Machine Learning for Public EV Charging Station Forecasting

Alexander Orzechowski

Department of Electrical & Computer Engineering
McGill University
Montréal, Québec, Canada

August 2021

A thesis presented for the degree of Master of Science

Abstract

The year-after-year increasing popularity of electric vehicles will cause unprecedented issues in managing the supply of electricity and charging spaces. It is in the interest of utility providers and everyday consumers to be able to plan for peaks and congestion. While past work has been done for localized, short-term forecasting, it has not included longer term forecasting, spatial diversity, while also lacking an exploration of features. This thesis proposes a methodology to forecast demand at public EV charging stations. We explore the potential of data-driven models to predict demand for the medium term or up to one week in advance. Moreover, we take into account not only the temporal-spatial relationship between an EV and a station, but also between different stations. To the best of our knowledge, this is the first study to propose machine learning to forecast medium-term public EV charging demand, to exploit weather and other features, and to study different stations and networks. The method was validated using data from eleven stations over three years from Scotland, UK. The prediction horizon is seven days. Our method predicts demand of a whole network with a MAE of 124.7kWh and SMAPE of 5.9%.

Abrégé

La popularité grandissante des véhicules électriques va provoquer des problèmes sans précédent pour la gestion de l'approvisionnement de l'électricité et des bornes de rechargement. Il est ainsi de grand intérêt pour les consommateurs mais aussi pour les fournisseurs de service de pouvoir gérer la congestion et les pics d'utilisation de ces bornes. Tandis que des études sur la prédiction locale à court terme de ces bornes ont été auparavant réalisé, celles-ci ne prennent pas en compte la prédiction à long-terme et la répartition géographique de ces bornes ainsi que d'autres caractéristiques qui leur sont associées. Nous explorons la construction de modèles dirigés par les données permettant la prédiction de la demande d'électricité à moyen-terme, jusqu'à une semaine en avance. Non seulement la relation spatio-temporelle entre un véhicule électrique et une station de recharge est prise en compte, mais aussi la relation entre différentes stations. Il s'agirait à priori de la première étude utilisant l'apprentissage automatique pour ce type de prédiction, exploitant les données météorologiques, le réseau entre les stations ainsi que d'autres caractéristiques. Les données récoltées sur une période de trois ans de 11 bornes de recharge situées en Écosse ont permis la validation de cette méthode, avec un horizon prévisionnel de sept jours. Notre méthode permet de prévoir une demande du réseau électrique avec un MAE de 108 kWh et un SMAPE de 5.9%.

Acknowledgements

My experience working on this thesis has been immensely educational and transformative. I could not have had better people to teach and guide me over the past two years. I owe it many for their wide-ranging support. First and foremost I would like to express my appreciation for my supervisor Professor Brett Meyer. I could not imagine having better supervision given the continuous inspirational and cultivating discussions during our work. I am deeply grateful to the co-collaborators and lab members whose work was pivotal in making my work come together, namely Mr. Loren Lugosch, Mr. Hao Shu, and Mr. Raymond Yang. I would like to extend my sincere thanks to all the members of an industrial partner, Mogile Technologies Inc, who met with my lab and provided domain expertise needed for my work. Lastly, my appreciation extends to the Natural Sciences and Engineering Research Council of Canada (NSERC). Without their generous financial support it would not be possible to complete the work achieved.

Contribution of Authors

In chapter 3 Raymond Yang assisted with the processing of the dataset and provided plots of the data analysis. Hao Shu and Raymond Yang assisted with the software tools for subsection 4.1.6. All other writing and intellectual contributions are my own original work.

Contents

List of Figures

List of Tables

List of Acronyms

ANN	artificial neural network.
API	application programming interface.
ARIMA	autoregressive integrated moving average.
CNN	convolutional neural network.
EU	European Union.
EV	electric vehicle.
GCN	graph convolutional network.
GPU	graphical processing unit.
ICE	internal combustion engine.
kWh	kilowatt-hour.
MAE	mean absolute error.
ML	machine learning.
MSE	mean squared error.
POI	point of interest.

PSF	pattern sequence-based forecasting.
RAM	random access memory.
ReLU	rectified linear unit.
RMSE	root mean squared error.
SGD	stochastic gradient decent.
SMAPE	symmetric mean absolute percentage error.
V2b	vehicle-to-building.
V2G	vehicle-to-grid.

Chapter 1

Introduction

Electric Vehicles (EVs) will be deployed in large scale in the next one to two decades [1], [2]. However, the electric grid and cities are not configured for widespread EV use, so changes are needed in a wide array of domains. Utilities providers will implement pricing models to incentivize customers to charge at non-peak times. Vehicle-to-grid and vehicle-to-building (V2G/V2B) technology is being studied to allow for grid flexibility. Power grid infrastructure is also being upgraded [3]. Meanwhile, consumers and utilities alike would benefit from demand forecasting: consumers, to avoid wait times at charging stations; utilities, to address short- to long-term power needs.

The ability to predict demand at public charging stations helps providers avoid higher costs of energy and excessive risks to grid disruption. Prediction can also help to alleviate traffic congestion, and decision making when EVs participate in the electricity trading market

[4]. Additionally, the scarcity of charging resources and waiting times are expected to increase due to increasing public charging demand as seen in charging data [5]. Demand prediction can further improve the operational efficiency of utilities providers because supplemental plants may need to be brought online during peak load periods.

Our objective in this thesis is to build a machine learning model to predict the demand at various stations in a charging network, as well as for the network as a whole, to support both consumers and utilities. Given past charging sessions, the model will predict future usage up to seven days in advance. Utilities providers will need to be able to dynamically adjust the pricing, and possibly power, of charging stations to influence when and where drivers charge their cars in order to not overload the electrical grid. In the meantime, drivers may wish to adjust their plans to avoid congestion, and therefore reduce waiting time. Our modeling approach addresses these needs.

Predicting public charging behavior is challenging for a few reasons. The first is because of the random nature of EV charging in a public setting. Specifically, Figure 3.1 (d) shows the distribution of energy values that can be requested at a public charging station, where energy demand varies from 1kWh to 40kWh (approximately 5km to 200km of driving range). Second, there is little consensus on the appropriate features to use to predict demand. Some studies rely on personal information. Furthermore, features such as holidays, weather, and activity at nearby stations have not been studied, despite their statistical significance [6]. There is also disagreement about whether energy, power, waiting time, or charging time, are the most appropriate metric to predict, as well as over what period of time [4], [6].

Load forecasting in power systems is well established and been characterized using a few different time horizons: short-, medium-, or long-term. Short-term load forecasting (STLF) includes a half-hour to 24 hours ahead. Medium-term (MTLF) ranges from one day to one year in advance, while long-term load forecasting (LTLF) is from one to ten years. Forecasting in a power grid is concerned with supporting decision making at the granularity of hourly, daily, weekly, and monthly changes in load, according to Gross and Galiana [7]. Load can be defined as the system load, peak system load, and the system energy [8]; we focus on predicting system energy demand at daily granularity.

Prior research has made progress on the challenge of forecasting for public charging stations. For example, in some studies data was for a car sharing platform, with data associated to registered users, meaning user-specific data could be leveraged. In contrast, others were for residential or university campus data. Overall, these works generally study short term demand [9, 10]. Our work extends to the medium-term demand for energy at public charging stations.

Previous works have primarily looked at machine learning (ML), and more rarely statistical methods. Most ML studies have used only past data, for example using past power consumption data to predict future power consumption [4, 11]. Some works have relied on traffic to predict the demand, or tracking and user driving behavior [4, 12]. All together, the related studies do not address the problem sufficiently because they did not consider public charging stations, medium to long-term forecasting, and a study of relevant features.

Lastly, previous research has been limited by the assumptions made the about historical data used to train models. Specifically, some work used simulated data. However, real-data studies are important because simulated charging data cannot capture all the relationships and stochastic effects of real data [3]. Methods for modelling demand at public EV charging stations rarely appear in the literature, primarily due to a lack of data from utilities [3]. Additionally, some studies assumed the availability of user-level behavioral data. While such data makes it easier to answer certain questions, trends toward protecting user privacy means that such data may not generally be available. Consequentially, our study focuses less on the question of when will the outlet currently being used be free, and more on what will be the aggregate demand at charging stations and networks, since the former is extremely stochastic in nature and has only been studied with user behavior data [12, 13].

In this thesis, we propose a robust ML pipeline that, given anonymized public charging station usage data, (a) constructs and labels a dataset consisting of daily station energy usage, (b) extends it using weather and other data sources, and (c) explores the impact of different time horizon parameters as well as different output layer structures. We evaluate machine learning and other methods for the prediction of daily demand at individual stations and in the whole network. We also study the potential of multi-task learning (MTL), and observe that when a single model predicts demand for individual stations as well as the entire network, accuracy improves. Using a dataset with over 66,000 charging events, an ANN model achieves a SMAPE of 5.9% for the network, and 21.3% for individual stations, on average.

Chapter 2

Related Work

This section details the related work by first explaining some of the general work done with ML and electric vehicles. Then in the next section we explain the work specifically for data-driven charging station forecasting, by splitting the work into non-ML and ML categories.

2.1 ML in the EV Domain

There is plentiful work surrounding machine learning for EVs. Much is concerned with the car itself. Some studies aim to optimize how energy is used on board the vehicle. Others predict range or maximize it by optimizing the EVs route considering charging placement. Many contributions have been to power management onboard an EV or at the charging stations. Some more work uses ML to detect anomalies in the EV charging schedule caused by misinformation inserted by an attacker [14]. A sizable amount of work optimizes the charging schedule of a fleet or set of EVs, as in [15]. Here, there is a set of vehicles that needs charging and an optimal schedule is desired given the charging facilities. So in general there is much data concerning EVs and accordingly, many areas to exploit learning.

2.2 Predicting Future Charging Demand

In the past seven years, there have been several studies on the application of forecasting demand at public EV stations (the focus of this thesis). Primarily, these methods typically categorize as probabilistic or machine learning. Also, we differentiate these works by their application area, e.g. public, workplace, or residential charging. In this section, we group works by their scope and summarize the contributions.

2.2.1 Non-ML Methods

Some earlier work has engaged statistical methods for predicting demand. This work in [16] fitted a day load curve to a statistical model based on actual measurement data. Chung et al. [17] used a kernel density estimator to predict EV user behavior on a college campus. [3] attempt to find the probability distributions for EV-related random variables. These works are early studies and do not establish if statistical models can be used to predict demand at charging stations.

2.2.2 ML Methods

The study in [10] employed machine learning (KNN) to predict the aggregate demand up to one day in advance at a few outlets on a university campus. The same university campus was studied in [18] which proposed a KNN combined with a new method known as pattern sequence-based forecasting (PSF). PSF is a recently proposed time-series forecasting algorithm that found success in energy price forecasting. Similarly, the work in [19] also studied the demand for the next day and examined a few models moving towards an ensemble learning method, with the benefit of ensemble learning ranging from less than one percent to eight percent. Also, their method was demonstrated only on the network demand and only in one city. Overall, these studies [10, 18, 19] only study demand for the next 24h at a single station or outlet and do not consider relevant features. Using no relevant features means models rely on the assumption that only past charging demand causes future charging demand.

In [9], Luo et al. considered relevant features but only tests on a vehicle sharing platform for instant demand in the next 24h. The study focuses on continuous expanding electric vehicle sharing systems. The model invented uses LSTMs and Dynamic GCN to output thedaily expected (mean) instant future demand at individual stations. Interestingly, pairwise correlations between stations were expressed as a graph, POIs and categorical weather were encoded, and the model can predict demand for a planned station (where historical data is unavailable).

Studies that predicted the demand in terms of specific EV users or non-anonymized data (user behavior) include [12,13]. For instance [13] used a linear mixed-effects model and found that the median energy consumption in a user's charging session history should be used to estimate demand. The study in [12] studied 252 EV users and proposed predicting a user's stay duration and energy consumed with a combined statistical and ML approach. These studies generally assume data such as how long a user drove since their last charge, average kilometers driven per day, or even user ID. It is true that we do not know what the future standard of data anonymity will be. However, limiting assumptions on what data is available is important for privacy and also for reconstruction of results. So if the same results can be achieved with without driving behavior data it would be preferable.

Xydas et al. in [6] evaluates the impact of EV charging in a region on the distribution network in the near future, yielding categorical output known as a risk factor. It presents a modeling framework but does not demonstrate the accuracy of the approach. Similarly, Straka et al. [20] and used non-anonymous data from the Netherlands to forecast the popularity of charging infrastructure, or more precisely, forecasting whether a given candidate site belongs to the top rank candidate sites or not.

Zhang et al. [4] first predicted traffic flow, i.e., the number of vehicles passing a point on a highway per hour, with a CNN then predicted the demand at one station on the side of a highway with a queuing probabilistic model. A MAPE of less than 25% was achieved for demand in the next day, but the study assumes the availability of traffic flow and personal data (e.g. distance traveled before the station, daily travel distance, etc) and only does immediate short-term predictions, meaning less than one day in advance.

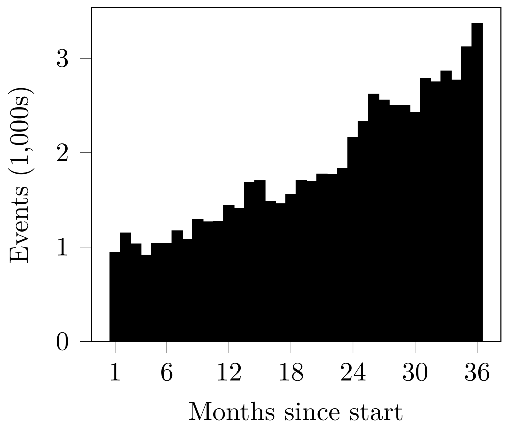
Chapter 3

Charging Events

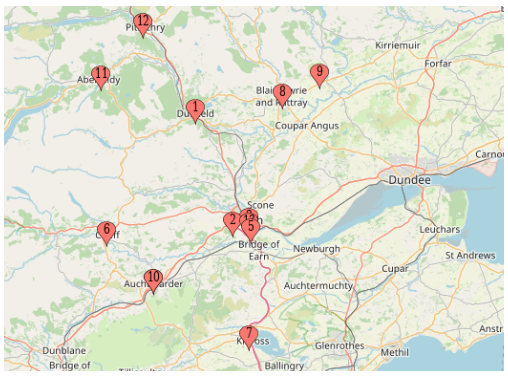
The approach considers charging data [5] with 66,664 charging events in a county in Scotland, UK. The county spans an area of 5,286 km^2. The data covers charging events over three years from September 1, 2016 to August 31, 2019. The distribution of the charging data over the three years is shown in Fig. 3.1 (a), which shows rapid year-over-year growth of EVs.

3.1 Charging Stations

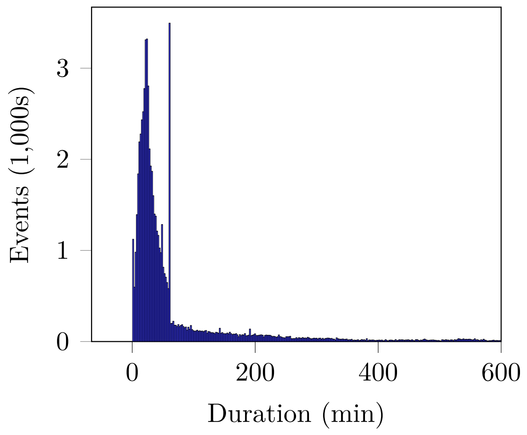
There are 13 stations in the dataset. Two stations are dropped because they have little charging activity, having fewer than 400 charging events. The remaining eleven stations all have greater than 1,000 events. We define a station as a charging location with a unique longitude and latitude (+/- a margin of 25 meters). So charging locations in the near vicinity of each other, such as in a parking lot, are grouped. Each station is one or more charging outlets or charging station identifiers. The location of the 13 stations are mapped out in Fig. 3.1 to show the spatial characteristics of the network. Last, we define a charging network as multiple charging stations.



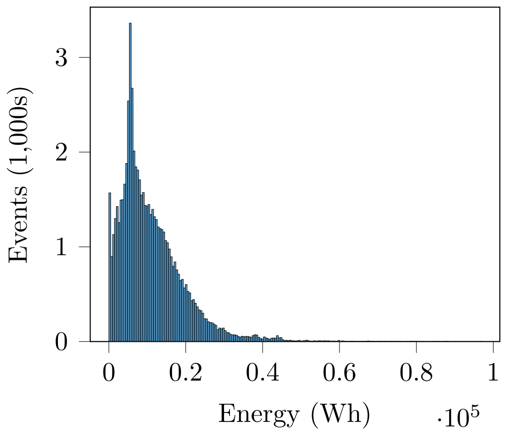
(a)



(b)



(c)



(d)

Figure 3.1: (a) Distribution of charging data of time since start of collection (b) Locations of the 13 stations on map (c) Charging event distribution of duration (d) Charging event distribution of energy consumed

3.2 Charging Port

There are three types of charging ports in the dataset, though some stations may only feature one or two. The CHAdeMO port is a DC port generally used in North America and Japan; the CCS port is also a DC port more commonly used in North America; the Type 2 port is an AC port generally used in Europe. Depending on the region and vehicles, some ports may be more in demand compared to others.

The charging levels in the network are all level 2 (fast) or level 3 (rapid). The power at level 2 chargers are either 7, 22, or 43kW AC. The power at level 3 stations is 50kW DC. Some stations offer no cost charging while others are paid.

3.3 Charging Details

Charging events are characterized by the starting time, charging duration in seconds, total energy consumption in watt hours. Both the average and peak power consumption are not available in the dataset but can be estimated from the energy and duration values. However, we do not handle power either as a dependant or independent variable, so no power values are considered.

The distribution of energy and duration is shown in Figure 3.1 (c) and (d). As shown, most events consume about 13kWh and most people stay at the station for less than one hour. One anomaly in the dataset is that two percent of the events have zero or near-zero energy.

Another is that about the same percentage of events have zero duration. The only other anomaly is that in Figure 3.1 the duration distribution is very different from that of energy. This is because duration in this sense means how much time the vehicle was connected to the port, not how much time active charging occurred. To be specific, normally energy is directly proportional to time via the $E = Pt$ relationship. Since the target we primarily work with in the study is aggregate energy, these anomalies do not influence the results.

Chapter 4

Methodology

Here we propose a solution to the problem of predicting the demand at public EV charging stations for up to a week in the future (short to medium term). Our methodology takes input from real-world data, prepares the data for input to models, trains machine learning algorithms, and outputs forecasts of the energy consumption.

The solution includes a ML prediction pipeline as follows in Figure 4.1. The charging data from section III is cleaned and split into sets for training, validation and testing. Then, the feature extraction and labeling module turns the charging data into a dataset. Then, normalization is performed to put features and labels on the same scale. Next a prediction model (e.g. ML model) is trained on data (excluding the test set). Finally, the inverse of the normalization is performed which gives the prediction of future demand.

The core of our method is the feature extraction and labeling module, while the other components of the pipeline are mostly standard. As such, we split the methodology sections into two parts. The first for feature extraction and labeling. The second for the model that actually makes predictions. Next we detail the feature extraction module. We discuss the numerous steps taken to develop the module, including timestamp generation, parameter selection, and feature engineering.

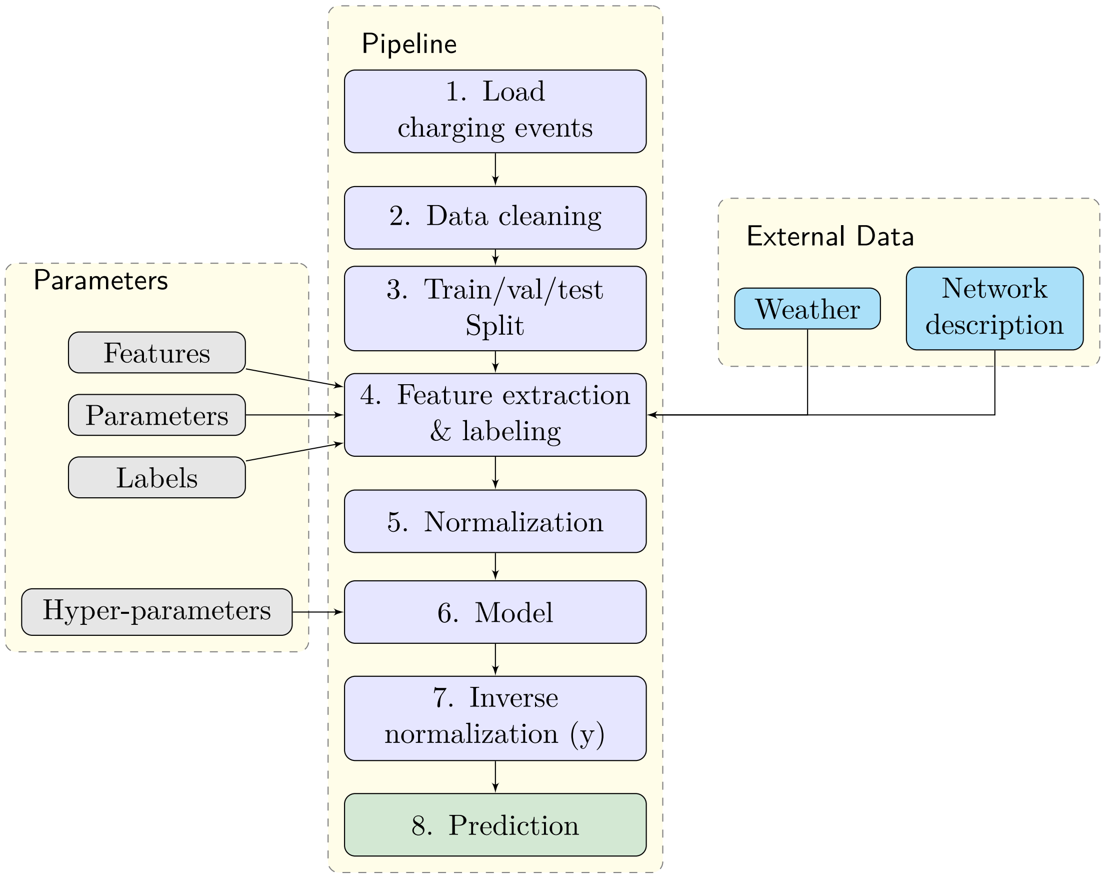


Figure 4.1: ML development pipeline

4.1 Feature Extraction & Labeling

For the charging data presented, and often in machine learning, the dataset does not come with labels, and must be labeled by the researcher. With a labeling strategy the dataset will have well defined attributes $\mathbf{x}$ and label $\mathbf{y}$. Our strategy centers around extracting as many features that correspond to each time as necessary.

4.1.1 Timestamps

Timestamps generation is one component used to generate training examples and label data. Each timestamp is a reference time by which some past data (e.g. demand) is grouped with some future data, and each corresponds to a training example. In other words, each timestamp takes a snapshot of what charging was occurring at a particular time.

Timestamps are used to generate the labeled dataset, and the number of timestamps is chosen by the researcher. The timestamps are uniformly distributed resulting in the distance between timestamps being:

$$dist_{ts} = \frac{E_{newest} - E_{oldest}}{N_{ts}} \tag{4.1}$$

where E_{newest}, E_{oldest}, and N_{ts} are the date of the newest event in the dataset, the date of the oldest event in the dataset, and the number of timestamps. Notably, we do not assign more weight to time periods with more charging because low activity times are just as important to predict as high activity times.

4.1.2 Prediction Delay

Secondly, prediction delay is another component of the feature extraction module, and arises in the following situation. Say we are interested in the tomorrow's demand, and it's 9:00 today, but we are interested in the demand tomorrow for the next 24 hours starting at 12:00. Then, the first day of prediction starts 3 hours from now and ends 27 hours from now. The possibility of shifting forward in the prediction horizon without changing the model leads to the property of prediction delay.

Prediction delay is one of the parameters used when making a labeled dataset. It is inserted into the validation and test sets, and subsequently, the training set. In other words, it is a time gap inserted between the time the features we collected and the time labels were collected. Prediction delay adds another challenge to the model because it means the prediction horizon does not always start at the current time. It can be shifted forward an arbitrary number of hours. So overall, prediction delay means predictions can have an arbitrary offset.

The prediction delay is inserted as a feature of each training example. It can be inserted in 4 ways:

1. For each charging event, a random prediction delay is selected from a continuous range of values.

2. For each charging event, a random prediction delay is selected from a discrete range of

values.

3. For each charging event, N random prediction delays are sampled from a continuous range and N corresponding charging events are formed. This would expand the number of entries in the dataset by a factor of N.

4. For each charging event, all possible prediction delays in a discrete range are used. If there are N unique prediction delays within the discrete range, there would be N charging events formed, each with a different prediction delay. This would expand the number of entries in the dataset by a factor of N.

The random prediction delay (option 3) appears the most realistic and challenging for a model, and it expands the data available for training. So, it is inserted as part of the test and validation set. Since the prediction delay is inherently part of the test set, the question that arises is if the delay can be learned from. In our experimentation we saw poor model performance when prediction delay was part of the test set but not the training set. The most likely reason for this is because overfitting can easily occur when there is varriation in the labels, but no corresponding change in the features. As per Figure 4.2, using just one delayed training example per timestamp in the training set was enough for the model to learn the offset. Furthermore, the overall effect was that more delayed training examples was better for model performance, mostly likely because of the simple fact that we are feeding the system more data.

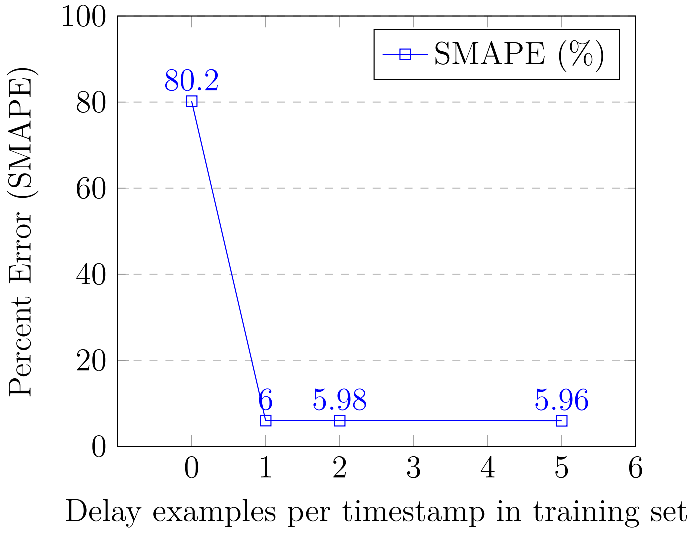


Figure 4.2: Prediction with our model on the test set with and without prediction delay. The test set is held constant at five delays per timestamp.

4.1.3 Parameters

The task of transforming charging events into training data gives rise to several time-series parameters, needed for operation of the prediction pipeline. A summary of these parameters is given in Table 4.1. These parameters can be chosen by simply using domain expertise and experimentation, or by doing a grid-search along possible values. The number of training examples does not need to be grown to very large values. Figure 4.3 shows how the number of timestamps affects the performance of our model, when the parameters of the test set are held constant. The dashed line shows each corresponding error curve of the best baseline. The dashed line is constant because this baseline is a simple average, so it is unaffected by the different parameters. According to the data, a N_{ts} value of two thousand will capture

Parameter	Description
N_{ts}	Number of time-stamps, where each corresponds to a training example
D_{ts}	Number of delays per timestamp
D_h	The depth of history to consider. This means the number of days in the past
D_{wp}	The depth of past weather information to consider
D_{wf}	The depth of future weather information to consider
H	The prediction horizon. How far in the future to consider demand
F	A set of features of size $\|F\|$
L	A set of labels of size $\|L\|$

Table 4.1: Parameters

the behavior of the dataset.

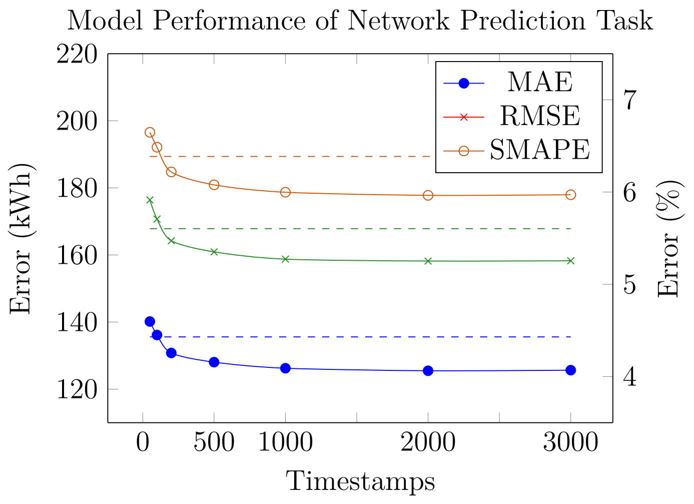
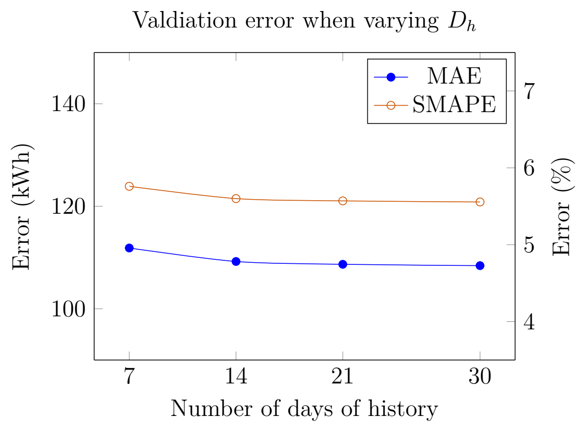


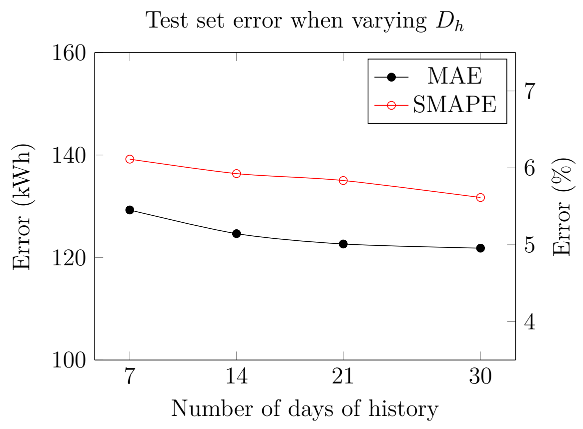
Figure 4.3: Model performance on test set with varying values of the N_{ts} parameter. Error of baseline shown as dashed line.

Similarly, we show how different depths of history affect prediction performance in Figure 4.4. Here validation error is plotted with D_h varying from seven to thirty days.

Prediction error tends to reduce slightly or remain constant as we look at greater depths of history.



(a)



(b)

Figure 4.4: Model performance of network prediction task with increasing depth of history on (a) the validation set and (b) the test set.

4.1.4 Feature Engineering

The goal of feature engineering is to select independent variables that are correlated with the dependant variables, trying to select a feature set that gives high accuracy predictions. Here we present a detailed discussion of features. Because feature fitness cannot be shown solely with statistical data analysis, one strategy taken is to choose features that intuitively might be correlated with increased usage of public charging stations. Features that are not at all correlated can be actually removed or effectively removed through overfitting prevention mechanisms.

Feature	Description	Units
Energy, $E_1, E_2, ..., E_{Dh}$	Past aggregate energy	Wh
Month	Month of year	One-hot-encoded
Day of Week	Day of the week of the training example	One-hot-encoded
Time of Day	The time of day (10:00, 16:00,...) of the training example	Seconds
Calendar Day	Time elapsed since January 1, 1970	Seconds
Time of Year	Time elapsed since January 1 of the current year	Seconds
Prediction Delay	Time buffer between time of features and labels	Seconds
Weather	The local weather data for the time preceding or succeeding the current time	Various, see Table 4.3
Average Duration	The average duration of an event in a timeframe	Seconds
Average Energy	The average energy at a station or network in a timeframe	Wh
Time-averaged Energy	Average Duration * Average Energy	Wh^2

Table 4.2: Features set

A summary of the features used is shown in Table 4.2. For a time-series forecasting application, a natural choice of feature is past data of the target. Since the target is daily energy consumed, we use past energy as a feature, listed in the first row of Table 4.2. Next we include some features that encode time, and as such can represent travel periods such as holidays. Additionally there are a few more demand-based features like event duration. Lastly, we have the past weather data as well as forecasts.

4.1.5 Weather

Interest from utilities on the effect of weather on public EV charging has risen recently. The effects of weather on charging demand should be considered because the range of EVs is less as temperature falls, while charging time increases. Furthermore, [6] established a statistical relationship between weather and charging station demand and hypothesized that in a cold climate such as the UK's, energy requirements increase due to the need to heat the vehicle cabin. Two possible reasons weather is considered important because it affects the performance of the EV and because it stimulates travel. However the primary reason is not known and is of little significance when feeding the information to a model.

Our method considers weather collected by governmental agencies at a central location in the county from an online database [21]. Since the geographical area is not large and the number of weather stations is limited, all stations are assumed to have the same weather. The weather data used for a forecast is shown in Table 4.3.

Here we consider variables that may affect the performance of the vehicle, including temperature. Another component that affects vehicle performance is use of the air conditioning, so variables such as cloud cover are included to account for the fact that the greenhouse effect makes it hotter in the cabin. In general, we use a simple strategy of choosing variables that plausibly effect charging, and checking assumptions later by looking at model performance.

We present our method as a first step towards weather data as an input to a forecasting model. We note that our strategy assumes accurate weather predictions because the weather data here are ground truth values, not predictions. Still, the discrepancy is of little significance because the parameter for the weather data horizon, D_{wf}, is set to three days. Moreover, some noise is implicitly added from the assumption that all cities have the same weather. Using good engineering judgement we keep this parameter at three days without the need for a noise model.

Furthermore, going back and checking our assumption, we obtain the graph in Figure 4.5. Anywhere from one to three days of future weather information provides similar results on our test set. Thus if the weather forecasts in a particular region do not support three days of reliable data, then that number could be reasonably lowered, or again, a noise model that fits the weather patterns could be found.

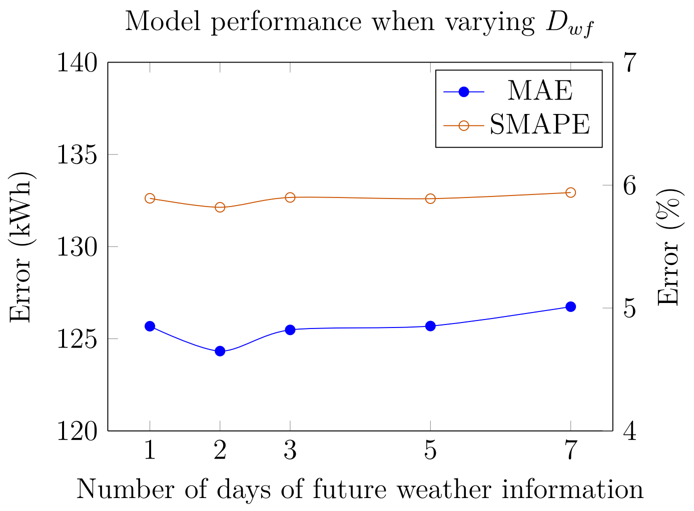


Figure 4.5: Test error of network prediction task with different weather horizons

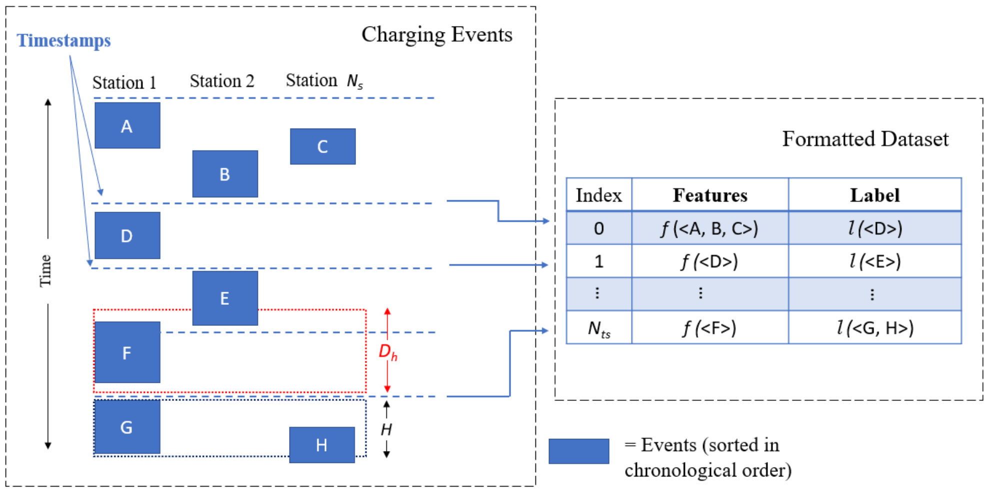
Variables	Description	Units
Max Temp	High temperature on a calendar day	°C
Average Temp	The average temperature recorded on a day	°C
Total Precip	The total precipitation, rain or snow, that fell on a calendar day	mm
Cloud Coverage	The percent of maximum cloud cover. In the range [0, 100] where 100 means there is no clear sky visible	%
Wind	The average wind on a calendar day	km/h

Table 4.3: Weather variable descriptions

4.1.6 Construction of Training Examples

Once a set of features is chosen, a dataset need to be organized so that various algorithms can interpret it. Namely, the charging event data as presented in chapter III cannot be fed to a model. The data in chapter III is organized for keeping track of transactions, not for quantifying past and future demand. Here we detail the fourth step of the pipeline which represents a module that extracts features and labels from charging data.

The general procedure for extracting features is shown in Figure 4.6. The charging data (left) is transformed into a dataset (right). First a set of timestamps is placed throughout the charging data. Then, the timestamps serve as a reference to collect events within certain time frames. The events and attributes in the past form the features and those in the future form the label to finally create one row in the dataset. The depth of history to consider, D_h is illustrated by the height of the red box, and similarly the prediction horizon, H, for the black box. The functions f and l represent functions that take the sum of certain features, e.g. energy, obtaining the aggregate behavior, and format all the features as a vector.



Index	Features	Label
0	f(<A, B, C>)	l(<D>)
1	f(<D>)	l(<E>)
⋮	⋮	⋮
N_{ts}	f(<F>)	l(<G, H>)

Figure 4.6: Extracting features from charging events

4.2 Model

With a labeled dataset, we can deploy learning-based and statistical models. With little to no precedent for this type of prediction, here we openly explore models. The emphasis of the study is on the ANN model because of its capabilities for learning multiple tasks.

4.2.1 ANN

An artificial neural network (ANN) or multilayer perceptron (MLP) [22] is a classic model used for learning a variety of regression or classification tasks from computer vision to speech recognition. While not a traditional time-series forecasting strategy, ANNs have been used for similar types of time-series predictions or load forecasting [19, 23].

Figure 4.7 shows the structure of an ANN for any task, regression or classification. Computing the result at the output layer is done by doing a forward pass of the network.

Nodes can either fire or not fire, according to activation function σ, and their values are a function of the values in the previous layer. The node values during a forward pass are computed using

$$a_k^l = \sigma_k^l \Big(\sum_{i=1}^{n_l - 1} a_i^{l-1} \omega_{ik} + b_k^l \Big) \tag{4.2}$$

where b^l is the bias term for layer l, a_k^l are the node values from layer l, and ω_{ik} are the weight values of layer $i = 0, 1, ..., l$.

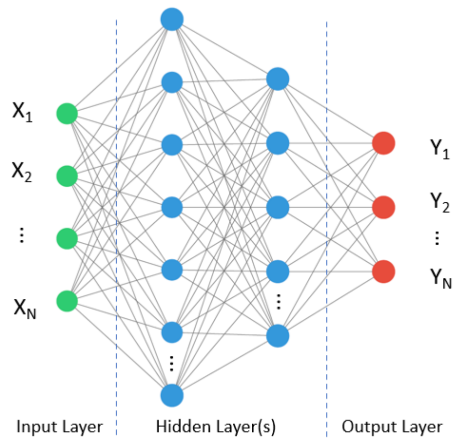


Figure 4.7: ANN structure with 2 hidden layers

There are two neural architectures used which were determined by a grid search:

1. A small net containing two layers of 128 nodes and a dropout later.

2. A large net containing three hidden layers with 2048, 1024, followed by 512 nodes and two dropout layers. This is the net used when studying multi-task learning, described further on in subsection G.

In the dropout layers the probability of zeroing a channel is ten percent. The ANN model and optimization framework is provided by the Pytorch library and the model is trained on a NVIDIA Tesla V100 GPU with 13GB RAM.

4.2.2 ARIMA

The autoregressive integrated moving average (ARIMA) is a traditional model for time-series forecasting, presented by Box and Jenkins. ARIMA is composed of three parameters, ARIMA(p, d, q) where the parameters are:

- p - order of the auto regressive (AR) component.
- d - the number of differences required to make the time-series stationary.
- q - order of the moving average (MA) component.

In short, ARIMA is a statistical model where the time-series is differenced one or more times and one assumes the output is a combination of AR and MA terms. Precisely, the output is given by

$$Y_t = \phi_0 + \phi_1 Y_{t-1} + ... + \phi_p Y_{t-p} + \epsilon_t - \theta_1 \epsilon_{t-1} - \theta_q \epsilon_{t-q} \tag{4.3}$$

where $\phi_0, \phi_1, ...\phi_p$ are the auto regressive coefficients, $\theta_0, \theta_1, ...\theta_q$ are the moving average coefficients, and ϵ_t is the random error at t. The three steps to making a prediction are model identification, parameter estimation and diagnostic checking [24].

Here, the parameters of ARIMA are searched for via the `auto_arima()` API from pmdarima [25]. The search returns the order of each of the three terms and the coefficients.

As seen, ARIMA concerns past data for forecasting, rather than generalizing from a combination of dependent variables. It is commonly applied for short-term forecasting, and more specifically it is commonly applied to financial markets [26], while also performing well at short-term load forecasting in electrical grids [27].

4.2.3 Normalization

Data may need to be put all on the same scale before training starts, depending on the model used. In other words, certain features should not be given more weight because their values are higher. Normalization should be applied when the model includes an optimization method using gradient decent. A normalization routine is applied to the features and labels as

$$z_{norm} = \frac{z - min(z)}{max(z) - min(z)} \tag{4.4}$$

where z is a feature or target along any dimension. The routine works to eliminate bias towards a particular variable by putting data in each dimension on the scale $[0, 1]$. The routine is also applied on the target (y), and as a result any output from the model will appear to be on an arbitrary scale and thus not useful. So, the last step of the pipeline before a prediction is found is to apply the inverse transformation by solving for z.

Here we normalize depending on the features found in the training set (2016 - 2018 data). Note that because charging activity can increase, data is not guaranteed to be on the scale $[0, 1]$. The implication of this is that that model behavior could change to give more weight to larger values if energy values increase significantly in the next months or years. Even though this situation is foreseen with the expected growth in charging, it is overcome by retraining.

4.2.4 Multi-Task Learning

In our approach, we use multi-task learning (MTL) to predict multiple outputs simultaneously. MTL is an inductive transfer mechanism that aims to improve generalization performance. MTL has been shown to be broadly beneficial to neural networks, with the main benefit that overfitting can be reduced and thus accuracy can be improved compared to single-task learning [28].

It is a natural fit to use Multi-task learning when the situation requires obtaining multiple predictions at once. For example, in drug discovery tens or hundreds of active compounds should be predicted, and further, in that domain, [29] found that MTL continuously increases the accuracy as the number of tasks increases. Even in a situation where one only cares about the performance of a single task, the use of auxiliary tasks can increase performance. Commonly, the model learns tasks that are closely related to the main task, allowing the model to learn a beneficial representation [30].

When introducing MTL, the question arises how to figure out what tasks are related. This is what is needed to determine the auxiliary tasks. However, the auxiliary tasks are usually determined in practice, and not known *a priori* [30].

Our approach explores using a single net for a few strongly related tasks. This approach is not new, and has been demonstrated by Sejnowski and Rosenberg in their application to learn both phonemes and their stresses [28, 31]. Our proposed methodology proposes using MTL with a spatial component to predict public EV charging demand. In other words, we are exploiting the charging demand in one location to improve the prediction at another.

Specifically, we use an algorithm to iterate over the different stations. In this algorithm the features are extracted and placed across the columns of the dataset. The stations represent the multiple tasks. The last station visited in this for-loop is a dummy station. The dummy station represents the task of predicting network demand. So the features and target of this station include the sum of the demand at all stations. The model parameters are shared across all tasks, known as hard parameter sharing.

Furthermore, here we discuss how features are extracted in more detail. There are different ways to build the training examples because we have charging behavior at different locations and different times. Thus the question arises how this data can be organized, and how to separate the demand at different stations. This leads to the pseudocode in Algorithm 1 describing the procedure for the construction of training examples. This algorithm helps explain how we incorporate MTL but also further explains how feature extraction works as presented in section 4.1.6.

```
Algorithm 1: Form labeled dataset
  Input: Dataset of charging events with basic charging data for each event
  Output: Labeled dataset for a charging network with $N_S$ many stations
  Initialize: time-series parameters: $D_h$, $D_{wp}$, $D_{wf}$, $N_{ts}$ with list of timestamps
    $list_{N_{ts}}$, $D_{ts}$ with list of delays $list_{Dts}$
  for $t$ in $list_{N_{ts}}$ do
      for $d$ in $list_{D_{ts}}$ do
          $dataset[i][featureIndex] \leftarrow d$
          $dataset[i][featureIndices] \leftarrow$ non-station specific features, labels
            corresponding to time $t + d$
          for $s$=1 to $N_s$ do
              $dataset[i][featureIndices \cup stationIndices] \leftarrow$ station specific features,
                labels corresponding to time $t + d$
          $i \leftarrow i + 1$
```

First, parameters are initialized and a list of of timestamps is generated. Then, we iterate through the list of timestamps, followed by a list of delay values. Next all the non station-specific features are extracted for the given time and appended to the row. Then, we iterate through the stations and append the station-specific features to the training example. We show later how stations can be modeled individually, making the last step optional. The output of Algorithm 1 for each training example can then be written in matrix representation as

$$example_{mat} = \left(\begin{array}{cccc|ccc} x_{11} & x_{12} & \cdots\cdots & x_{1N_x} & y_{11} & \cdots & y_{1N_y} \\ x_{21} & x_{22} & \cdots\cdots & x_{2N_x} & y_{21} & \cdots & y_{2N_y} \\ \vdots & \ddots & & & & & \\ x_{N_s1} & x_{N_s2} & \cdots\cdots & x_{N_sN_x} & y_{N_s1} & \cdots & y_{N_sN_y} \end{array}\right) \tag{4.5}$$

after which *example* is finally formed by flattening $example_{mat}$ to make the data 1-dimensional, allowing for seamless input to most prediction algorithms, e.g., ANN or Random Forest. Alternately, the 2-dimensional structure can be kept for feeding to a different model such as a CNN. The number of rows in the dataset is

$$|dataset|_r = N_{ts} * D_{ts} \tag{4.6}$$

and the number of columns in the dataset is

$$|dataset|_c = (N_x + N_y)N_s \tag{4.7}$$

where N_x and N_y are the number of columns in the labeled dataset corresponding to the features and labels respectively. More precisely we have $N_x = \sum_{i=1}^{|F|} D_{h_i} * |f_i|$ and $N_y = \sum_{i=1}^{|L|} D_{h_i} * |l_i| \quad f_i \in F, l_i \in L$

Besides potentially higher accuracy, a second natural benefit of our MTL implementation is generating additional outputs at the same time. Moreover, here we generate the demand at multiple stations on multiple days at the same time. We see this as beneficial in this application because the alternate approach would be where many ML models needs to be tuned, trained, and maintained. We do not anticipate such an approach to scale well, particularly with the expected rapid growth in EV charging. For instance, the industry figures show the EU had about 220,000 public stations in 2020. However, the industry told the EU to expect 1 million by 2024, and 3 million by 2029 [32].

Finally, this means in our approach the demand in different areas is obtained with just one inference operation. Also, simply summing the outputs will obtain the demand at any combinations of stations, as they are obtained directly from the right-hand side of Equation 4.5. Thus overall we provide a framework for stations focusing attention on other station's behavior, possibly improving the prediction at stations or networks, and generating output for multiple tasks.

Chapter 5

Experimental Set-up

5.1 Testing Strategy

To test the effectiveness of the approach the dataset is divided into a training, validation, and test set. We choose a testing strategy which is suitable for time-series applications known as last block validation [10]. The strategy insures validation on a mix of seasons of the year, as depicted in Figure 5.1, which shows the time frame of each set and how many charging events are in each set. The testing strategy enables us to find a robust model where separate seasons do not need their own prediction models.

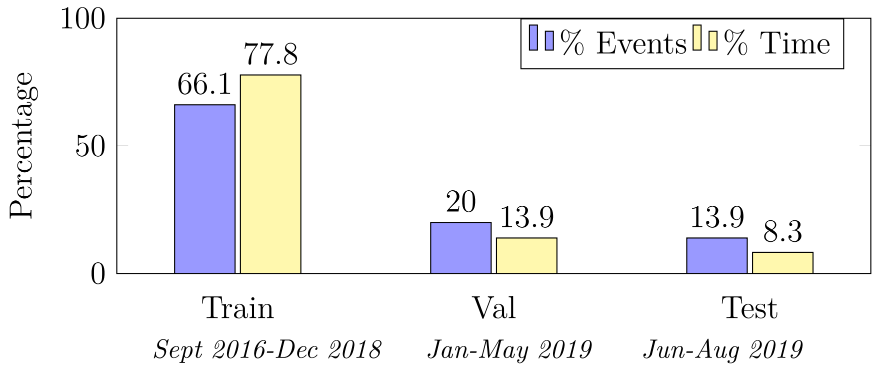


Figure 5.1: Testing data distribution

5.2 Approaches

Here we test various learning-based models to demonstrate our method. In the last ten years, the two mainly used ML techniques in the area of predictions for the smart-grid were SVM and ANN, and a few other (mostly supervised) techniques were used less commonly [23]. We test the ANN as discussed earlier, along with other models including random forests [33], k-nearest neighbors (kNN) [34], linear regression (LR), and support vector regression (SVM or SVR) [35].

We also use ARIMA, considered both a statistical and learning-based model, along with baselines. The only feature fed to the ARIMA model is past energy demand, and as with all models it is set up to predict future energy demand on the same test set. Thus we remove the other features, otherwise the same procedure is used to generate training and test data.

We choose the following baselines because they can be determine if any useful learning takes place. The baselines are as follows:

1. Baseline 1 - The data from the previous week is the forecast for next week.
2. Baseline 2 - The average of the previous week's data is forecast as the demand for all of the next week.

The baselines are forms of persistence forecasting. Borrowed from meteorology, persistence forecasting assumes futures behavior will be the same as current behavior and is one dependable ground work for forecasting [36].

5.3 Loss Definition

We define the loss function for training a neural net, or other gradient decent based methods, as the mean squared error

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{5.1}$$

with equal weight given to each output.

Many loss functions can be defined to report results for regression tasks. The loss functions we include here are the mean absolute error (MAE), root mean squared error (RMSE) and symmetric mean absolute percentage error (SMAPE). MAE and RMSE are universal and can be applied to any regression task, for instance energy consumption, charging duration, etc. We include SMAPE because it is a common percentage metric and as such considers the error as a percentage of the target. These three metrics are defined as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i| \tag{5.2}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2}{n}} \tag{5.3}$$

$$SMAPE = \frac{100\%}{n} * \sum_{i=1}^{n}\frac{|\hat{Y}_i - Y_i|}{|\hat{Y}_i| + |Y_i|} \tag{5.4}$$

5.4 Network vs Station Level Modelling

In this study we take a view of charging from the network and station granularity levels. As such, network demand can be modelled in multiple ways. Table 5.1 describes three ways to model network demand we studied (see first three rows), with each row also indicating if the strategy uses MTL. Strategy A is the most simple and does not attempt to use MTL. In contrast, strategies B and C do leverage MTL and organize the tasks to form a dataset exactly as in Algorithm 1.

The modelling of a station's demand is similar. Stations can be either modelled individually (method D), or modelled with other stations or tasks (method E). Next, we test different models paired with the ways to model demand. The label for the experimentation is the aggregate daily energy value and remains constant.

Method	Network/ Station	MTL	Description
A	Network	No	Sum demand at all stations and feed to a model dedicated to this task.
B	Network	Yes	Feed each station's demand to a model individually, in addition to the network demand, and take the dedicated network demand label.
C	Network	Yes	Same as B, but instead of taking the dedicated network label, assume the network demand is the sum of individual station's demand.
D	Station	No	Each station gets its own model.
E	Station	Yes	Each station is part of a shared model, as in method B and C.

Table 5.1: Ways to model demand

Chapter 6

Results and Discussion

6.1 Application

The following application was created to make predictions for the relevant stakeholder. An application uses the time of day and takes input from the user on the prediction delay. The other inputs to the application are a trained model, along with the normalization parameters that were used to train the model. The application needs only to tap a database with the latest weather data, then perform inference on a trained model to forecast the energy demand for the next seven days. Optionally, the program can also take as input some testing data. This testing data serves as a tool to validate the model. Here, forecast can be plotted on top of the actual charging data and baselines so that visual validation can take place.

6.2 Performance

6.2.1 Network Demand

Here we show the results of predicting the charging demand in the network, meaning aggregate demand in the whole geographical area.

Model	MAE	RMSE	SMAPE
ANN	**125.481**	**158.192**	**5.964**
ARIMA	145.450	174.362	6.890
KNN	370.265	408.069	20.551
Linear Regression	127.431	160.031	6.040
Random Forest	150.275	187.819	7.091
SVM	518.202	491.897	28.904
Baseline 1	167.595	209.761	7.929
Baseline 2	135.574	167.847	6.385

Table 6.1: Network result using method A

(i) *STL* - Forecasting network demand with single-task learning (strategy A) is shown in Table 6.1. With a single task, any of the machine learning models discussed can be applied without modification. The ANN performs the best, with about seven percent better MAE and SMAPE than the next best method which is baseline 2.

(ii) *MTL* - Here we show the performance on the task of predicting network demand with MTL. Recall from Table 5.1 that this method is further divided into (B) and (C). Table 6.2 shows that the ANN performs the best with method C, which assumes the network demand is the sum of station demand. Moreover, the use of MTL for learning network demand led to improvements in MAE and SMAPE of 1.2% and RMSE of 2.1%.

	With weather			Without weather		
Model	MAE	RMSE	SMAPE	MAE	RMSE	SMAPE
ANN (B)	126.731	157.908	6.003	135.068	168.035	6.300
ANN (C)	**124.670**	**155.518**	**5.925**	134.016	165.742	6.258

Table 6.2: Predicting network energy for next week using method B and C

6.2.2 Individual Station Demand

Because there are multiple stations, here we assume the goal is the reduce the average error across all of them. There is no need to weight busier stations because SMAPE is a percentage error so it already provides relative error values.

Model	Avg MAE	Avg RMSE	Avg SMAPE
ARIMA	31.979	**38.486**	25.247
ANN	**31.315**	**39.649**	**21.308**
ANN-*no weather*	31.888	40.308	22.116
Random Forest	36.316	45.267	28.942
KNN	57.117	46.579	33.350
Linear Regression	33.105	40.977	35.977
SVM	49.473	59.722	33.643
Baseline 1	42.593	53.661	33.629
Baseline 2	32.691	40.773	25.466

Table 6.3: Individual station result with method D

(i) *STL* - With single-task learning we arrive at Table 6.3. The top performing models are the ANN and ARIMA. The ANN performs similarly to the ARIMA model in terms of MAE and RMSE. However, when considering SMAPE, the ANN performs much better (12% relative improvement or 3% absolute improvement).

(ii) *MTL* - Multi-task learning can predict the demand at every station and the network at the same time. We show the results for stations with this approach in Table 6.4. We note that MTL with method E generates results about the same as method D. However method E is slightly worse than D across the three metrics. This shows that, overall across all stations, individual station demand at each station was not useful as an auxiliary task.

	With weather			**Without weather**		
Model	MAE	RMSE	SMAPE	MAE	RMSE	SMAPE
ANN (E)	**32.591**	**41.088**	**22.351**	33.566	42.239	22.576

Table 6.4: Predicting individual station demand for next week with method E

Since the ANN with method D performed the best at predicting individual station demand we show the breakdown of each station's performance shown in Table 6.5. This table shows the variation in performance among the stations, where the error of most stations is around 20% SMAPE. However, the table shows that there was variation depending on station. For example, the error at the Friarton station increased to about 44%.

In more detail, Figure 6.1 shows an example of predicted and actual values in the test set. The values are plotted next to Baseline 2, which is a moving historical average. For about half of the time period, the baseline performs the same or better than the model. Nonetheless, considering the magnitude of the error, as we have throughout the results, the overall accuracy is best with the ANN model.

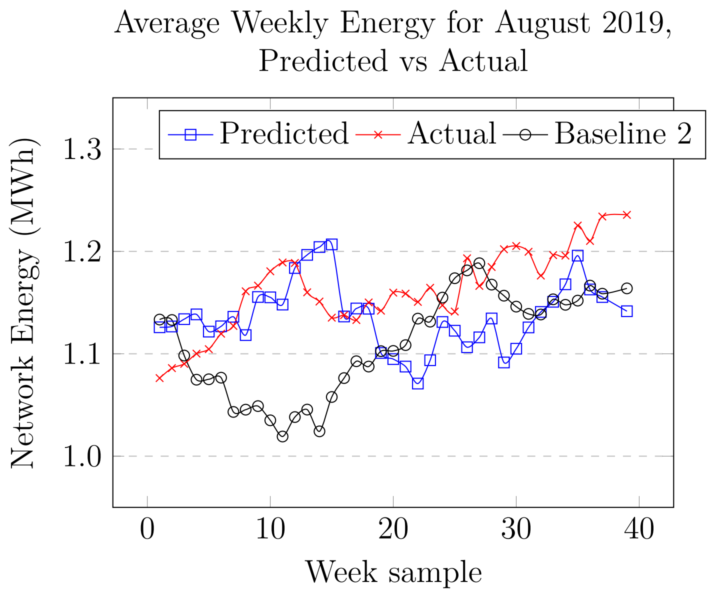


Figure 6.1: Model performance showing predicted vs actual and baseline 2 values for August 2019 in the test set.

6.3 Discussion

We showed that, in the application area of public car charging, machine learning can train on historical data, combined with various features, to outperform ARIMA which is a classic time-series forecasting model. We showed that machine learning can perform accurate

Station	MAE (kWh)	RMSE (kWh)	SMAPE (%)
Atholl	36.078	45.522	18.333
Broxden	54.005	67.199	12.130
Canal	12.530	15.846	30.644
Crown Inn	24.619	29.991	11.866
Friarton	13.992	18.722	43.549
King Street	29.757	36.680	21.150
Kinross	52.884	66.158	11.866
Leslie Street	21.249	28.403	25.510
Mill	26.211	33.095	19.934
Rie-Achan	42.397	56.845	20.416
South Street	32.915	40.112	20.478
Average	*31.315*	*39.649*	*21.308*

Table 6.5: Breakdown of station results test error using ANN with method D

forecasts, with an ANN showing the highest potential in each application area when compared to the other models. Notably, when we choose features to form a dataset which an ANN then trains on, we outperform the ARIMA model and simple baselines which inherently do not exploit features.

The lowest MAE from predicting network energy for seven days was 124.670kWh, while in the test set the average daily energy consumed in the network was 1.086 GWh, so the MAE while predicting the next seven days of demand is less than twelve percent of the average daily network demand. Not only can low percentage error be achieved from our strategy, but the accuracy values for the network prediction were only half of that in [4] which only studied the immediate short term demand and relied on traffic information, which may not be as readily available as weather information. It is reasonable to state, however, that with more perfect and abundant information that more accurate results can be found. Specifically, user driving behavior and traffic flow could eliminate more of the randomness of the prediction problem. For example it was shown in [4] that there is a strong positive correlation between traffic flow and charging load at a fact charging station on the side of a highway.

The lowest average error for predicting individual station demand was the ANN with method D with a MAE of 31.315 kWh. This method was only about four percent less error than Baseline 2. However, the results here are across all stations in the network and the standard deviation is $\sigma = 13.3$ which means the ANN method benefits some stations more than others.

Additionally, we consistently saw that weather as an input improves predictions. Because none of these baselines can encode features, in a geographical region with less stable weather, the performance gap between these statistical models and machine learning could widen. We took a first step towards using weather by incorporating a few days of past and future weather data. In another work, one could use larger values of D_{wf}, but data augmentation would have to be used to consider that weather forecasts can be very inaccurate more than a few days ahead. For future study, a successful implementation might require an accurate noise model to perturb the weather data, since forecasts drop in accuracy significantly with prediction horizons longer than a few days.

Another way to expand the study is to look at extended prediction horizons. In Figure 6.2 we adapt the model to predict for longer prediction horizons. The plot shows the performance on each day with the extended horizon, with the baseline being the average of past demand

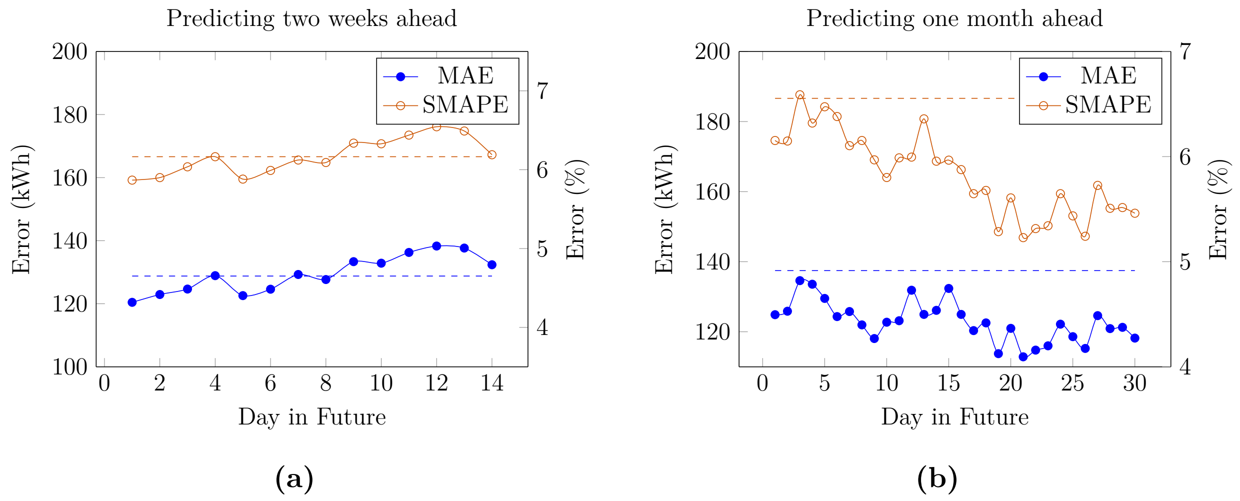


Figure 6.2: Model performance of network prediction task with expanded prediction horizon. Our model deployed to predict for two weeks in (a) and a larger model to predict the next month in (b)

shown as a dashed line. In plot (a), predicting two weeks ahead, the model performs better or equivalent to baseline up to eight days in the future. In plot (b), the model is grown slightly to accommodate predicting one month in the future. In general, the model shows promising results on prediction horizons longer than the previously studied one week.

Furthermore, the results show that a one model per station strategy can be used, returning a reliable forecast, like has been demonstrated before for short term predictions [10]. We saw that a dedicated model for each station, compared to a shared model, works better on average, as shown in Table 6.4 and Table 6.5. To summarize, the multi-task learning method was not beneficial for the station prediction task, but it was for the network prediction task.

Interestingly, however, some station's forecasting accuracy can be improved with MTL strategy E. Three out of eleven stations showed better results with method E compared to D. This improvement was from two to five percent, as shown in Figure 6.3. So, this MTL technique can be employed effectively for predicting the individual station demand, with the benefit being localized. More over, feeding data to an ANN model could serve as framework for determining if the behavior at pairs of stations is correlated.

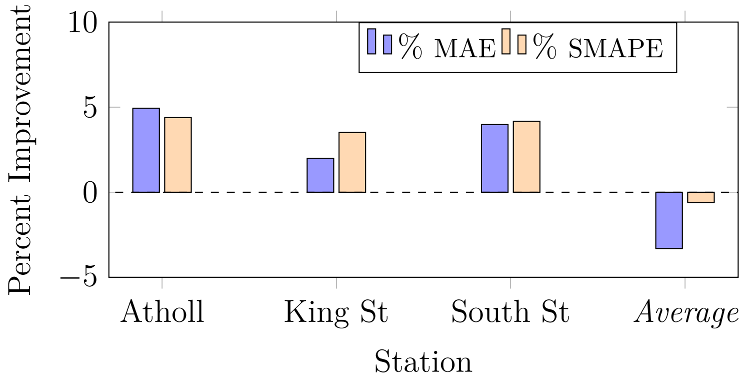


Figure 6.3: Stations with improvement using MTL (percent change from method D to method E)

Lastly, a discussion on the efficiency of the approach is warranted. With a traditional forecasting model such as ARIMA, not only was the accuracy less but also the efficiency. The ANN is trained one time covering the whole validation/test set. However, the model that performed slightly worse than ANN, ARIMA, required model parameters to be updated throughout the duration of the test set.

Chapter 7

Conclusion and Future Work

In this study we demonstrated the effectiveness of machine learning to make predictions of future public EV station demand for up to a week in advance. We formulated the prediction of charging as a regression problem, and quantified demand as the daily energy consumption. In our solution, we chose a set of parameters to format a labeled dataset, introduced flexible prediction horizons into the dataset, carried out feature engineering, and implemented known machine learning models for prediction.

Comparing the accuracy of our model to a few baselines showed promising results from the various performance metrics. Our machine learning model performed better than a few statistical methods by 2-7%. Moreover, we took a step towards using weather data, to improve the prediction by a few percent. Lastly, we transformed the dataset to investigate a form of multi-task learning that improved prediction performance by about 1% for network charging demand.

Future research directions may include incorporating custom ML models into the methodology. We saw that performance varies greatly depending on charging station, so a custom model could work to make results less location-dependent. Also, one may wish to reduce the parameter or feature search space when labeling data. Finally, additional datasets, characterized by different weather or behavioral data, could be explored.

Bibliography

[1] "U.K. to ban new gas cars by 2030 as leader touts green industrial revolution," *The Associated Press*, November 2020.

[2] D. Jones, "General motors aims to be carbon neutral by 2040," *NPR*, January 2021.

[3] M. G. Flammini, G. Prettico, A. Julea, G. Fulli, A. Mazza, and G. Chicco, "Statistical characterisation of the real transaction data gathered from electric vehicle charging stations," *Electric Power Systems Research*, vol. 166, pp. 136–150, 2019.

[4] X. Zhang, K. W. Chan, H. Li, H. Wang, J. Qiu, and G. Wang, "Deep-learning-based probabilistic forecasting of electric vehicle charging load with a novel queuing model," *IEEE Transactions on Cybernetics*, pp. 1–14, 2020.

[5] Open-Data Team, "Electric vehicle charging station usage." *Open Data Perth and Kinross*. Accessed July 10, 2020. [Online], 2017.

[6] E. Xydas, C. Marmaras, L. Cipcigan, N. Jenkins, S. Carroll, and M. Barker, "A data-driven approach for characterising the charging demand of electric vehicles: A UK case study," *Applied Energy*, vol. 162, pp. 763–771, 01 2016.

[7] G. Gross and F. Galiana, "Short-term load forecasting," *Proceedings of the IEEE*, vol. 75, no. 12, pp. 1558–1573, 1987.

[8] H. Alfares and N. Mohammad, "Electric load forecasting: Literature survey and classification of methods," *International Journal of Systems Science - IJSySc*, vol. 33, pp. 23–34, 01 2002.

[9] M. Luo, B. Du, K. Klemmer, H. Zhu, H. Ferhatosmanoglu, and H. Wen, "D3P: Data-driven demand prediction for fast expanding electric vehicle sharing systems," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 4, Mar. 2020.

[10] M. Majidpour, C. Qiu, P. Chu, R. Gadh, and H. R. Pota, "Fast prediction for sparse time series: Demand forecast of ev charging stations for cell phone applications," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 1, pp. 242–250, 2015.

[11] S. Shahriar, A. R. Al-Ali, A. H. Osman, S. Dhou, and M. Nijim, "Machine learning approaches for ev charging behavior: A review," *IEEE Access*, vol. 8, pp. 168980–168993, 2020.

[12] Y.-W. Chung, B. Khaki, T. Li, C. Chu, and R. Gadh, "Ensemble machine learning-based algorithm for electric vehicle user behavior prediction," *Applied Energy*, vol. 254, p. 113732, 2019.

[13] Tianyang Zhang, Xiangyu Wang, Chi-Cheng Chu, and R. Gadh, "User demand prediction and cloud-based smart mobile interface for electric vehicle charging," in *2016 IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, pp. 348–352, 2016.

[14] Y. W. Chung, M. Mathew, C. Rodgers, B. Wang, B. Khaki, C. Chu, and R. Gadh, "The framework of invariant electric vehicle charging network for anomaly detection," in *2020 IEEE Transportation Electrification Conference Expo (ITEC)*, pp. 631–636, 2020.

[15] R. Yin, D. Black, and B. Wang, "Characteristics of electric vehicle charging sessions and its benefits in managing peak demands of a commercial parking garage," in *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pp. 1–7, 2020.

[16] J. Wang, K. Wu, F. Wang, Z. Li, Q. Niu, and Z. Liu, "Electric vehicle charging station load forecasting and impact of the load curve," *Applied Mechanics and Materials*, vol. 229-231, pp. 853–858, 11 2012.

[17] Y. Chung, B. Khaki, C. Chu, and R. Gadh, "Electric vehicle user behavior prediction using hybrid kernel density estimator," in *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pp. 1–6, 2018.

[18] M. Majidpour, C. Qiu, P. Chu, R. Gadh, and H. R. Pota, "Modified pattern sequence-based forecasting for electric vehicle charging stations," in *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pp. 710–715, 2014.

[19] X. Huang, D. Wu, and B. Boulet, "Ensemble learning for charging load forecasting of electric vehicle charging stations," in *2020 IEEE Electric Power and Energy Conference (EPEC)*, p. 1–5, IEEE, Nov 2020.

[20] M. Straka, P. De Falco, G. Ferruzzi, D. Proto, G. Van Der Poel, S. Khormali, and L. Buzna, "Predicting popularity of electric vehicle charging infrastructure in urban context," *IEEE Access*, vol. 8, pp. 11315–11327, 2020.

[21] "World weather online," *WorldWeatherOnline.com*, 2021.

[22] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[23] T. S. Bomfim, "Evolution of machine learning in smart grids," in *2020 IEEE 8th International Conference on Smart Energy Grid Engineering (SEGE)*, pp. 82–87, 2020.

[24] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[25] T. G. Smith *et al.*, "pmdarima: Arima estimators for Python," 2017–. [Online; accessed June 2021].

[26] Y. Wang and Y. Guo, "Forecasting method of stock market volatility in time series data based on mixed model of arima and xgboost," *China Communications*, vol. 17, no. 3, pp. 205–221, 2020.

[27] J. W. Taylor and P. E. McSharry, "Short-term load forecasting methods: An evaluation based on european data," *IEEE Transactions on Power Systems*, vol. 22, no. 4, pp. 2213–2219, 2007.

[28] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, 07 1997.

[29] B. Ramsundar, S. Kearnes, P. Riley, D. Webster, D. Konerding, and V. Pande, "Massively multitask networks for drug discovery," *arXiv:1502.02072*, 02 2015.

[30] S. Ruder, "An overview of multi-task learning in deep neural networks," *CoRR*, vol. abs/1706.05098, 2017.

[31] T. J. Sejnowski and C. R. Rosenberg, "Nettalk: A parallel network that learns to read aloud," Tech. Rep. JHU/EECS-86/1, John Hopkins University Department of Electrical Engineering and Computer Science, 1986.

[32] V. Eckert, “EU told 1 million public ev charging stations needed by 2024,” *Reuters*, February 2021.

[33] L. B. Statistics and L. Breiman, “Random forests,” in *Machine Learning*, pp. 5–32, 2001.

[34] N. S. Altman, “An introduction to kernel and nearest-neighbor nonparametric regression,” *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[35] H. Drucker, C. Burges, L. Kaufman, A. Smola, and V. Vapnik, “Support vector regression machines,” *Adv Neural Inform Process Syst*, vol. 28, pp. 779–784, 01 1997.

[36] R. Lamb, “How far in advance should i check the weather forecast?.” HowStuffWorks, Mar. 2, 2009. [Online].