=1}^{n} \frac{1}{\sqrt{\tilde{v}_i}}}, \text{ and} \tag{5-6}$$

$$\hat{s}_k = \sum_{j=1}^{n} w_j (\tilde{c}_{j,k} - \tilde{\mu}_{j,k-2}) \tilde{c}_{j,k}^2 - \tilde{\mu}_{j,k-2}^2 . \tag{5-7}$$

More detailed information on the GasDay ensemble can be found in [13].

5.2 Experiment

This experiment follows the same pattern as those in Chapter 3 and Chapter 4. This experiment is run using the entire 2015-2016 heating seasons worth of forecasts for each of 67 operating areas. This is less than the 88 areas used in Chapter 3 and parts of Chapter 4 because of infrastructure issues related to generating input sets with 73 features as opposed to 26. The results are also compared on unusual days.

Unlike with previous experiments, in this experiment, it matters how many and which areas fall on each side of the histogram. It is expected that most ensembles will perform better with the additional deep neural network component. Additionally, the amount of improvement provided is important. The addition of a deep neural network component would necessitate many infrastructure changes within the GasDay project, so the total improvement to all models needs to meet a certain threshold for it to be included in the production GasDay ensemble. That threshold is determined by a variety of factors outside of the scope of this research and thus cannot be defined in this thesis.

5.3 Results

The first results, from comparing the GDDPP to the GDDPP with an additional DNN component are as expected. As can be seen in Figure 5-1, the additional component gives at least some improvement on all days to each area. The p-values on unusual days are also shown in Table 5-1.

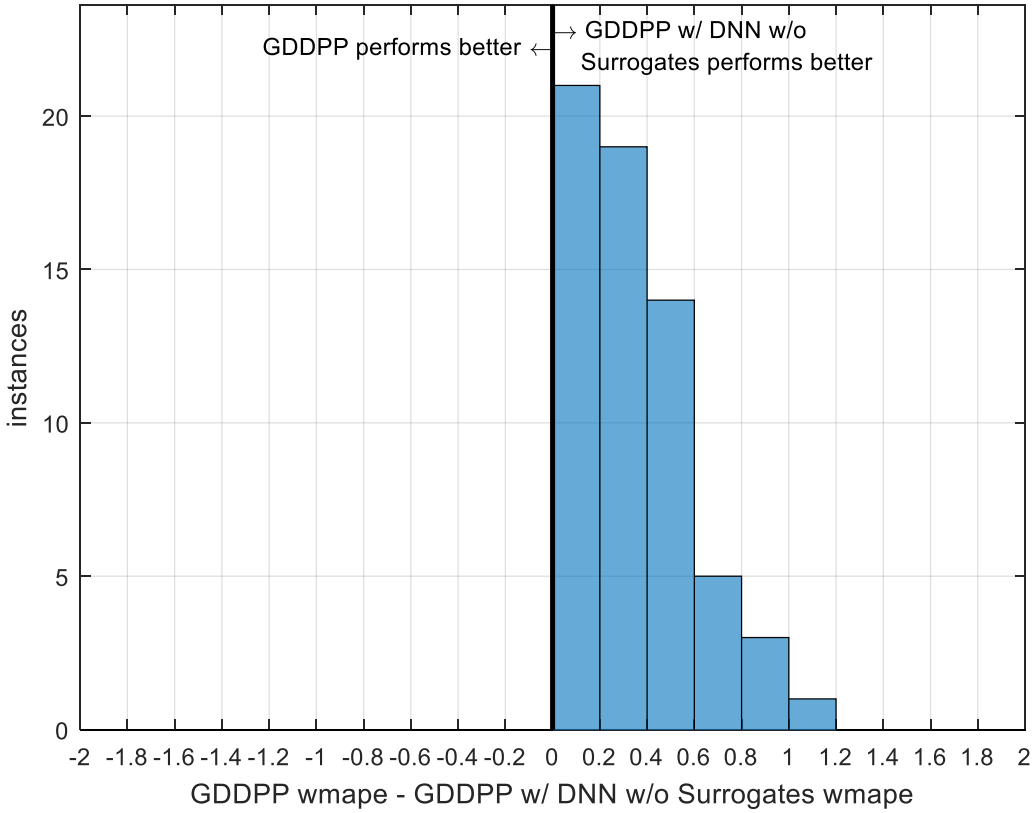

Figure 5-1: This is a histogram of the differences in WMAPE between the GasDay ensemble with the proposed DNN component without surrogates and the current GasDay ensemble. Values on the right indicate areas where the GasDay ensemble with the proposed DNN component without surrogates performs better.

Table 5-1: Right-tailed t-test comparing the current GasDay ensemble to the GasDay ensemble with the proposed DNN component without surrogates on each unusual day type. Values less than 0.05 indicate unusual day types on which the GasDay ensemble with the proposed DNN component without surrogates performs significantly better. Histograms for each of these values are included in Appendix A.6.

Unusual Day Type	**p-value**
All Days	**1.48×10^{-17}**
Coldest Days	**8.26×10^{-9}**
Colder Than Normal Heating Days	**5.76×10^{-5}**
Warmer Than Normal Heating Days	**2.10×10^{-6}**
Windiest Heating Days	**2.14×10^{-6}**
First Heating Days	**2.32×10^{-4}**
First Non-Heating Days	**0.0204**

The GasDay ensemble with a DNN component trained on 40,000 surrogates also performed well in comparison to the GDDPP. This can be seen in Figure 5-2. The p-values for unusual day types are included in Table 5-2. Again, these results are clear, conclusive, and as expected.

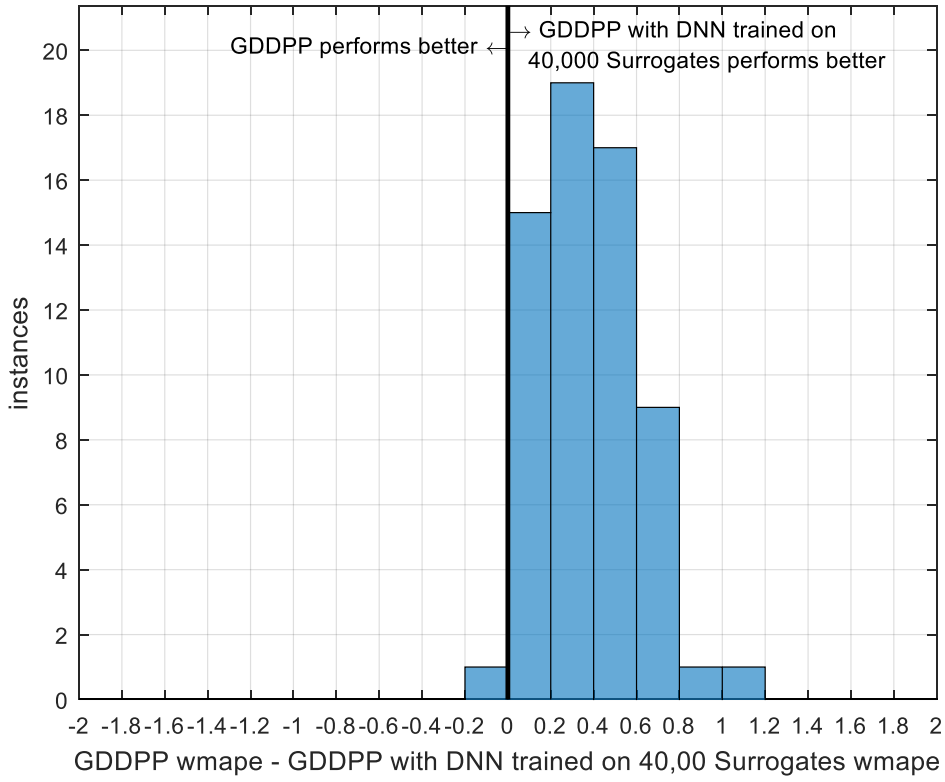

Figure 5-2: This is a histogram of the differences in WMAPE between the GasDay ensemble with the proposed DNN component trained on 40,000 surrogates and the current GasDay ensemble. Values on the left of the thick line at 0 indicate areas where the current GasDay ensemble performs better. Those on the right indicate areas where the GasDay ensemble with the proposed DNN component trained on 40,000 surrogates performs better.

Table 5-2: Right-tailed t-test comparing the current GasDay ensemble to the GasDay ensemble with the proposed DNN component trained on 40,000 surrogates on each unusual day type. Values less than 0.05 indicate unusual day types on which the GasDay ensemble with the proposed DNN component trained on 40,000 surrogates performs significantly better. Histograms for each of these values are included in Appendix A.7.

Unusual Day Type	**p-value**
All Days	**8.30×10^{-19}**
Coldest Days	**1.59×10^{-8}**
Colder Than Normal Heating Days	**2.58×10^{-5}**
Warmer Than Normal Heating Days	**1.61×10^{-9}**
Windiest Heating Days	**1.81×10^{-7}**
First Heating Days	**4.65×10^{-6}**
First Non-Heating Days	**3.65×10^{-3}**

However, a comparison between the GasDay ensembles with the two different DNN components has unexpected results. First, as in Figure 5-3, the GasDay ensemble with DNN component trained on 40,000 surrogates seems to perform better. This is supported by the p-values in Table 5-3. Although not all of the unusual days have significant differences, most of them still favor the component trained on surrogates.

This is interesting because in Section 4.3 it was shown that, on five layer networks, having some surrogates improved the model. Then, in Section 4.4, it was shown that on four layer models, using surrogates actually made the networks perform worse. Here, using the same four layer models as in Section 4.4, the use of surrogates again is shown to be better. In the end, this puts the unspoken assumption of this thesis that better component models result in better ensemble forecasts into question. On the

other hand, perhaps this simply shows that the ensemble is able to take advantage of some specialization that comes from the use of surrogate data. Further work is needed to answer this question.

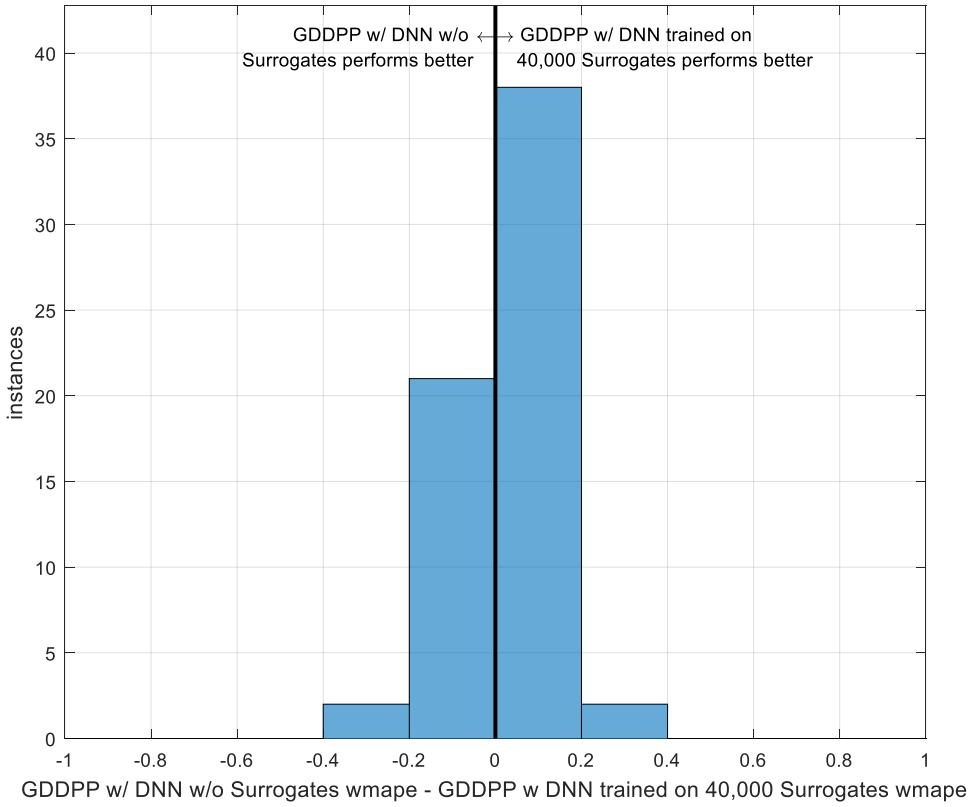

Figure 5-3: This is a histogram of the differences in WMAPE between the GasDay ensemble with the proposed DNN component trained on 40,000 surrogates and the proposed DNN component without surrogates. Values on the left of the thick line at 0 indicate areas where the proposed DNN component without surrogates performs better. Those on the right indicate areas where the GasDay ensemble with the proposed DNN component trained on 40,000 surrogates performs better.

Table 5-3: Right-tailed t-test comparing the proposed DNN component without surrogates to the GasDay ensemble with the proposed DNN component trained on 40,000 surrogates on each unusual day type. Values less than 0.05 indicate unusual day types on which the GasDay ensemble with the proposed DNN component trained on 40,000 surrogates performs significantly better. Histograms for each of these values are included in Appendix A.8.

Unusual Day Type	**p-value**
All Days	0.0704
Coldest Days	0.519
Colder Than Normal Heating Days	0.143
Warmer Than Normal Heating Days	**0.0155**
Windiest Heating Days	**1.54×10^{-3}**
First Heating Days	**0.0150**
First Non-Heating Days	0.267

With that in mind, it is important to note that the magnitudes of the differences are quite small, which is important to consider given that preparing surrogates for each area has much higher cost in both resources and time. Therefore, this thesis concludes that using surrogates to train a DNN component of the GasDay ensemble is better than not using it, but cannot be concluded that it is worth the extra infrastructure and overhead. It is also concluded from this chapter that in general including a DNN component provides significant value to many areas and some value to nearly all areas.

CHAPTER 6

Contributions and Future Work

This chapter reiterates the major contributions of this thesis and discusses future advancements that can be done to improve the DNN and GasDay forecasts.

6.1 Contributions

This section discusses the major contributions of this thesis, of which there are two. The first is the overall improvement to the GasDay forecast and the knowledge gained about deep neural network regression for forecasting natural gas consumption. The second is a groundwork for proposing and examining new models for the GasDay ensemble.

6.1.1 Overall GasDay forecast improvement

The primary business contribution of this thesis is the improvement to the GasDay ensemble. A new component model that provides at least some improvement across nearly all areas has been proposed and examined. The average improvement provided by this component on the 63 areas examined is 0.36 points of WMAPE, with the max improvement for an area being 1.12 points of WMAPE. When compared to the previous magnitudes, the percent improvement is 6.98% with a single area having a 20.01% improvement.

Other contributions come in the form of academic knowledge. Few of these things were examined fully in this thesis because of their scope and limitations such as network training time and the size of this document. The first of these is a better understanding of how the number of inputs and the number of neurons in the hidden layers impact

forecasting performance. The second is appreciation of the complex relationship that the choice to use surrogate data has with performance. Finally, there is more general understanding of how deep neural networks can perform in the context of regression forecasting.

6.1.2 Groundwork for proposing new component models

The second, equally important, contribution of this thesis is a groundwork for proposing new component models and examining their usefulness to the GasDay daily short-term load forecasting system. This groundwork is laid out in the organization of this thesis. In Chapter 3, a test was performed to compare the current component models to the new component, keeping as many model parameters as possible the same. If the new model performs reasonable under these conditions, then it makes sense to move on. This first step also may not make sense if there are not many parameters that can be held constant. For instance, if a decision tree component were used, there are few parameters that can be held constant with the current components, so it may make sense to skip this step.

Next, in Chapter 4, further examination was done by adjusting the parameters of the model. Doing this, a good set of parameters was found. In this case, it was not possible to find the best set of parameters, but it may be possible for other learners, in which case this should be done here.

Finally, in Chapter 5, the component models were used as part of the GasDay ensemble to see what kind of value the new component provides. Where previous chapters answer academic questions, Chapter 5 answers the business question as to

whether it makes sense to make the infrastructure changes needed to include the new model.

6.2 Future work

This section contains a variety of research interests that were beyond the scope of this thesis, but which may warrant further investigation. The first couple of ideas are related to other neural network architectures and techniques that may prove useful to the short-term load forecasting problem. The next few are related to improvements that could be made to the neural network model proposed here.

6.2.1 Convolutional neural networks

Convolutional neural networks are much larger than restricted Boltzmann machine based neural networks in terms of number of layers and neurons in each layer. This is done by convolving close neurons in the previous layer rather than being fully connected like a restricted Boltzmann machine [33]. These could be extremely powerful forecasters, as they have had an impact on the fields of sentence classification [34], image recognition [35], and speech recognition [36]. Usually the output layer of a convolutional neural network is a fully connected layer trained with a more traditional transfer function. The largest foreseeable problem with convolutional neural networks is that they require that all the inputs have the same type. In other words, it doesn't make sense to convolve temperature values with wind values. Therefore, the suggested structure is a convolutional neural network built on the last 72 to 168 hours of temperature or wind adjusted heating degree days (HDDW) with a few additional inputs such as day of week and day of year in the final output layer. This is illustrated in Figure 6-1.

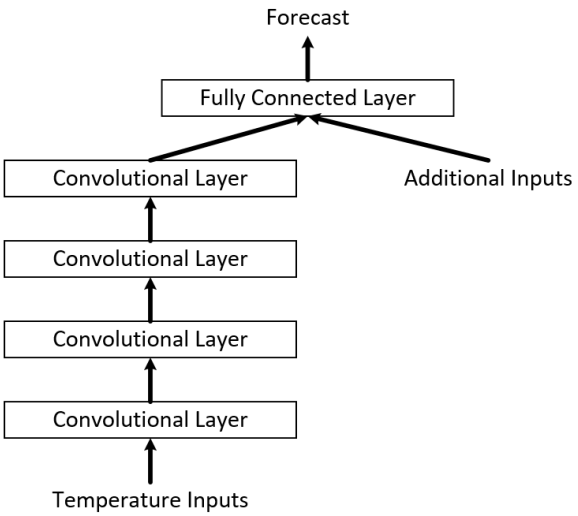

Figure 6-1: Illustration of one possible architecture for convolutional neural networks for forecasting.

6.2.2 Long short-term memory recurrent neural networks

Another interesting type of neural networks for the regression forecasting problem are long short-term memory (LSTM) recurrent neural networks [37]. Recurrent neural networks are designed to capture information from sequences of data. They do that by using the output of the model at one iteration as the input to the model at the next iteration, as shown in Figure 6-2. To calculate weights for a recurrent neural network, a process called back propagation through time must be used. Back propagation through time is described in detail in [38]. Unfortunately, basic recurrent neural networks trained using gradient descent, like those in Figure 6-2, do not tend to perform well because of the exploding gradient problem [39].

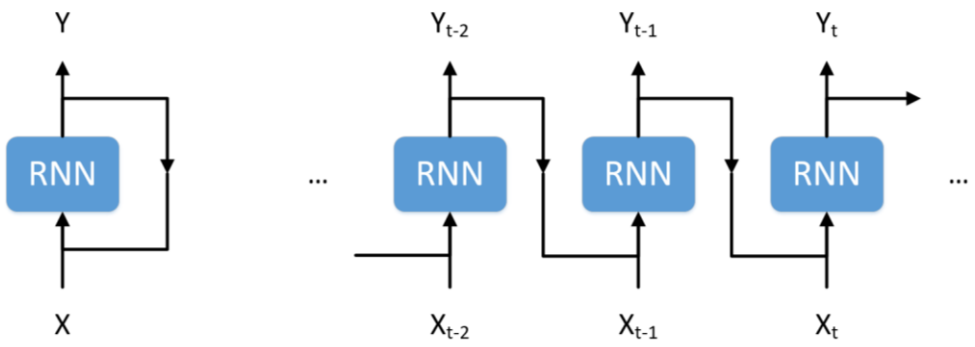

Figure 6-2: A basic recurrent neural network. An unfold version is also shown to better visualize back propagation through time.

To solve this problem, LSTM recurrent neural networks are used. LSTM recurrent neural networks can hold onto information for many more time steps than traditional recurrent neural networks [37].

LSTM neural networks certainly have some interesting implications for time series forecasting, as they can monitor and adjust to recent trends. Obviously, an LSTM recurrent neural network could be used as another component model, and the method for doing that would be similar to the content of this thesis. What might be more interesting is to use an LSTM recurrent neural network to determine the weight to be given to the outputs of each component model, similar to what the GasDay ensemble, described in Section 4, does now. This concept is shown in Figure 6-3.

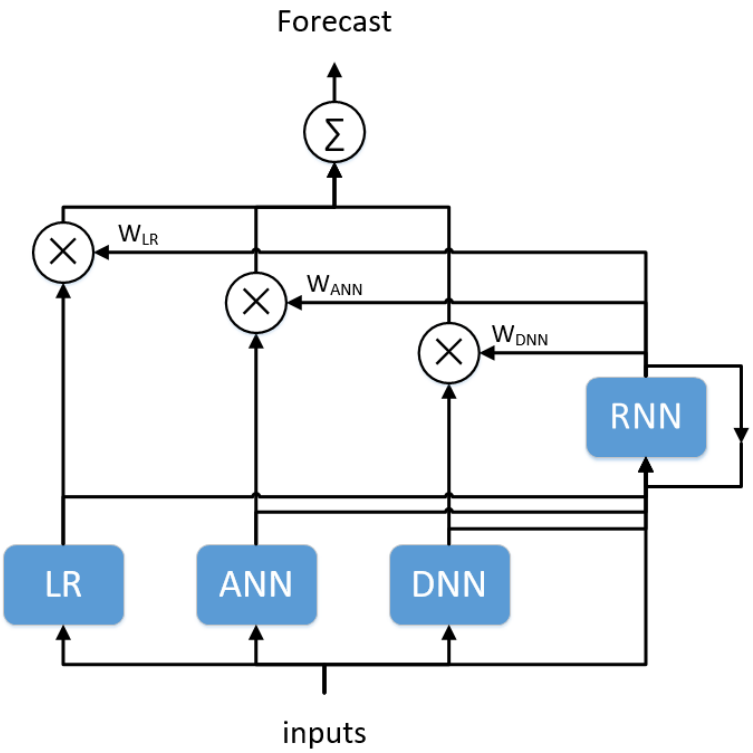

Figure 6-3: LSTM recurrent neural network used to ensemble forecasts.

6.2.3 Feature selection

In a fully connected neural network, a network where the inputs and the outputs of each layer are all fed into each neuron of the following layer, like those that are being analyzed in this thesis, true attribution analysis is difficult. But one simple way to see a rough total impact of each feature is shown in Figure 6-4 and Equation 6-1. By doing this, input features that are have little to no impact on the final forecast can be replaced with those that do.

$$\begin{pmatrix} Attribution_{x_1} \\ Attribution_{x_2} \\ Attribution_{x_3} \end{pmatrix} = \begin{pmatrix} w_{1,4} & w_{1,5} \\ w_{2,4} & w_{2,5} \\ w_{3,5} & w_{3,5} \end{pmatrix} \begin{pmatrix} w_{4,6} & w_{4,7} \\ w_{5,6} & w_{5,7} \end{pmatrix} \begin{pmatrix} w_{6,8} \\ w_{7,8} \end{pmatrix}. \quad (5\text{-}8)$$

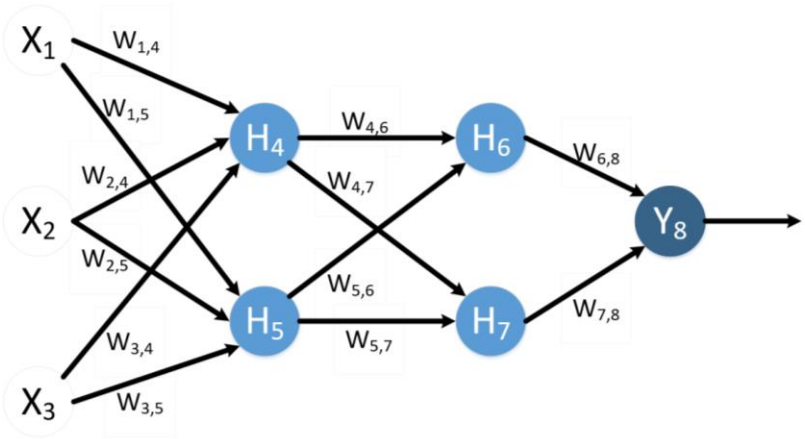

Figure 6-4: Simple neural network used to show a simple attribution analysis. In equation 6-1, the absolute values of the weight matrices are multiplied together. Absolute value is used in this case, as the sign of the weights are unimportant.

This analysis is only done on one area for the sake of brevity. Additionally, if it can be shown that this analysis is useful for one area, then it can be inferred that there are other areas that need a similar analysis. The results of this analysis are shown in Table 6-1. As can be seen in this table, the autoregressive flow features beyond the first one do not have much of an effect on the final forecast. Hence, for this model and potential others, it might be helpful to replace most of these autoregressive terms with exogenous ones. Likewise, many of the wind values have low attribution, and little performance would be lost by removing these inputs. In contrast, the time components and many of the temperature related features have higher attribution, and removing them may result in lower performance.

The most important conclusion here is that some of the features selected are not well correlated with flow and should be removed in favor of other inputs. Finding these other inputs is not needed for this thesis, but should be done in future work.

Table 6-1: Feature attribution, as describe in Figure 6-4, for one large 73 input network trained on a single area.

Feature Type	Attribution	Feature Type	Attribution	Feature Type	Attribution	Feature Type	Attribution
Temp	130.9	Temp	80.0	Temp	75.3	Flow	80.9
Temp	97.1	Temp	84.9	Temp	78.3	Flow	58.6
Temp	82.0	Temp	76.0	Temp	75.7	Flow	54.9
Temp	75.3	Temp	73.1	Wind	65.3	Flow	43.8
Temp	74.7	Temp	73.6	Wind	63.9	Flow	46.2
Temp	77.3	Temp	80.3	Wind	53.8	Flow	55.6
Temp	76.8	Temp	75.4	Wind	49.8	Time	107.6
Temp	78.3	Temp	73.5	Wind	55.7	Time	119.5
Temp	162.0	Temp	74.5	Wind	53.2	Time	101.2
Temp	175.9	Temp	81.2	Wind	55.3	Time	105.3
Temp	108.9	Temp	75.1	Wind	54.2	Time	309.9
Temp	105.4	Temp	77.7	DPT	104.7	Time	198.7
Temp	108.5	Temp	77.1	DPT	93.8	Time	263.5
Temp	93.8	Temp	78.6	DPT	87.2	Time	276.2
Temp	92.6	Temp	73.3	DPT	83.3	Time	315.7
Temp	98.1	Temp	74.3	DPT	77.5	Time	298.6
Temp	106.6	Temp	77.5	DPT	77.0		
Temp	84.5	Temp	76.1	DPT	80.2		
Temp	80.1	Temp	74.9	DPT	78.3		

6.2.4 Networks for ensemble learning

This section is motivated by interesting results at the end of Chapter 5. It was shown that one forecasting model clearly performed better than another forecasting model when they were evaluated individually. Then, when the forecasting models were evaluated as part of an ensemble the model that performed worse individually resulted in a better ensemble forecast. Further work should be done to understand why this happened as the knowledge an examination of how different components perform as part of the ensemble would inform future decisions on what types of component models to pursue.

6.3 Conclusions

In conclusion, deep neural networks are powerful forecasters and provide better individual forecasts than either of the current GasDay component models. Additionally, they provide some improvement when used as components of the GasDay ensemble. There are many possibilities for future work that can be explored, including several other neural network types and architectures as well as further analysis of the inputs used.

APPENDIX A

Additional Figures

This Appendix contains many figures that are interesting but not needed for the comprehension of this thesis. These figures supplemental and some of the information need to understand their significance is in the main body of this thesis. Unusual days are defined in Appendix B.

A.1 Unusual days graphs for Section 4.1 comparing the small 26-input DNN to the Large 73-input DNN

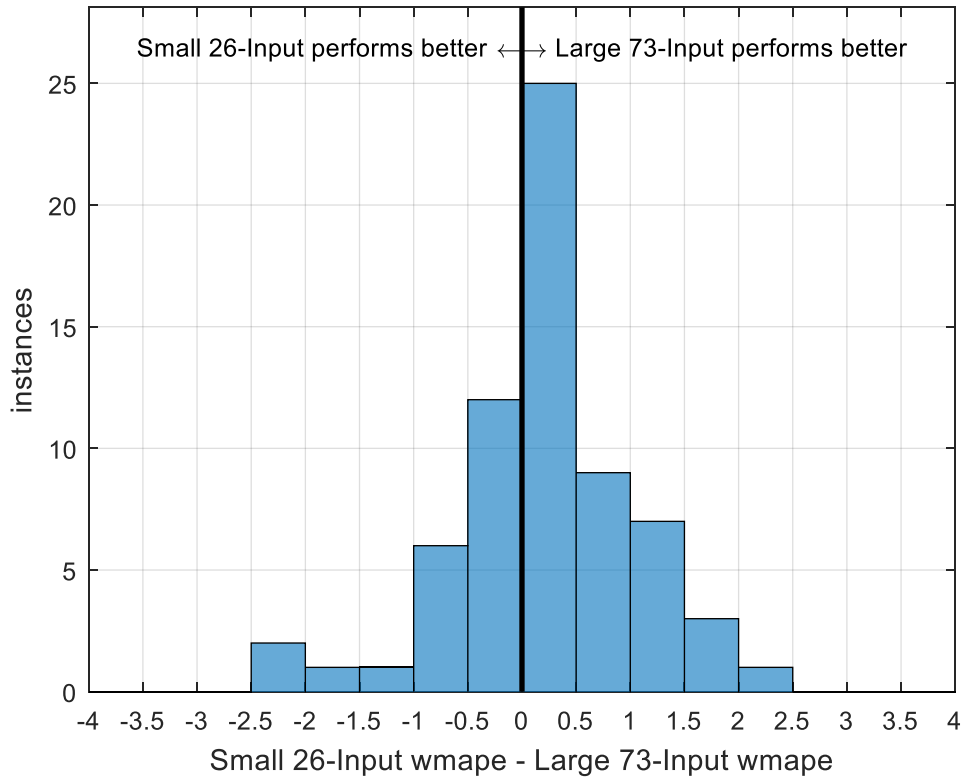

Figure A-1: Coldest days.

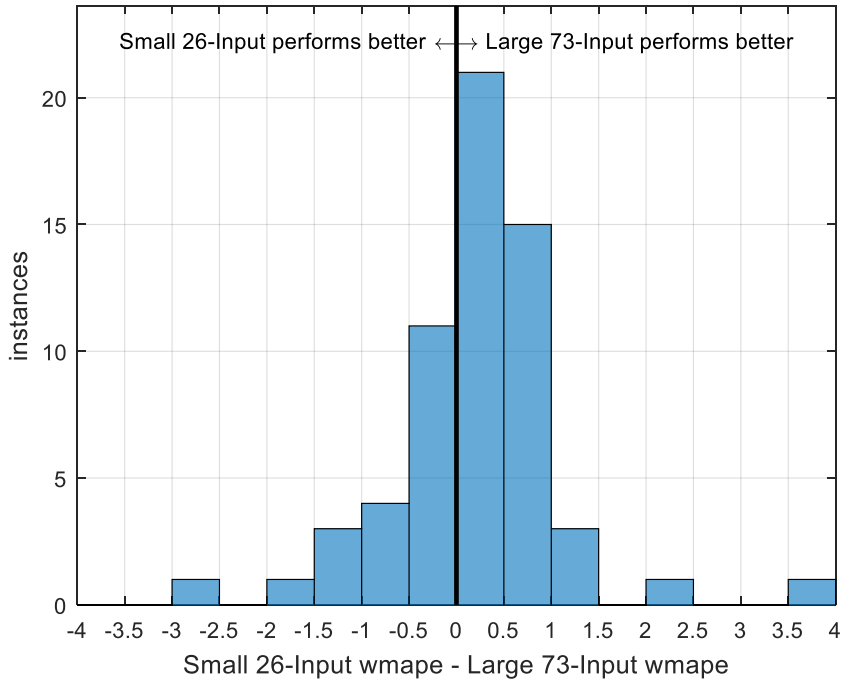

Figure A-2: Colder than normal days.

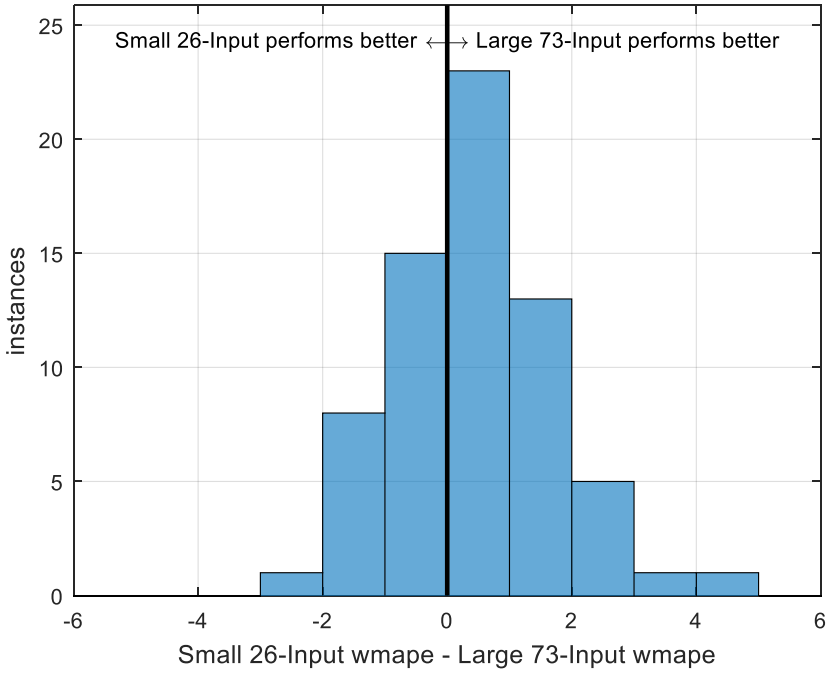

Figure A-3: Warmer than normal days.

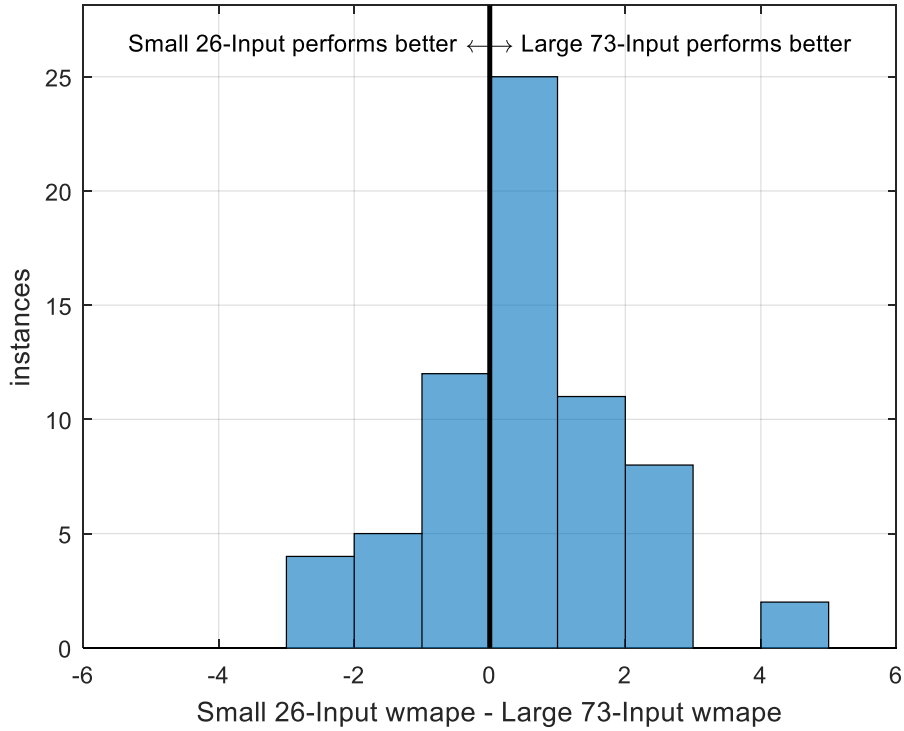

Figure A-4: Windiest days.

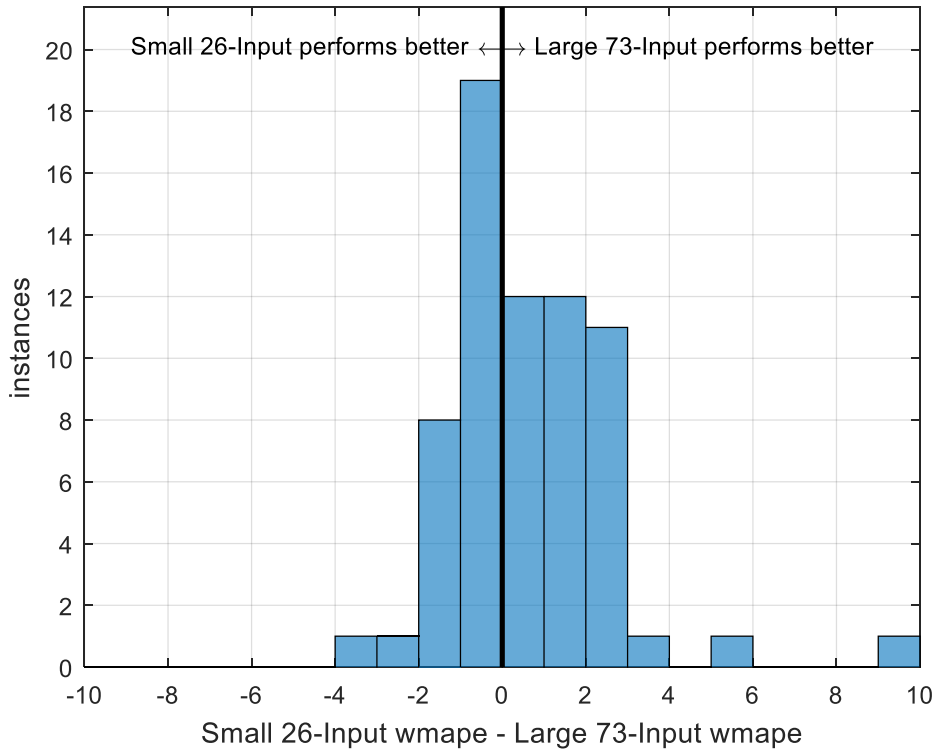

Figure A-5: First non-heating days.

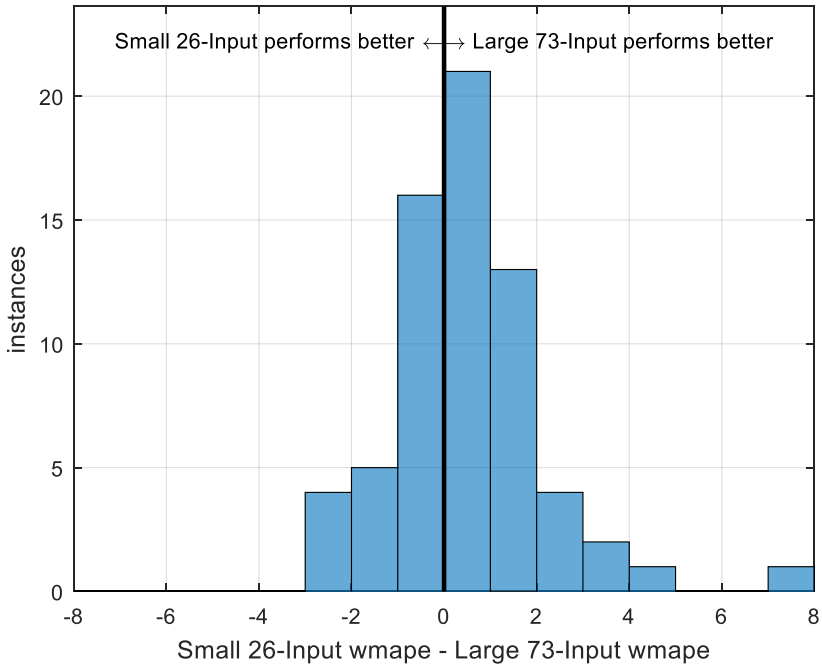

Figure A-6: First heating days.

A.2 Unusual days graphs for Section 4.1 comparing the small 26-input DNN to the Small 73-input DNN

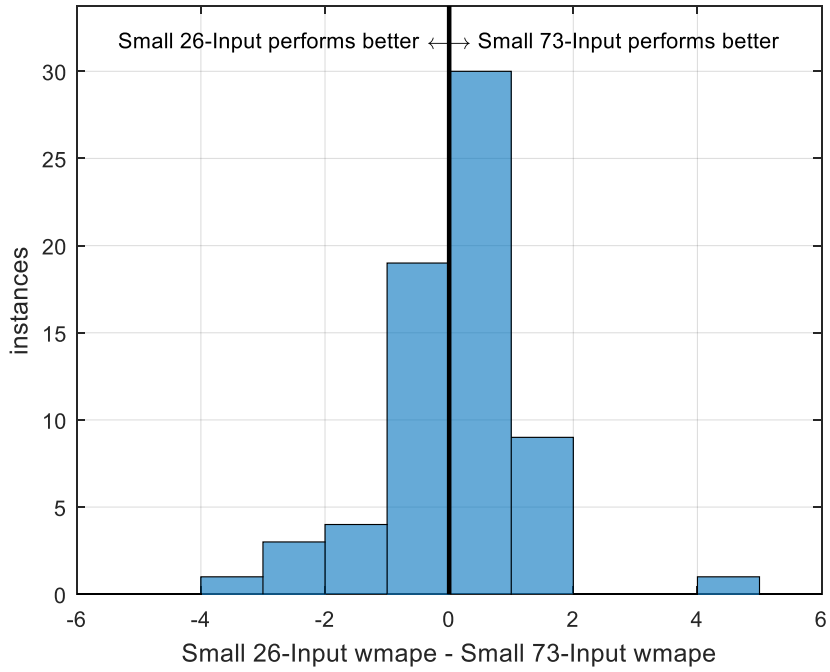

Figure A-7: Coldest days.

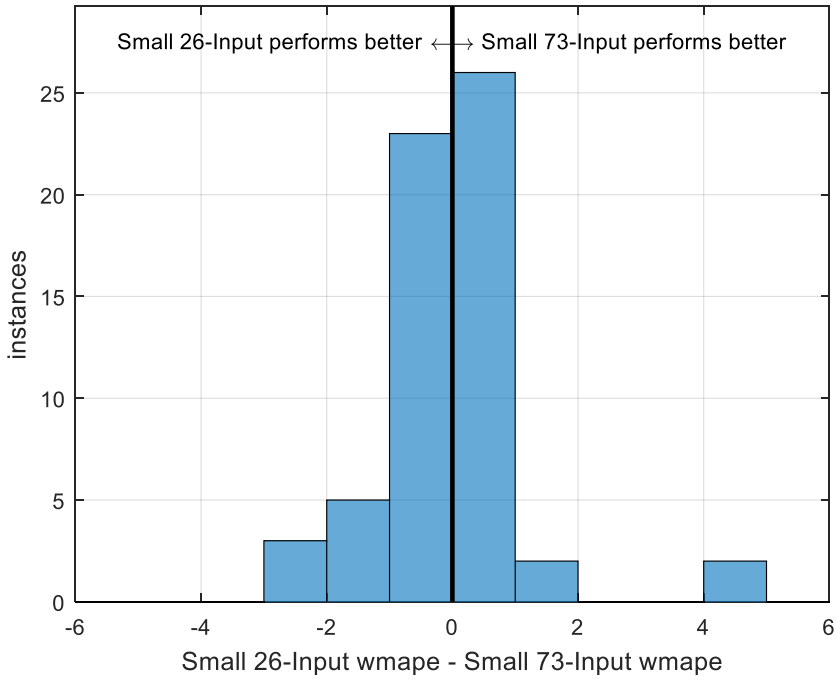

Figure A-8: Colder than normal days.

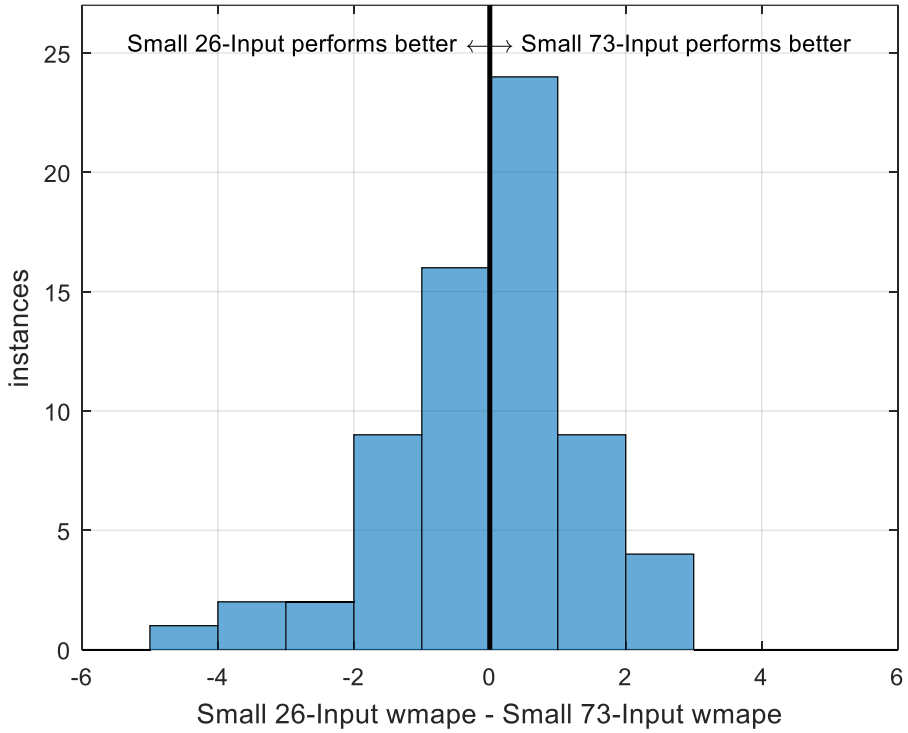

Figure A-9: Warmer than normal days.

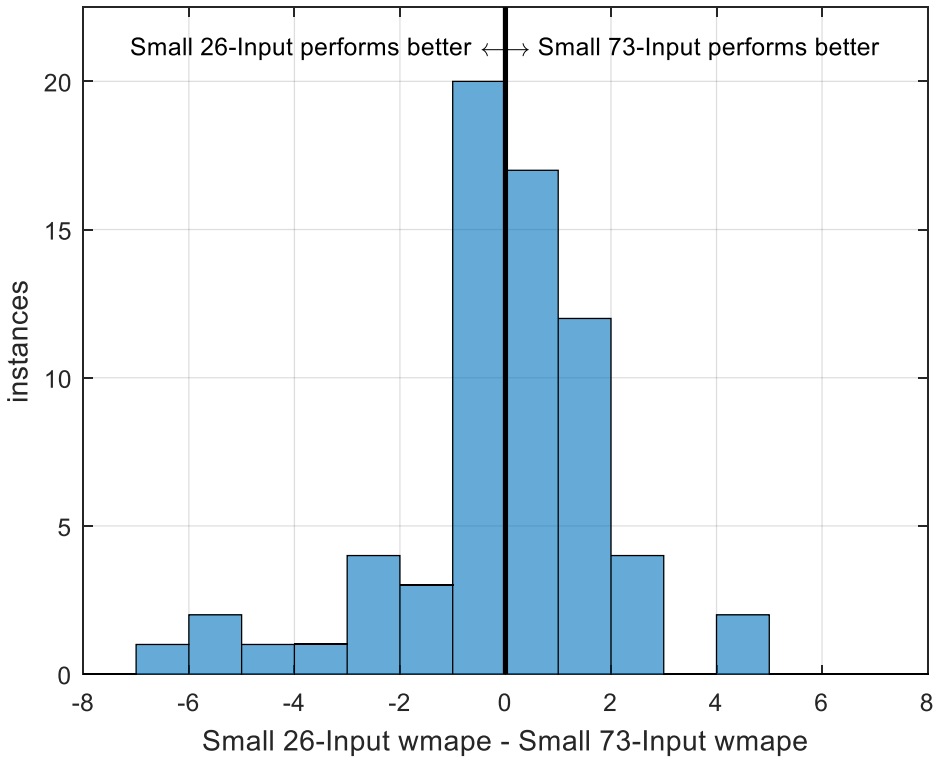

Figure A-10: Windiest days.

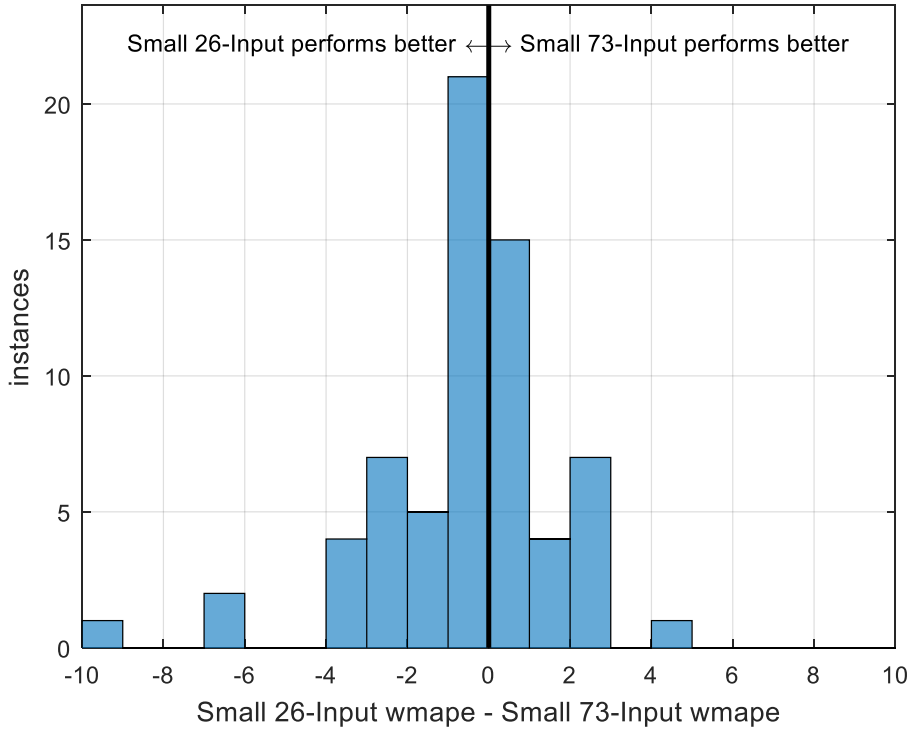

Figure A-11: First non-heating days.

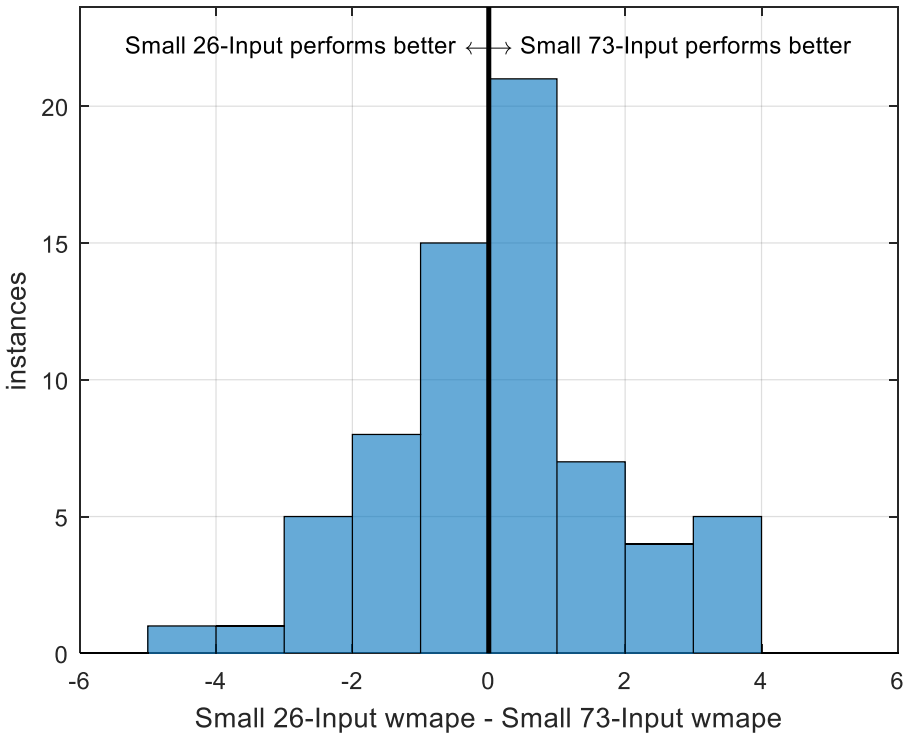

Figure A-12: First heating days.

A.3 Unusual days graphs for Section 4.1 comparing the small 26-input DNN to the Large 26-input DNN

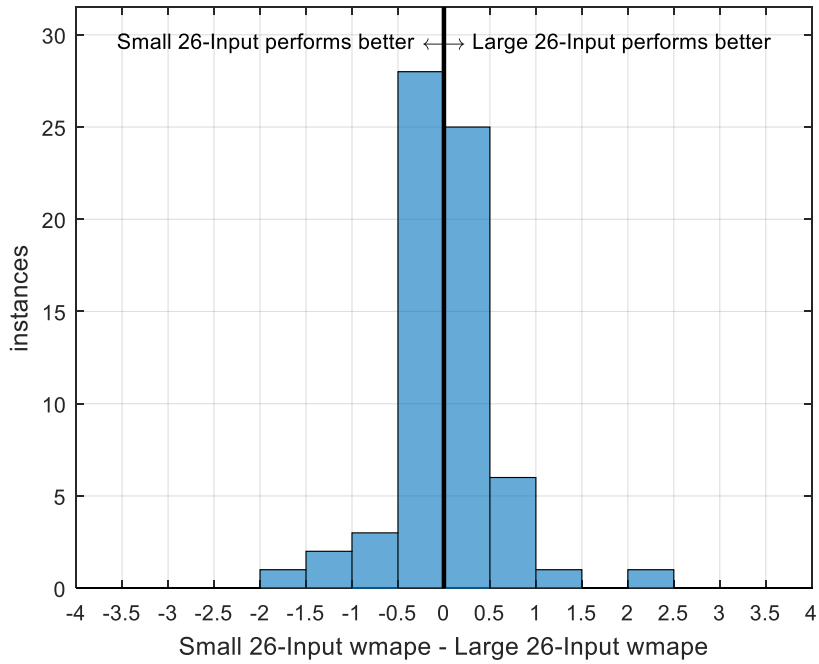

Figure A-13: Coldest days.

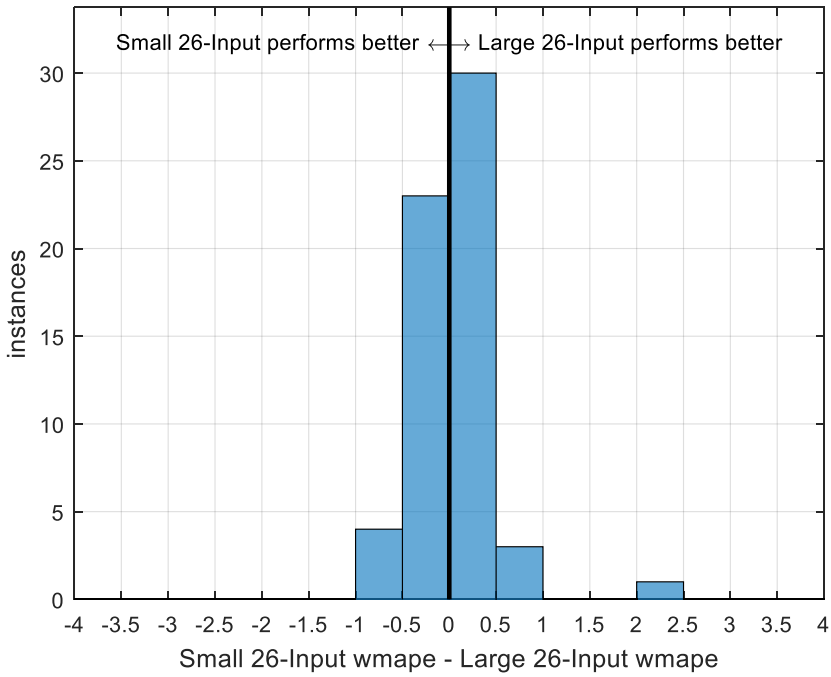

Figure A-14: Colder than normal days.

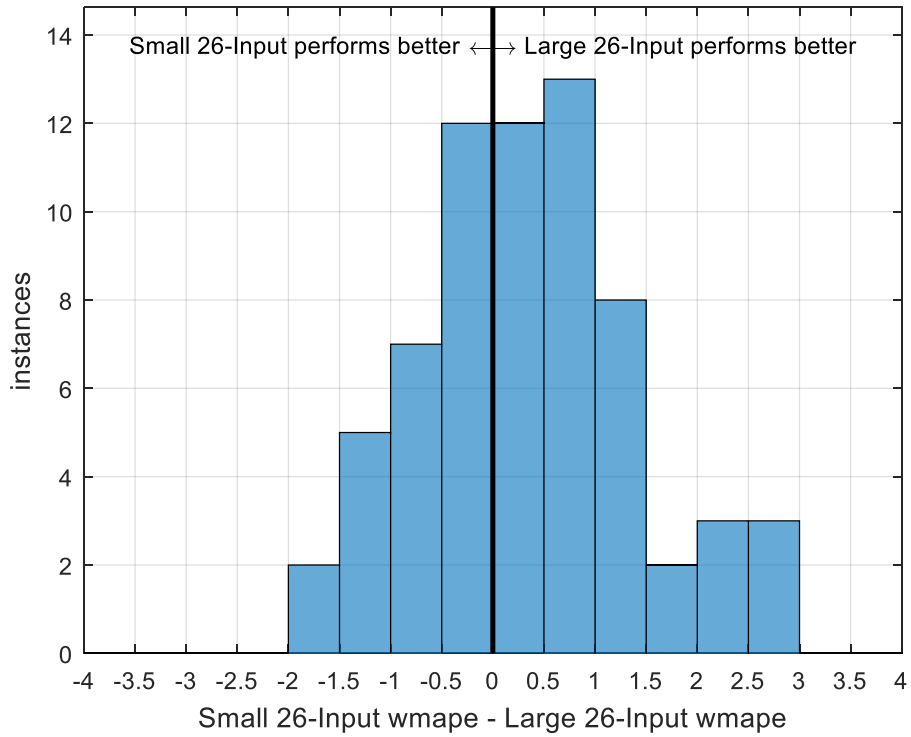

Figure A-15: Warmer than normal days.

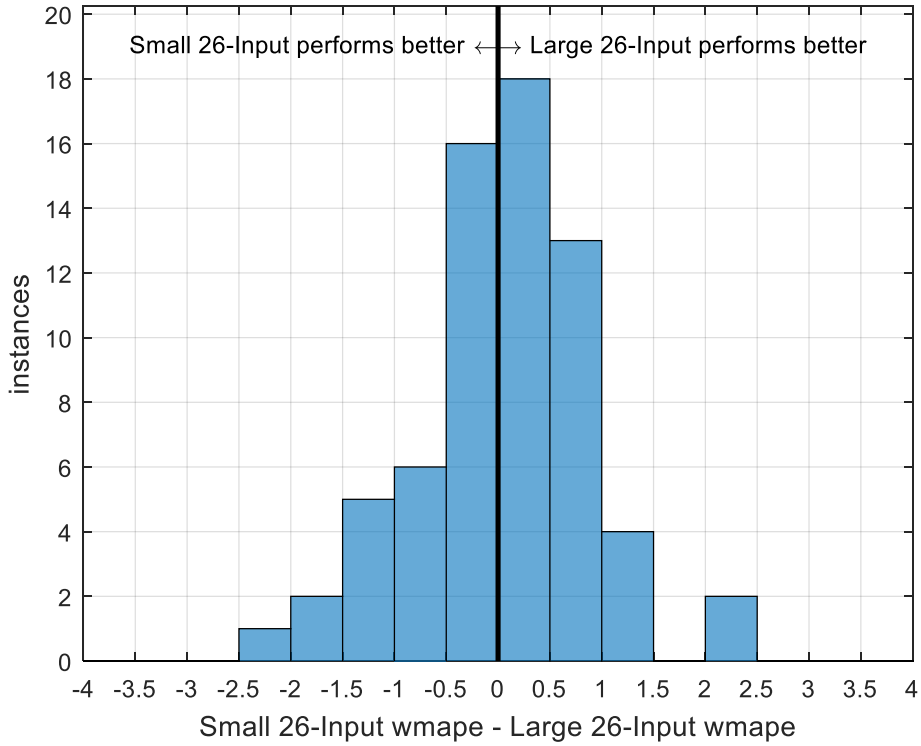

Figure A-16: Windiest days.

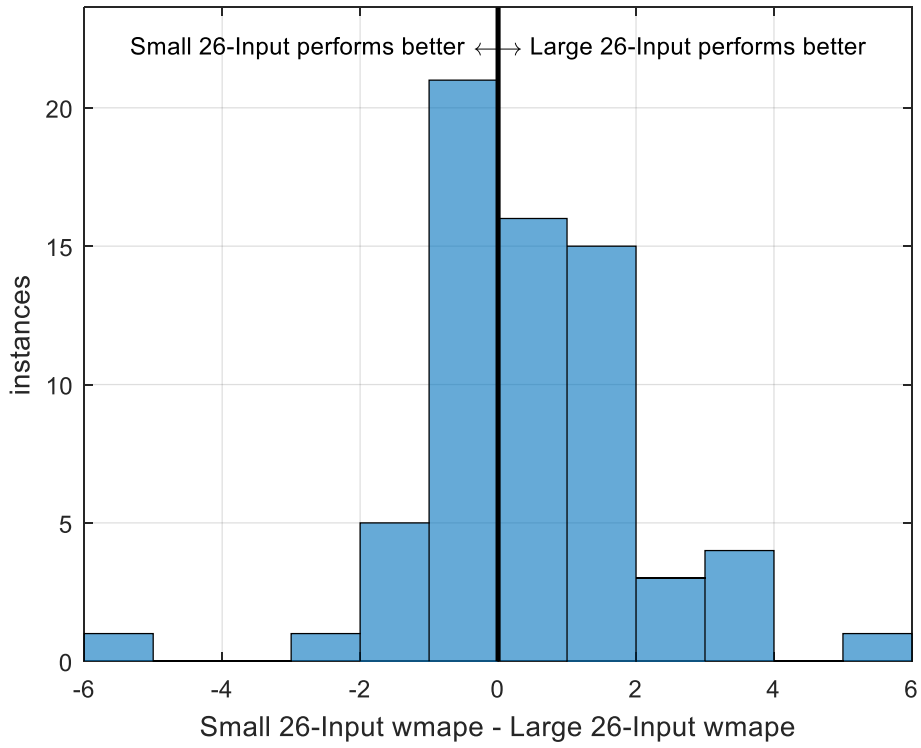

Figure A-17: First non-heating days.

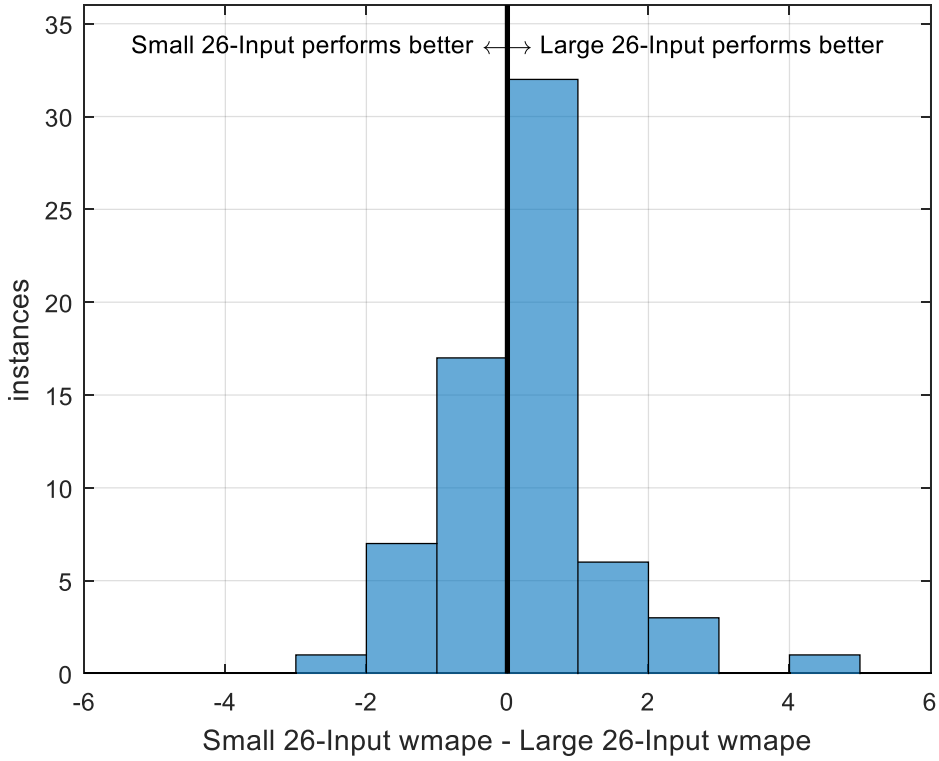

Figure A-18: First heating days.

A.4 Unusual days graphs for Section 4.3 comparing the DNN using 0 surrogates to the DNN using 40,000 surrogates

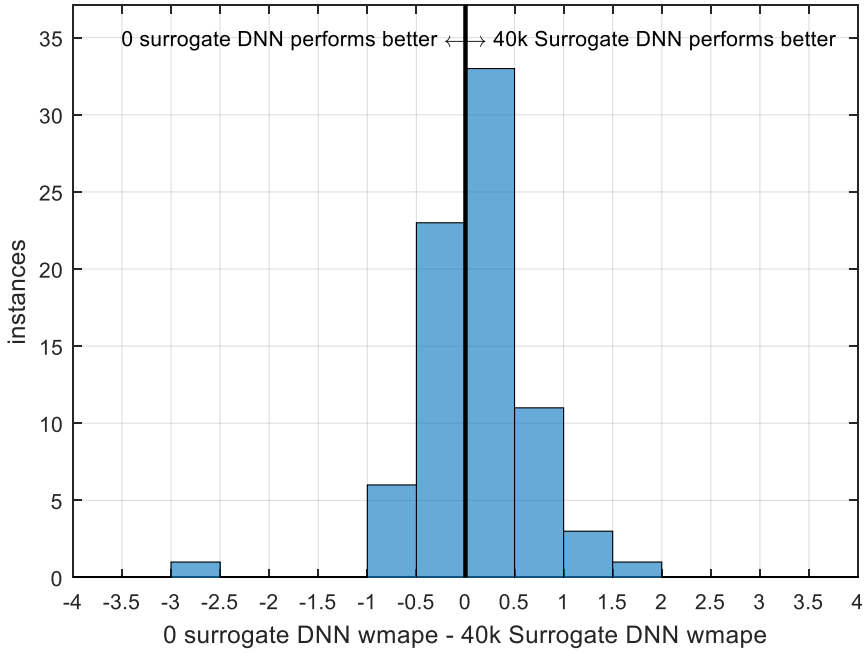

Figure A-19: Coldest days.

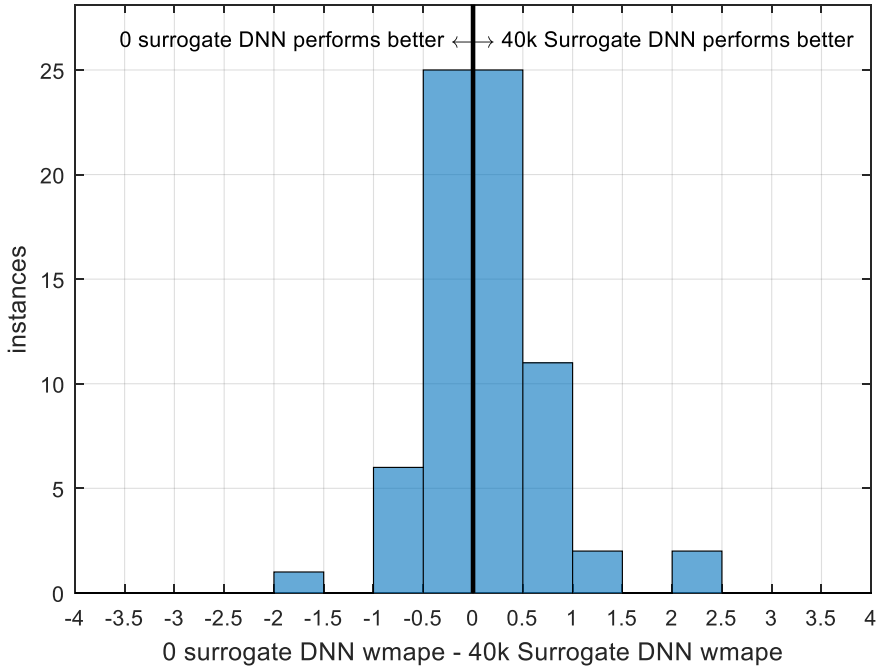

Figure A-20: Colder than normal days.

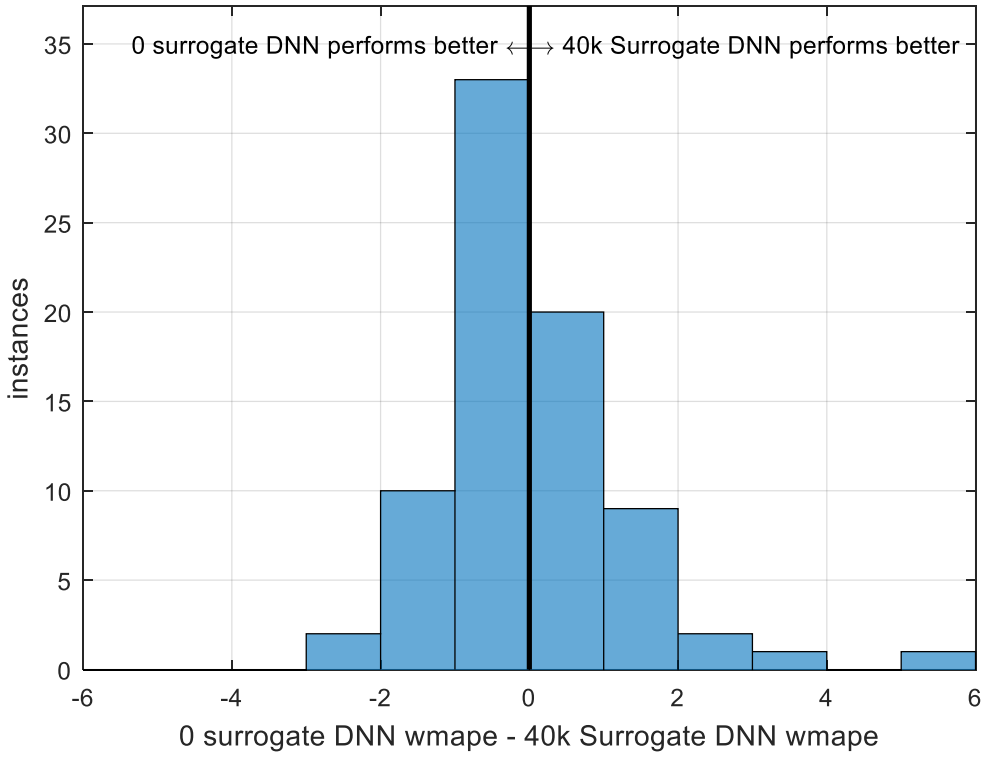

Figure A-21: Warmer than normal days.

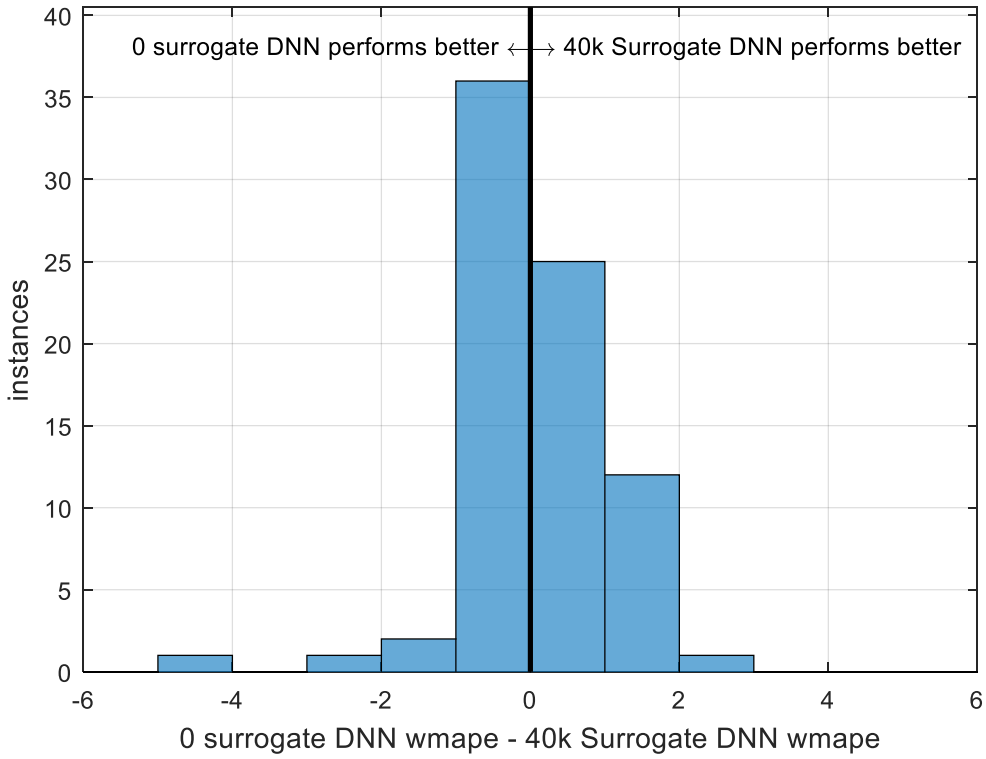

Figure A-22: Windiest days.

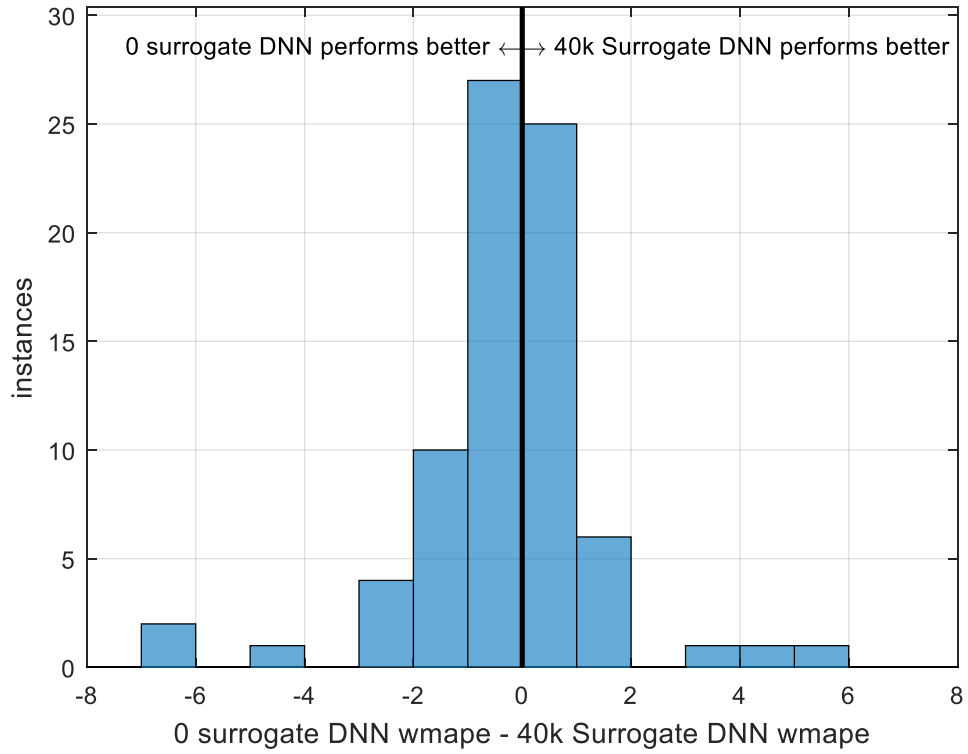

Figure A-23: First non-heating days.

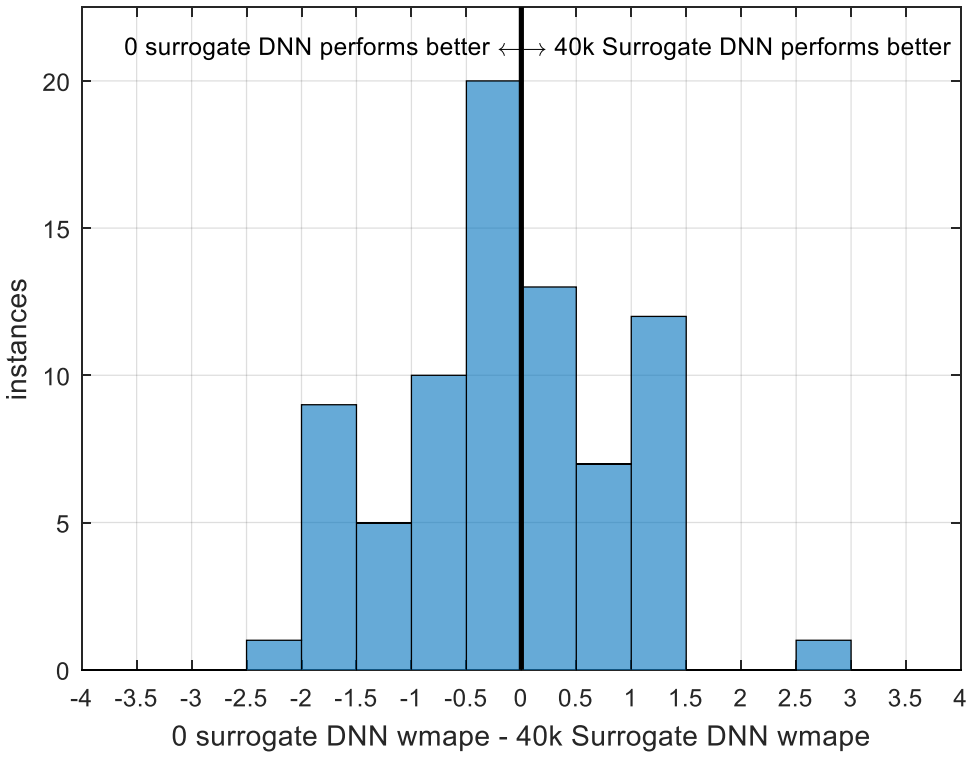

Figure A-24: First heating days.

A.5 Unusual days graphs for Section 4.3 comparing the DNN using 500,000 surrogates to the DNN using 40,000 surrogates

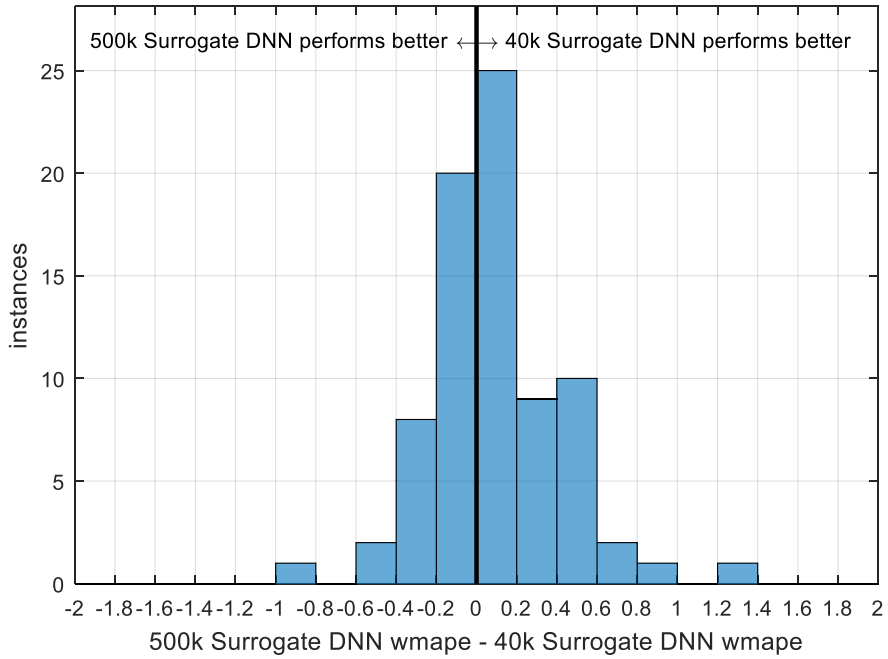

Figure A-25: Coldest days.

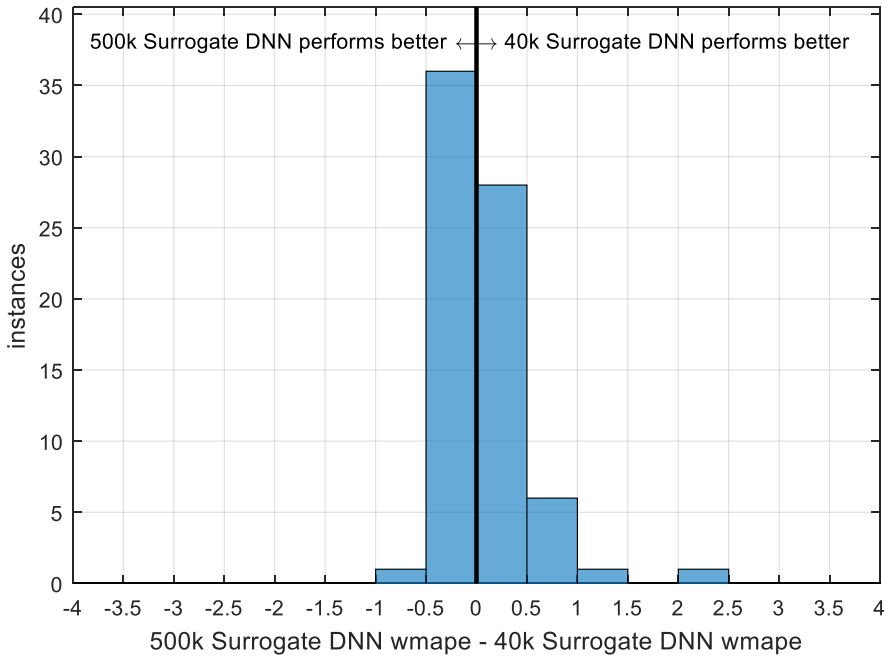

Figure A-26: Colder than normal days.

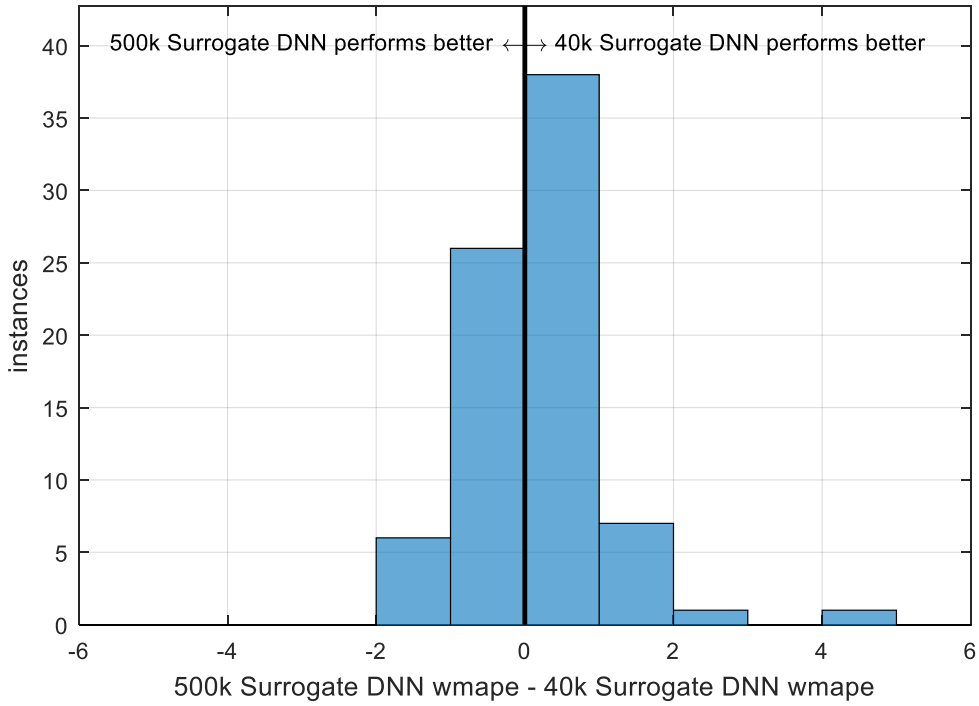

Figure A-27: Warmer than normal days.

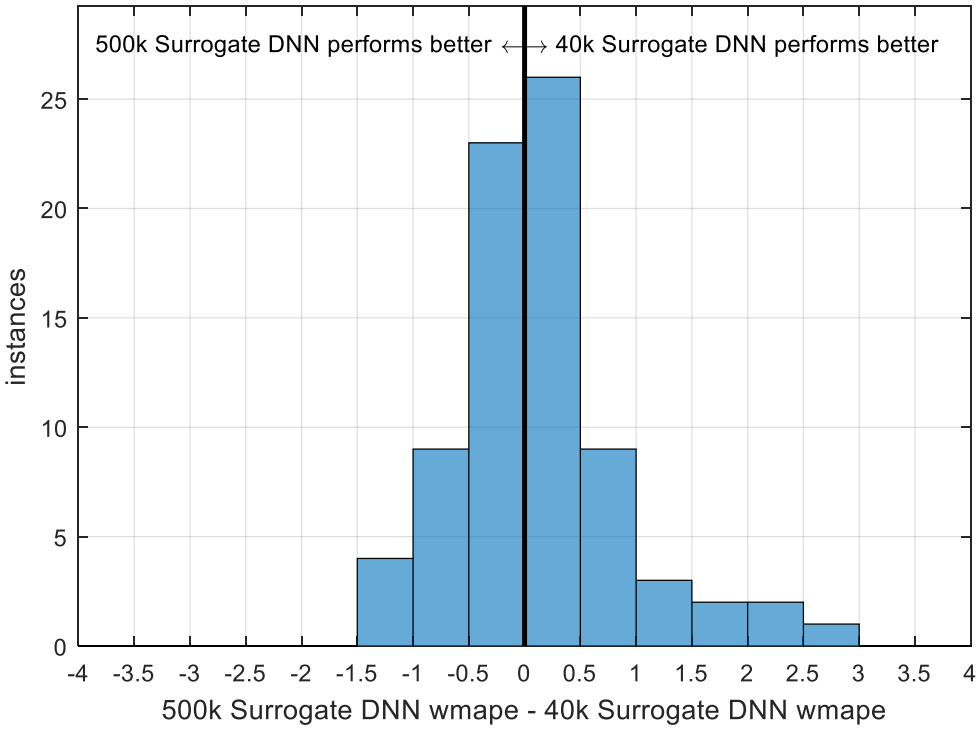

Figure A-28: Windiest days.

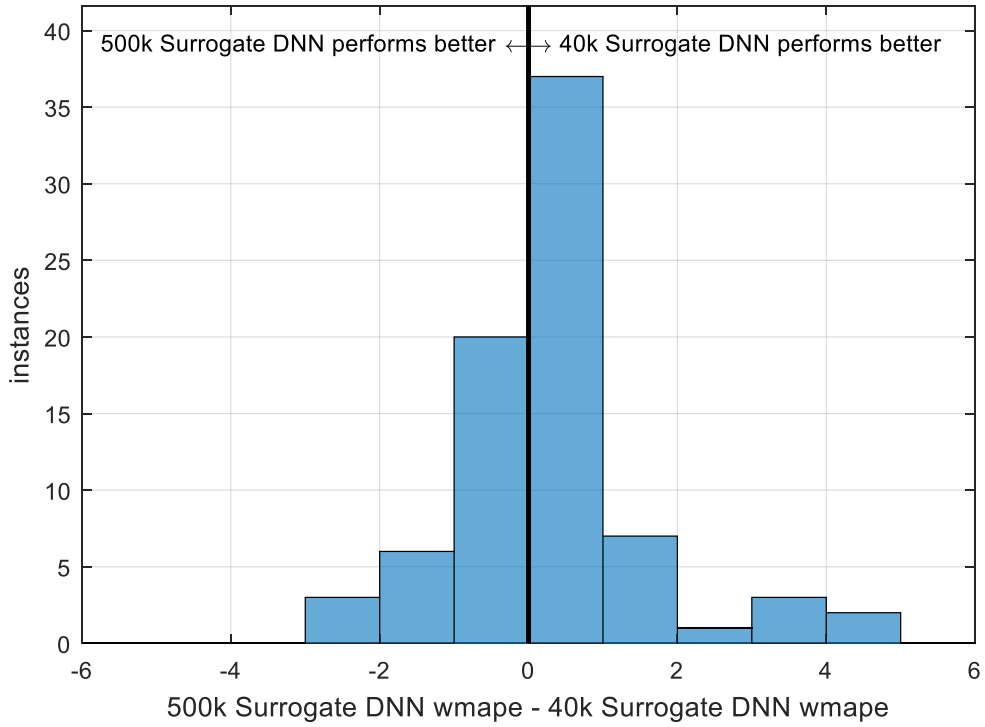

Figure A-29: First non-heating days.

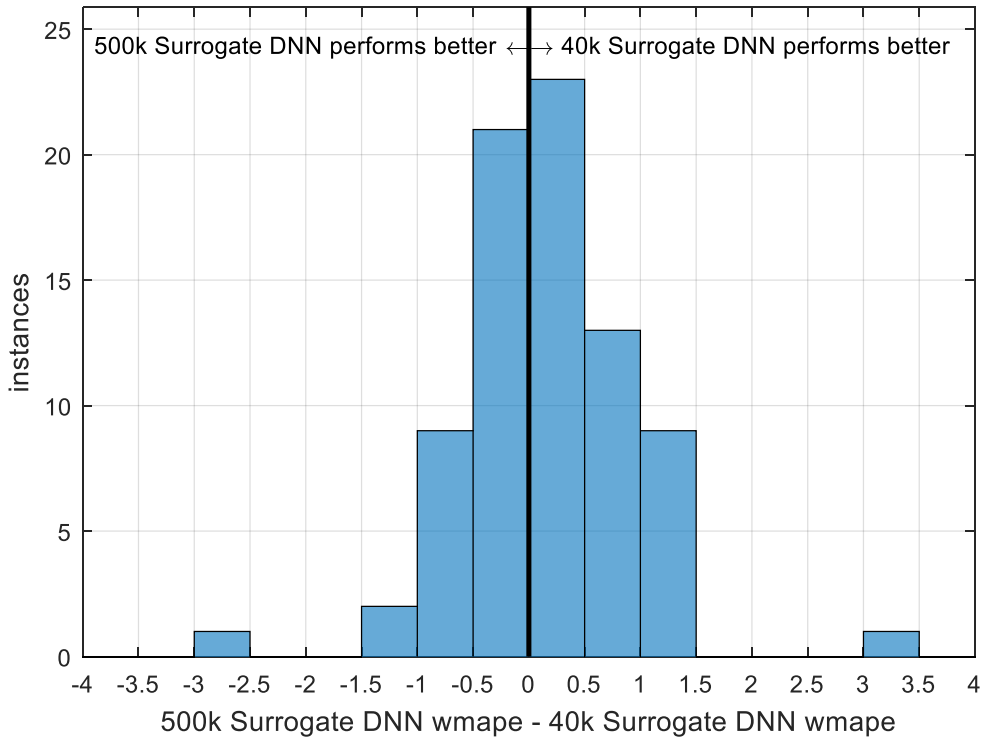

Figure A-30: First heating days.

A.6 Unusual days graphs for Section 5.3 comparing the current GasDay ensemble to the ensemble with a DNN component using 0 surrogates

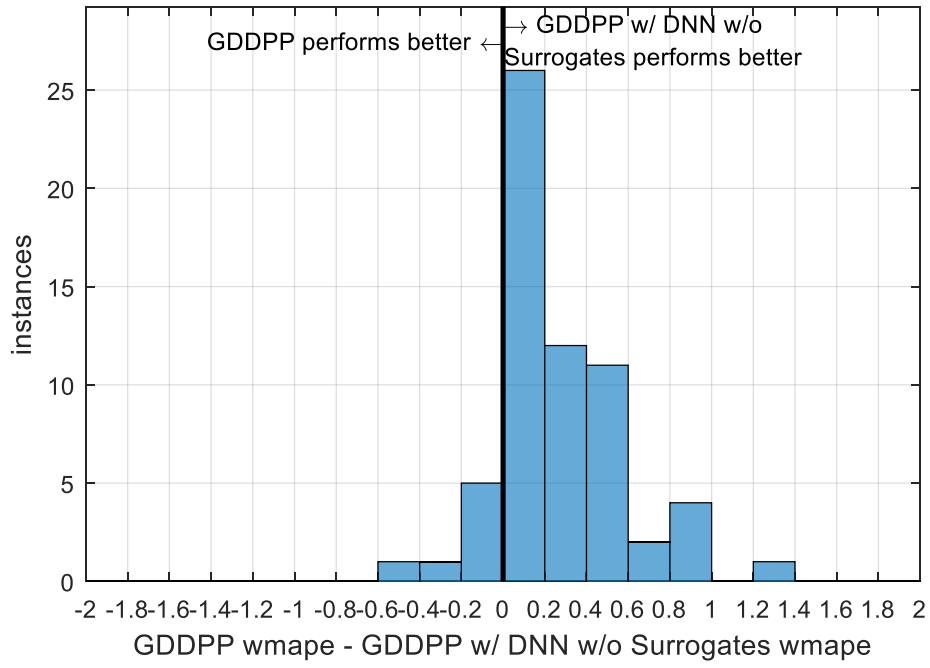

Figure A-31: Coldest days.

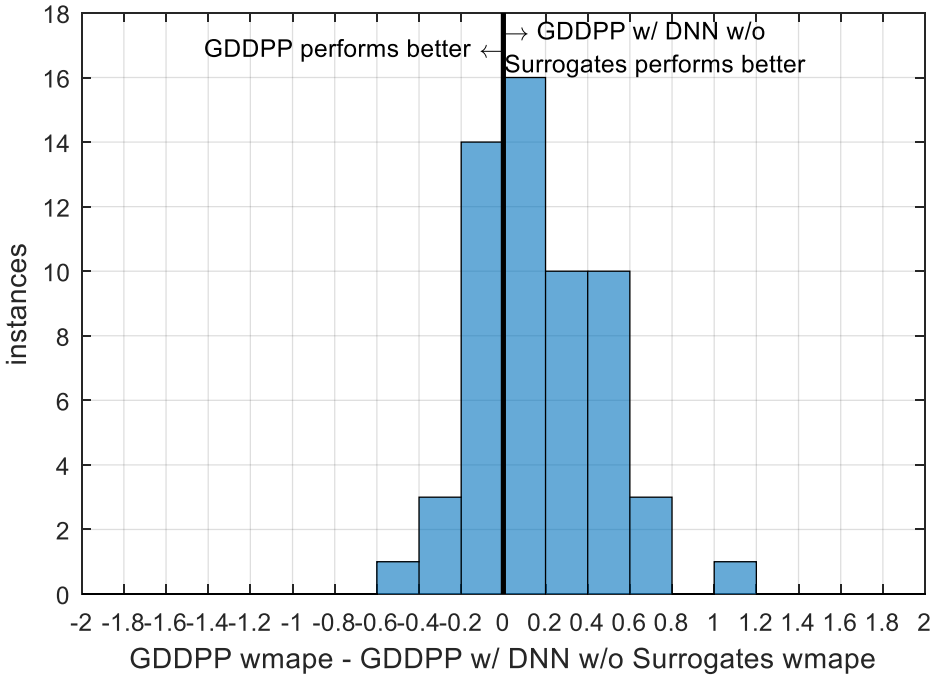

Figure A-32: Colder than normal days.

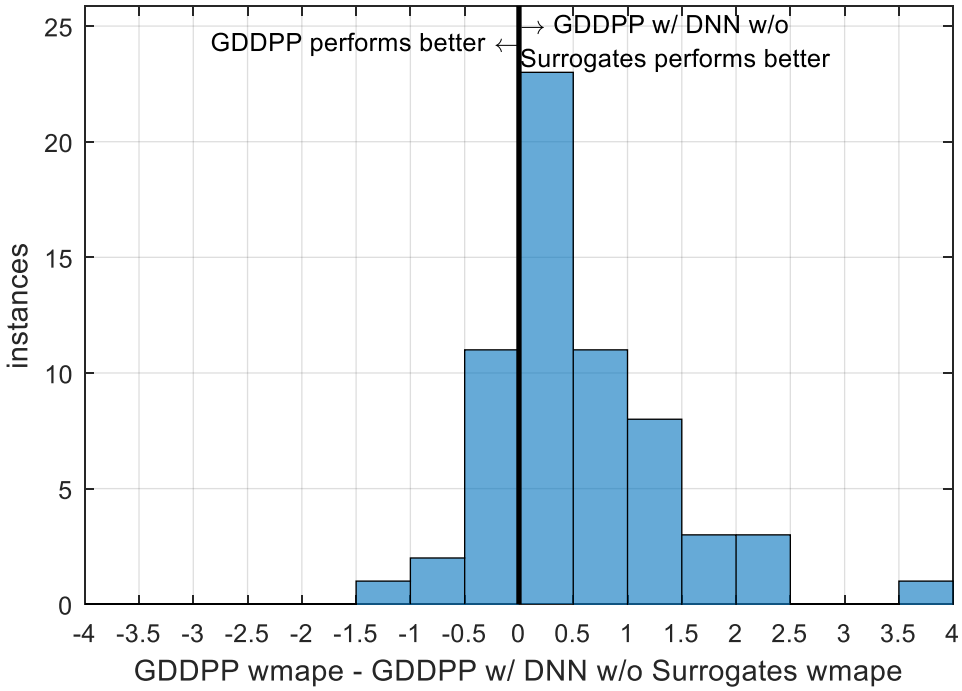

Figure A-33: Warmer than normal days.

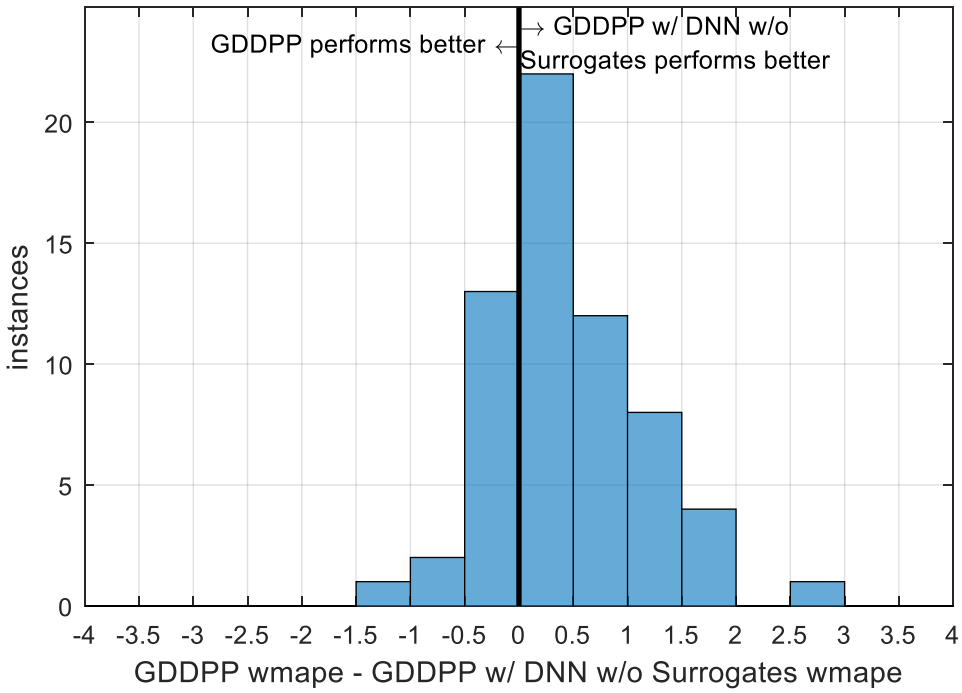

Figure A-34: Windiest days.

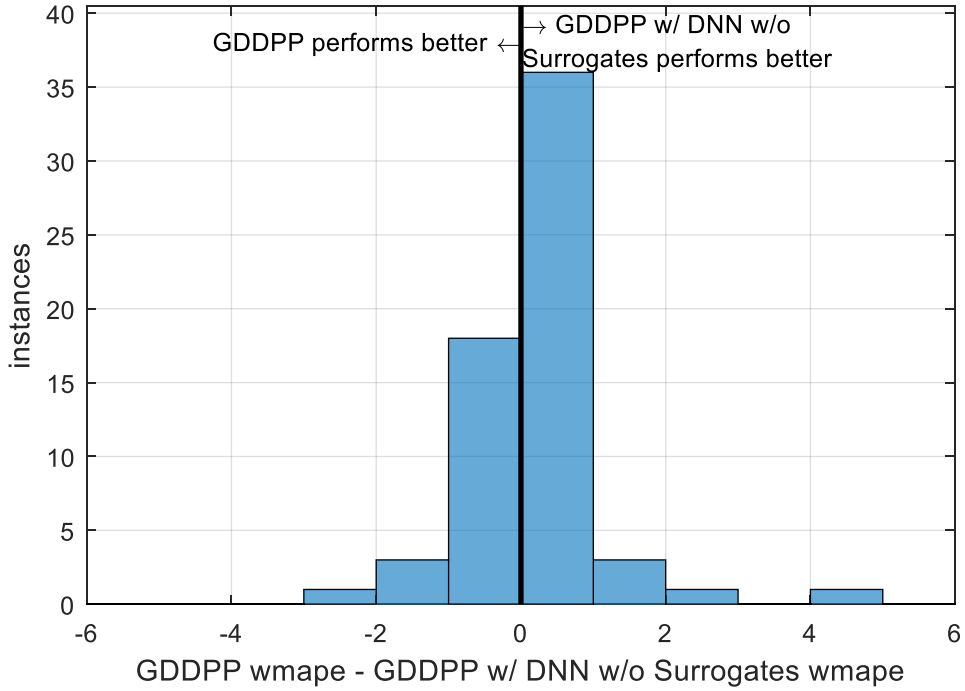

Figure A-35: First non-heating days.

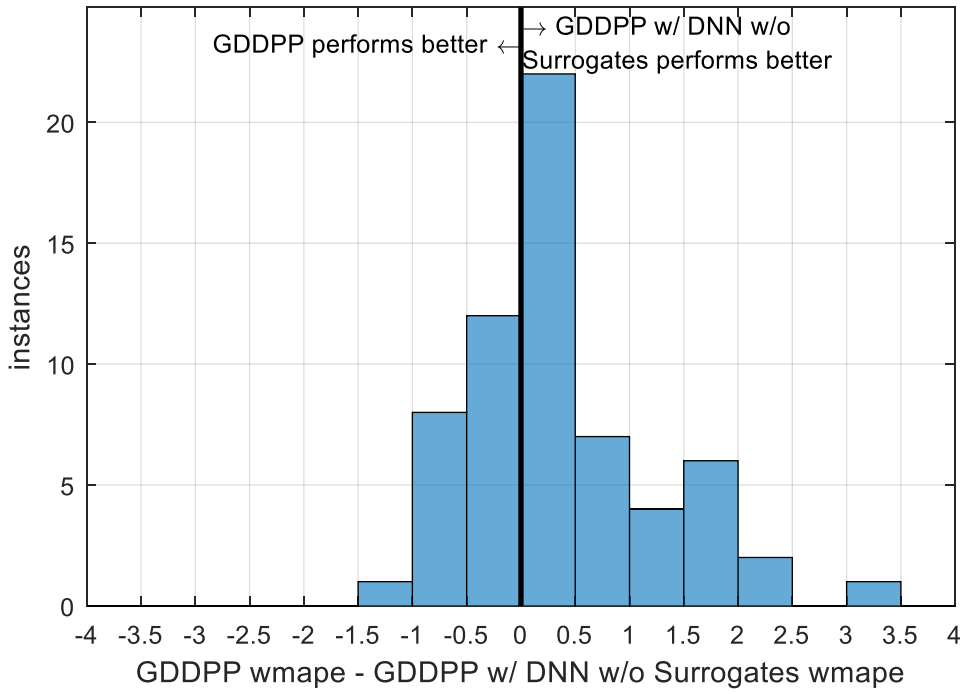

Figure A-36: First heating days.

A.7 Unusual days graphs for Section 5.3 comparing the current GasDay ensemble to the ensemble with a DNN component using 40,000 surrogates

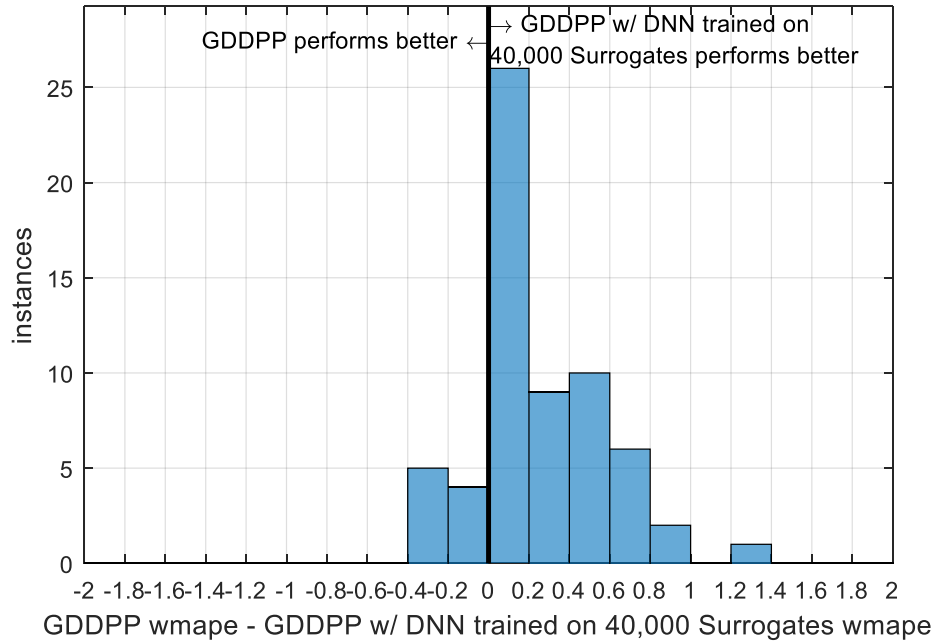

Figure A-37: Coldest days.

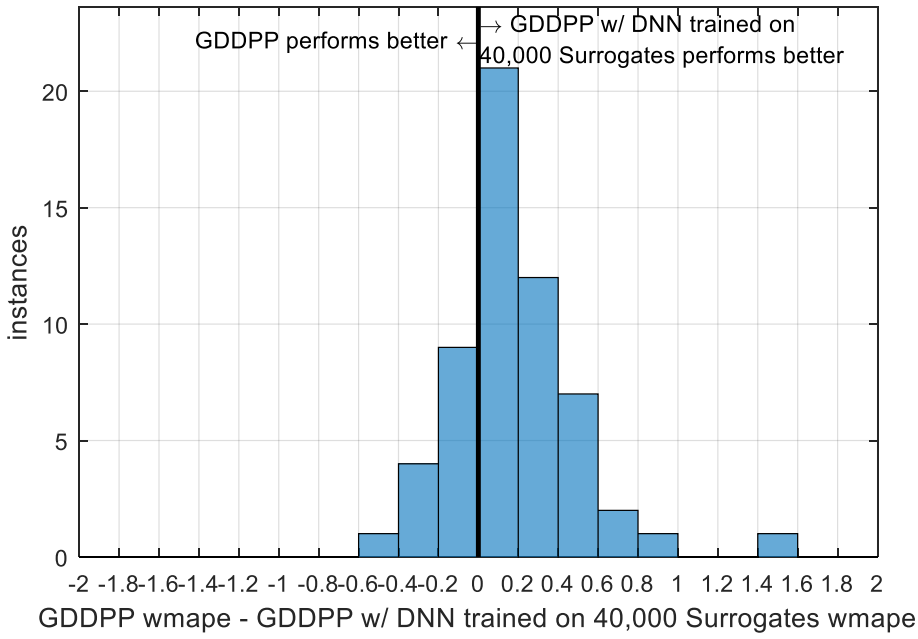

Figure A-38: Colder than normal days.

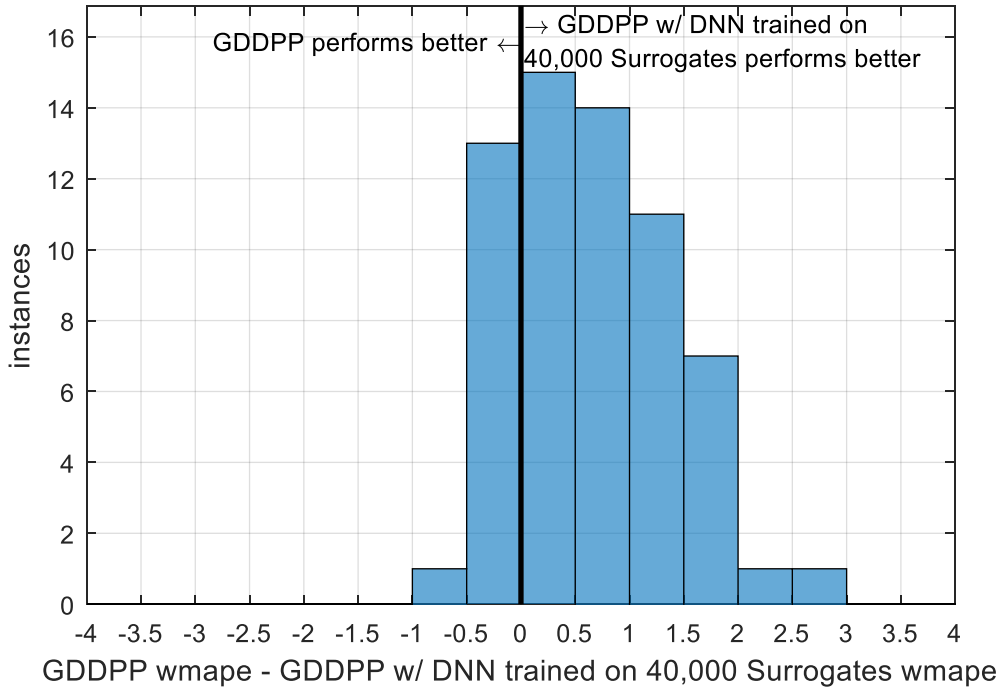

Figure A-39: Warmer than normal days.

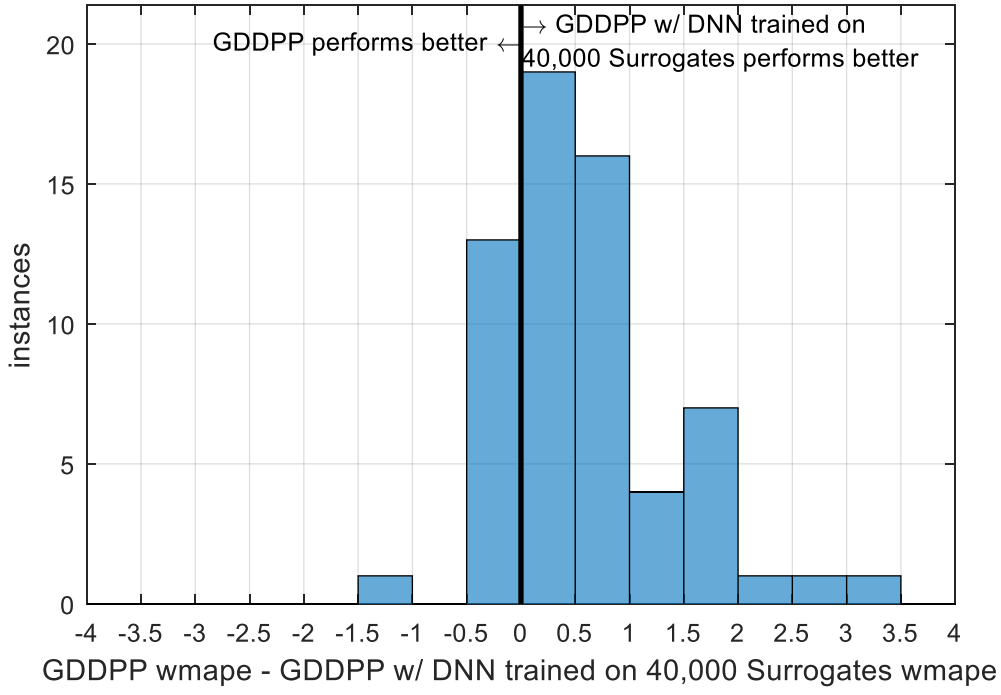

Figure A-40: Windiest days.

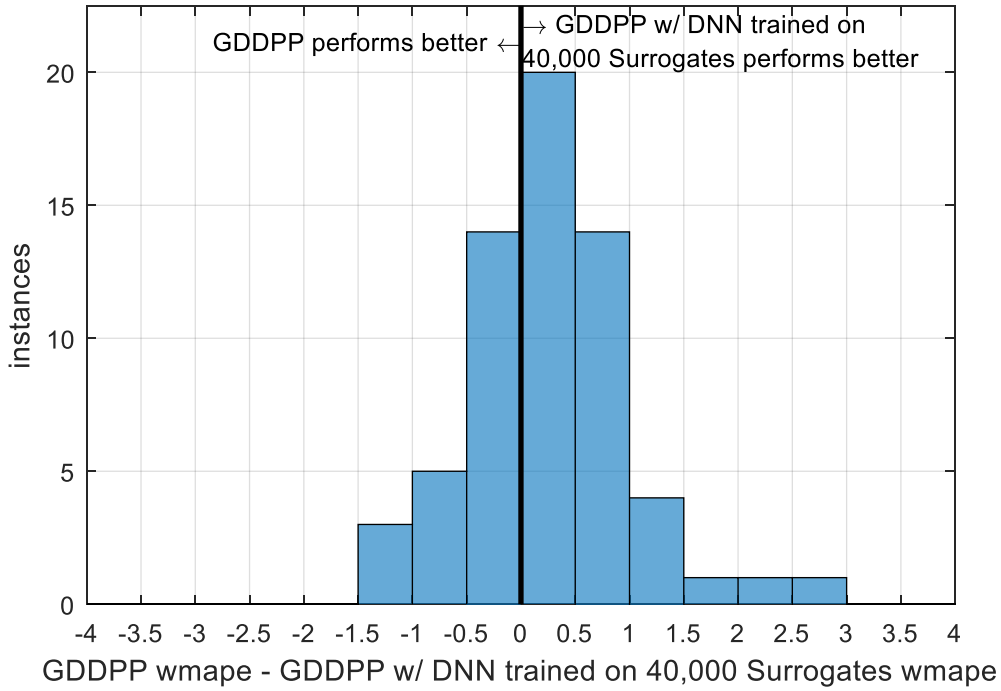

Figure A-41: First non-heating days.

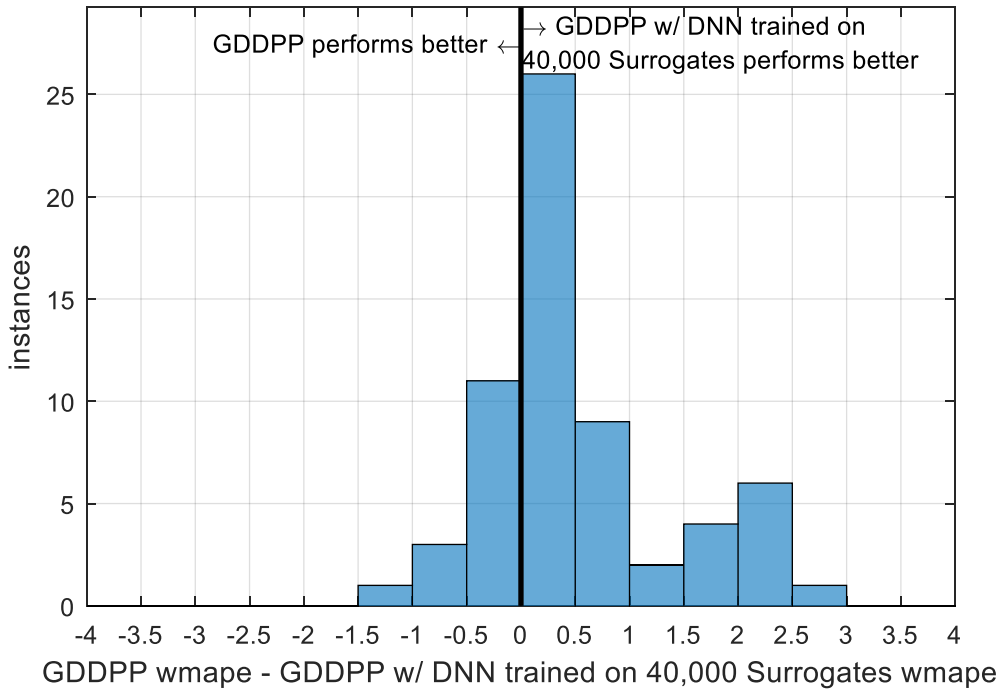

Figure A-42: First heating days.

A.8 Unusual days graphs for Section 5.3 comparing the ensemble with a DNN component using 0 surrogates to the ensemble with a DNN component using 40,000 surrogates

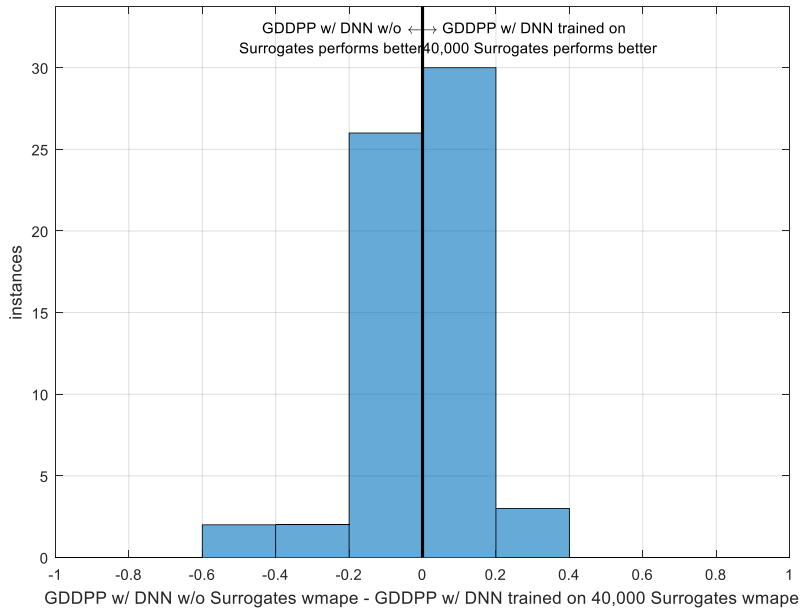

Figure A-43: Coldest days.

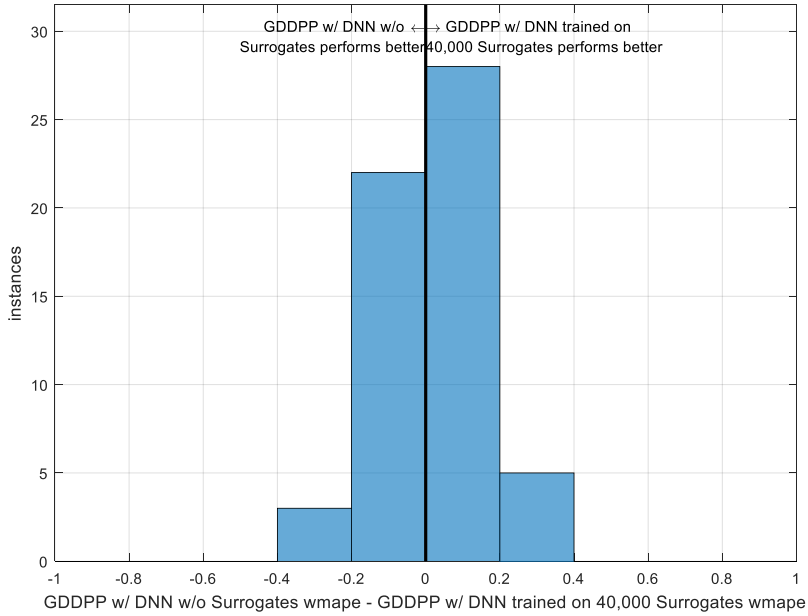

Figure A-44: Colder than normal days.

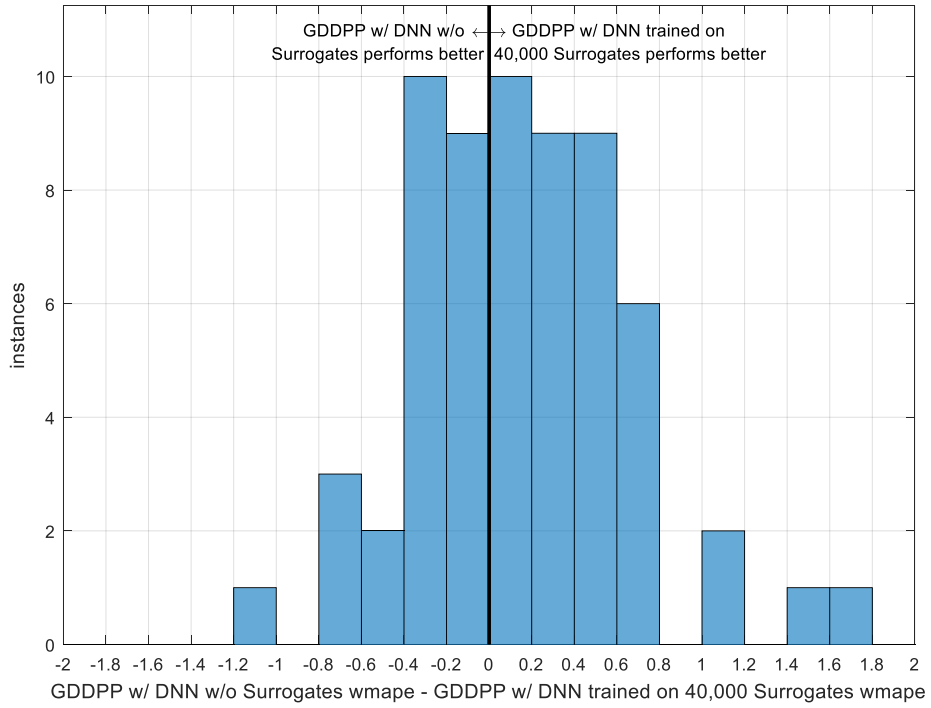

Figure A-45: Warmer than normal days.

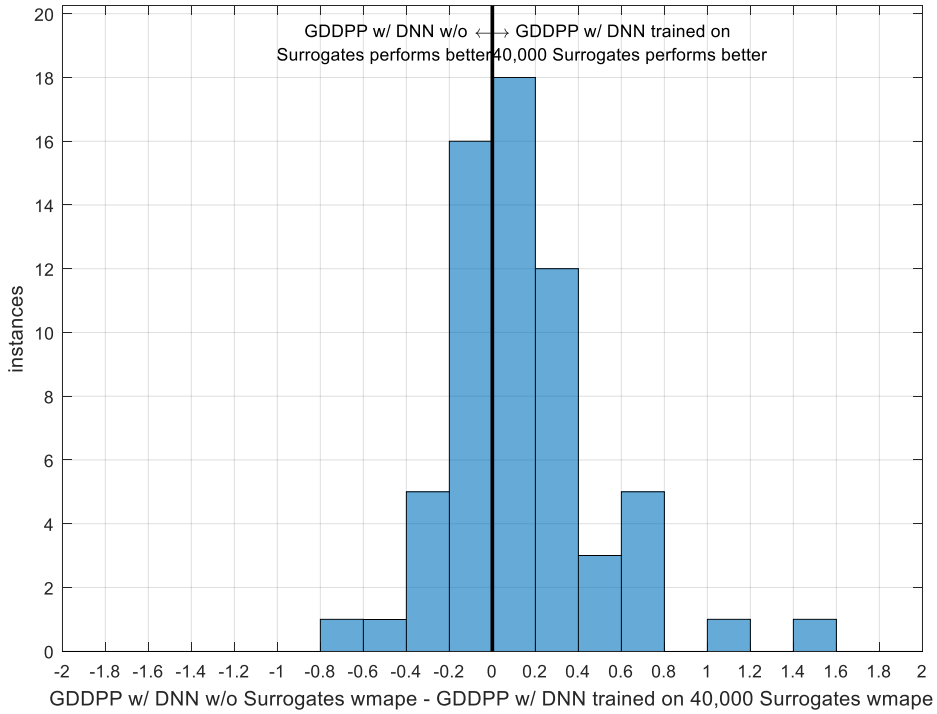

Figure A-46: Windiest days.

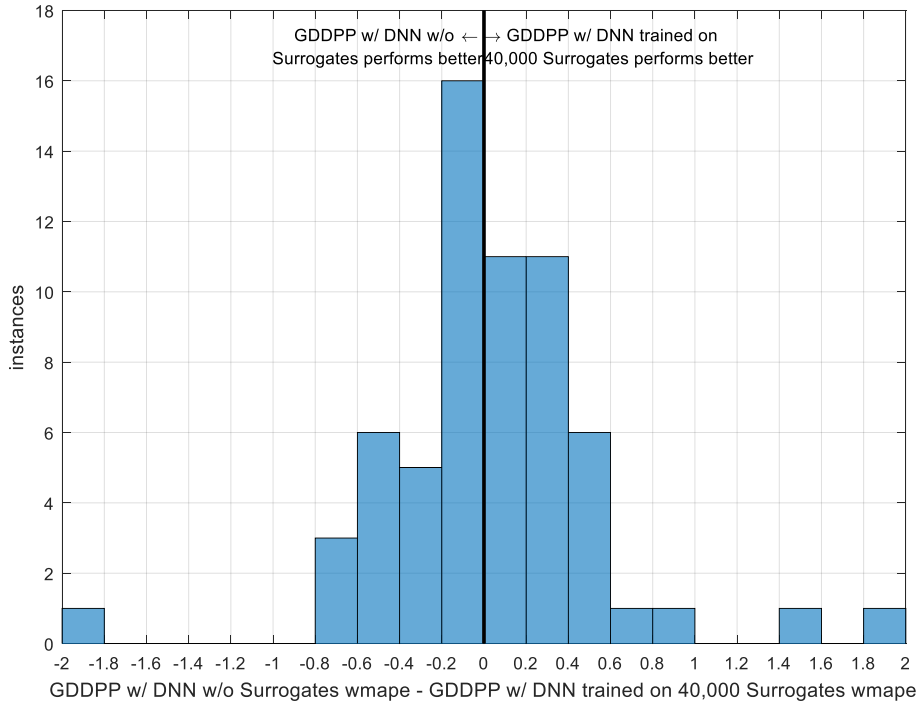

Figure A-47: First non-heating days.

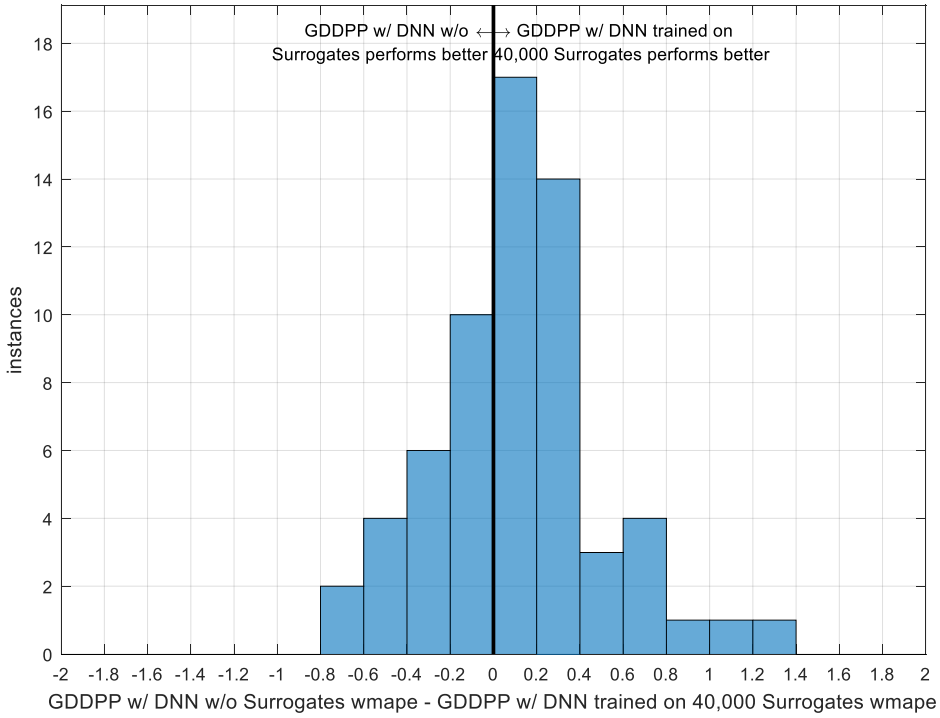

Figure A-48: First heating days.

APPENDIX B

Unusual Days

Unusual days are days that are either especially difficult or critically important to forecast natural gas demand well. For instance, the first heating days of the heating season and the first non-heating days after the heating season are not as critical to forecast well, but they are difficult to forecast because they usually have large swings in temperature. In contrast, the coldest days are not difficult to forecast well, but it is important to have accurate forecasts on those days because of the magnitude of the forecast. The rest of this Appendix describes how each of the unusual day types are determined.

The coldest days are simplest; they are the 18 days (5%) in a year with the lowest temperature. The windiest days are also simple. They are the 11 heating days (3% of heating days) with the highest wind speeds in a year.

Colder than normal days and warmer than normal days requires an understanding of what normal weather is. For the sake of this thesis, normal weather can be referred to as the expected temperature based solely on day of the year. Therefore, the colder than normal days are the 18 heating days in a year with temperatures farthest below normal, and the warmer than normal days are the 11 heating days in a year with temperatures farthest above normal. Often, there is heavy overlap between colder than normal days and coldest days.

Colder than yesterday days and warmer than yesterday days are the 11 heating days with the greatest decrease in temperature from the day before and the 11 heating days with the greatest increase in temperature from the day before, respectively.

Finally, the first heating days are 18 days that occur after the temperature dips below the heating degree day reference temperature described in Section 1.4. Meanwhile, the first non-heating days are 18 days that occur directly after the temperature has risen above the heating degree day reference temperature.

BIBLIOGRAPHY

[1] *Natural Gas Explained*. https://www.eia.gov/energyexplained/index.cfm?page =natural_gas_home.

[2] *Understanding Natural Gas Markets*. http://www.spectraenergy.com/content/documents/SE/Fact_Sheets/Understanding_Natural_Gas_Markets.pdf.

[3] J. G. Asbury, C. Maslowski and R. O. Mueller, "Solar availability for winter space heating: An analysis of the calendar period 1953-1975," United States, 1979.

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

[5] N. R. Draper and H. Smith, *Applied Regression Analysis.* (3. ed. ed.) 1998.

[6] B. I. Ishola, "Improving Gas Demand Forecast during Extreme Cold Events." ProQuest Dissertations Publishing, 2016.

[7] 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.

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

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

[10] R. Hecht-Nielsen, "Theory of the backpropagation neural network," *Neural Networks,* vol. 1, pp. 445, 1988.

[11] T. L. Ruchti, R. H. Brown and J. J. Garside, "Kalman based artificial neural network training algorithms for nonlinear system identification," *Proceedings of 8th IEEE International Symposium on Intelligent Control*, pp. 582-587, 1993.

[12] R. T. Clemen, "Combining forecasts: A review and annotated bibliography," *International Journal of Forecasting,* vol. 5, *(4),* pp. 559-583, 1989.

[13] R. Brown *et al*, "Forecasting by Tracking and Combining Models," *37th Annual International Symposium on Forecasting,* 2017.

[14] 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.

[15] C. Szegedy *et al*, "Going deeper with convolutions," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[16] 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.

[17] X. Qiu *et al*, "Ensemble Deep Learning for Regression and Time Series Forecasting," *Ciel*, 2014.

[18] E. Busseti, I. Osband and S. Wong, "Deep learning for time series modeling," *Stanford University*, 2012.

[19] M. Dalto, J. Matusko and M. Vasak, "Deep neural networks for ultra-short-term wind forecasting," *Proceedings of the 2015 IEEE International Conference on Industrial Technology (ICIT)*, pp. 1657-1663, 2015.

[20] T. Kuremoto *et al*, "Time series forecasting using a deep belief network with restricted Boltzmann machines," *Neurocomputing,* vol. 137, pp. 47-56, 2014.

[21] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks." *Aistats*, pp. 249-256, 2010.

[22] J. Kennedy, "Particle swarm optimization," *Encyclopedia of Machine Learning*, Springer, pp. 760-766, 2011.

[23] S. Ryu, J. Noh and H. Kim, "Deep neural network based demand side short term load forecasting," *Energies,* vol. 10, *(1),* pp. 3, 2016.

[24] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends® in Machine Learning,* vol. 2, *(1),* pp. 1-127, 2009.

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

[26] Y. LeCun *et al*, "A tutorial on energy-based learning," *Predicting Structured Data,* vol. 1, pp. 0, 2006.

[27] L. K. Hansen and P. Salamon, "Neural network ensembles," *Tpami,* vol. 12, *(10),* pp. 993-1001, 1990.

[28] P. E. Kaefer *et al*, "Using surrogate data to mitigate the risks of natural gas forecasting on unusual days," *35th Annual International Symposium on Forecasting*, 2015.

[29] A. S. Weigend, D. E. Rumelhart and B. A. Huberman, "Generalization by weight-elimination with application to forecasting." *Nips,* pp. 875-882, 1990.

[30] Stuart Geman, Elie Bienenstock and René Doursat, "Neural Networks and the Bias/Variance Dilemma," *Neural Computation,* vol. 4, *(1),* pp. 1-58, 1992.

[31] N. V. Chawla *et al*, "SMOTE: Synthetic Minority Over-sampling Technique," 2011.

[32] I. Goodfellow *et al*, "Generative adversarial nets," *Advances in Neural Information Processing Systems,* pp. 2672-2680, 2014.

[33] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems,* 2012.

[34] Y. Kim, "Convolutional Neural Networks for Sentence Classification," 2014.

[35] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2014.

[36] T. N. Sainath *et al*, "Deep convolutional neural networks for LVCSR," *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

[37] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.,* vol. 9, *(8),* pp. 1735-1780, 1997.

[38] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Jproc,* vol. 78, *(10),* pp. 1550-1560, 1990.

[39] Y. Bengio, P. Simard and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *Tnn,* vol. 5, *(2),* pp. 157-166, 1994.