Master Thesis

A Testament to the Patience of Supervisors

Bjørn Ingeberg Fesche

Thesis submitted for the degree of
Master in Informatikk: Programmering og nettverk
60 credits

Institutt for informatikk
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2018

Master Thesis

A Testament to the Patience of Supervisors

Bjørn Ingeberg Fesche

Master Thesis

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Chapter 1

Signature Discovery

1.1 Introduction

One of the key challenges facing power companies and motivating the adaption of advanced metering infrastructure (AMI), is the rapidly increasing market share of electric vehicles (EV). Tackling the large power surges associated with EV charging is going to require deep insight into the nature of these charging cycles.

A natural question to ask is, what does a typical charging cycle look like? I attempt here to answer this question based on data from the research and technology organization Pecan Street.

1.2 Data Preparation

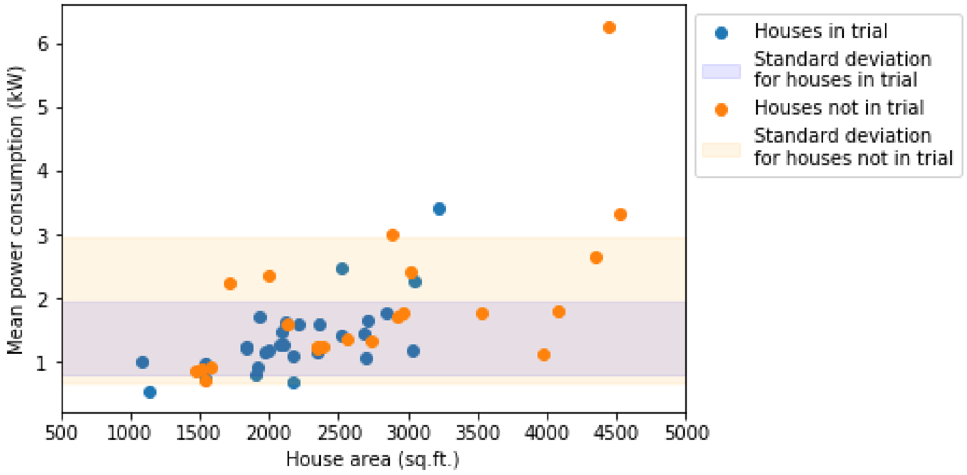
The data I have used comes from the Pecan Street Dataport database (www.dataport.cloud). Pecan Street is a nonprofit research institute focused on the utility sectors and has data from over 1400 households freely available for academic use.

1.2.1 Raw Data

For this experiment I collected power data from the EV-charger outlet, as well as the household net power drain on the grid (power use minus power generation) at 1-minute resolution. 55 houses had records of this data from January 1st to December 31st, 2017. Out of these houses, 12 houses had incomplete records, and were initially discarded for convenience, and later used for testing.

Out of these 55 houses, four were town homes, sharing at least one wall with a neighbor. The rest were free standing single-family homes. 30 of the houses were enrolled in a CCET electricity pricing and customer behavior trial in 2013 and 2014. Out of these, three were in the text message group and received text messages on days with a high peak load asking them to reduce their power consumption. Four were in the UT text message group and received text messages asking them to restrict usage of specific appliances on peak days. Eighteen households had access to information about their hypothetical electricity cost with an hourly rate. Sixteen of these received financial incentives for reducing their power consumption according to this rate. The rest were in the control group.

Figure 1.1: House area in square feet versus the household mean power use (gross). The shaded areas indicate one standard deviation of the power use.



It is unlikely that a study concluded three years before had much to say on the behavior of the consumers. The differences observed in Figure 1.1 can probably be attributed to the larger share of big houses among those which did not participate in the trial.

Before using the data, I downsampled it to a 15-minute resolution, taking the mean over each 15-minute interval. This in order to match the likely resolution of Norwegian smart meters and reduce the computational effort necessary.

Figure 1.2: When grouped by length, the charging signatures across all houses display groupings in typical charging patterns, mostly defined by their maximum power drain. The y-axes here are the effect measured at the EV charger outtake in kW, while the x-axes are timesteps of 15 minutes.

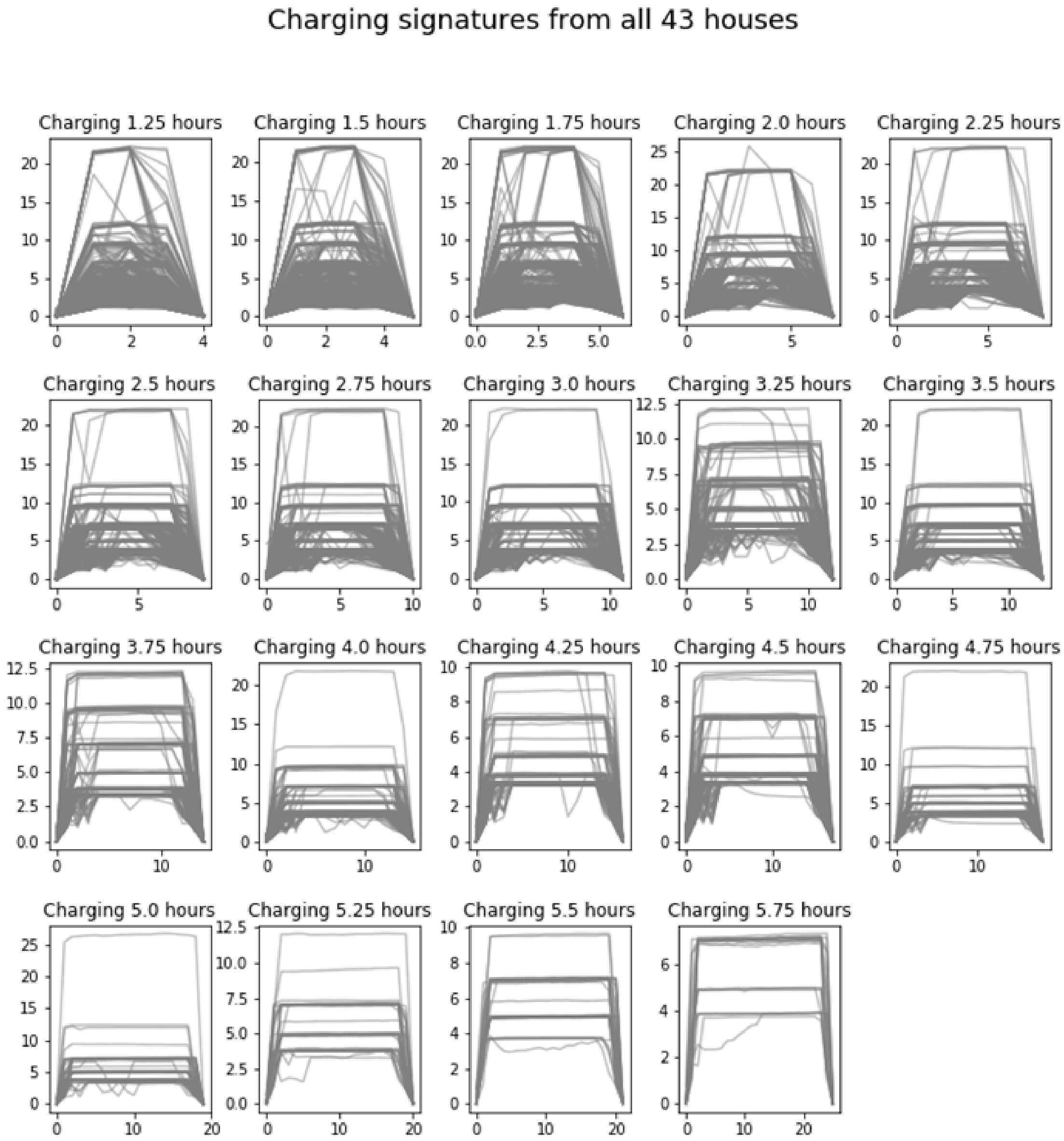
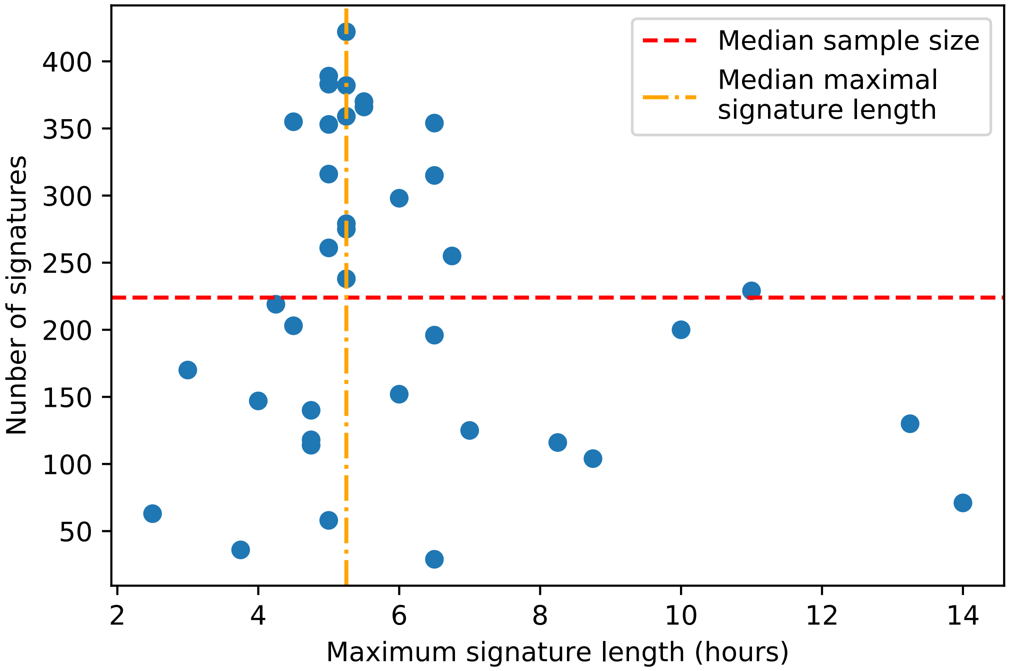


Figure 1.3: The number of charging signatures extracted from each house plotted against the maximum signature length in hours. The red and yellow lines indicate respective medians.



1.2.2 Extracting Charging Signatures

To get data with which I could train a model, I created a library of EV-charging signatures. This was done by collecting all instances of at least three consecutive measurements from the EV-charger of a house with a value of at least 1 kW. To preserve the shape of the signal, the signatures were also padded at the beginning with the last preceding measurement, and at the end with the first succeeding measurement. Figure 1.3 compares the data drawn from each house by number of signatures collected versus the longest signatures collected. Figure 1.2 shows all collected charging signatures plotted by duration.

1.3 Typical signatures

Figure 1.2 conveys a decent idea about the shape of all the individual charging signatures in the data. Interestingly, the signals do appear to come in regular forms. For many applications, such as signal discovery by matched filters, using all the discovered charging signatures is infeasible. A typical, representative pattern, or an aggregate signature for each charging duration are therefore needed.

The raw signatures are first sorted by their lengths to be more easily comparable. For the purposes of finding typical charging signatures, every signature of a length of which there are less than 100 signatures is discarded to reduce the variance of the results. Signatures of the three shortest charging durations are also discarded, as they are very prone to noise.

A way to discern discrete categories into which the signatures fall is required. Figure 1.4 shows the average maximum power drain per charging event for each house plotted as a horizontal line. From this perspective there appears to be seven clear groupings of signatures. Figure 1.5 shows the maximum power drain per individual signature. This figure is less easily interpretable but appears to reveal at least four or five distinct groups.

Figure 1.4: The mean maximal signature value per house plotted as horizontal lines. Darker lines indicate more houses with the same value.

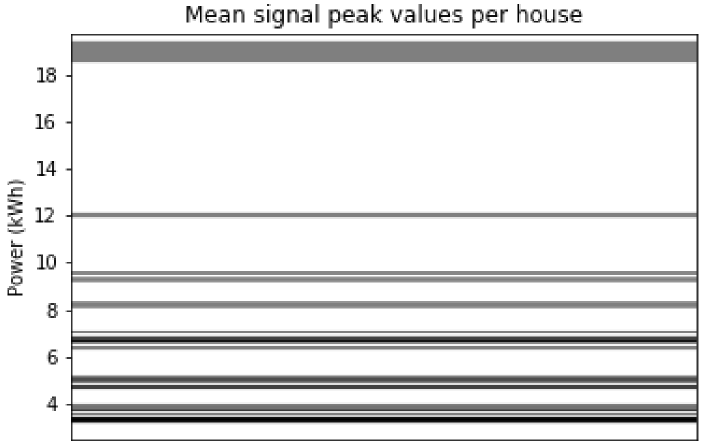
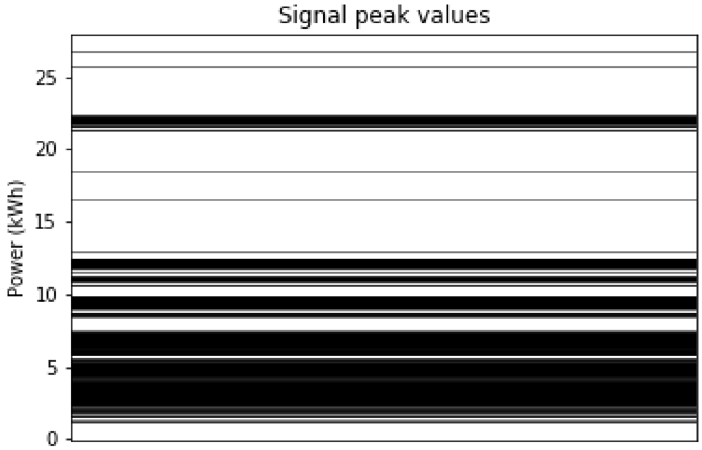


Figure 1.5: The same plot, but with all signatures plotted individually instead of taking the mean for each house.



1.3.1 Clustering

Clustering methods are a class of generally unsupervised machine learning techniques, which attempt to discover natural formations, or clusters, in the data. Clustering methods are thus well suited for the problem of discovering typical charging signatures. Among the most common clustering methods is the k-means algorithm. K-means fits k centroids to the data by first creating centroids randomly, then iteratively assigning each data point to the centroid closest to it (by Euclidean distance) and moving each centroid towards the center of mass of the points currently assigned to it. More formally, at every time step, each data point x_p is assigned a cluster C_i belonging to a centroid c_i by:

$$C_i = \{x_p : \|x_p - c_i^2\| \leq \|x_p - c_j^2\| \ \forall j, \ 1 \leq j \leq k\}$$

Then every centroid is moved by:

$$c_i^{(t+1)} \ = \ \tfrac{1}{|C_i|} \textstyle\sum_{x_j \in C_i} x_j$$

Finally, when the centroids' movement converges, or a maximum number of iterations is reached, the algorithm stops.

1.4 Choosing k

To determine what is a good clustering, a metric is required. One such metric is the silhouette score. The silhouette score measures the relationship for each data point between intra-cluster distance and the distance to the data points in the closest neighboring cluster. Specifically, the silhouette score $s\left[i\right]$ for a data point i is defined as:

$$s\left[i\right] = \frac{b[i]-a[i]}{max(b[i],a[i])}$$

Where $b\left[i\right]$ is the average distance from data point i to every data point in the closest cluster which i does not belong to, and $a\left[i\right]$ is the average distance from i to every data point in the same cluster as i. This means that a data point with a score close to 1 is perfectly clustered, while one with a negative score is incorrectly clustered.

There are two common ways to use this metric: By examining the average silhouette score of all clustered points, and by examining at the silhouette scores of the individual data points within each cluster. The latter method is useful when choosing the number of clusters because it reveals clusters which are particularly well or poorly suited, for example by comparing the maximum silhouette score of a cluster to the global average. In the present case, both analyses are useful.

Another factor which influences our choice of k is that since I are looking specifically for typical nuances in the signatures, it makes sense to choose more clusters rather than fewer.

Figure 1.6 shows the average silhouette score of all data points for each signature length, with the mean per choice of k in black and the mean of these means shown as a horizontal red line. Since I are interested in discovering more, rather than fewer typical signatures, a good choice of k from this graph could be at the “elbow”. This is where increasing the number of clusters no longer corresponds with a drop in silhouette score, implying that there are no more sensible partitions of the data after this.

Figure 1.7 shows the proportion of “poor” clusters – that is, clusters where all the points fall beneath the global average silhouette score – per choice of k for each signature length. Figure 1.8 shows the distribution of “perfect” clusterings – that is, clusterings in which no cluster falls completely below the global average – per choice of k. Together, these two graphs show that a k around seven yield the lowest number of poor clusters. This coincides well with the intuition gained from Figure 1.4 and Figure 1.6.

Figure 1.6: Each grey line indicates the average silhouette score of a single signature length. The black line shows the mean of these scores. The value of k I are interested in is the last value before the scores hit "bottom".

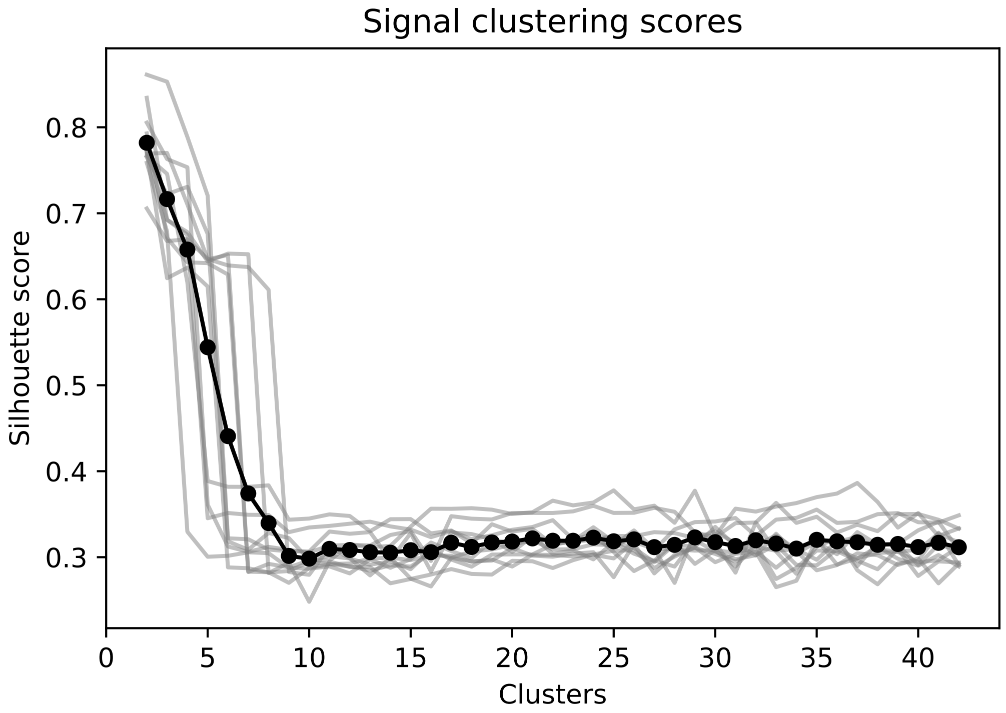


Figure 1.7: Each grey line represents the proportion of clusters falling completely beneath the average silhouette score for all signatures. The x-axis is the number of clusters in the clustering.

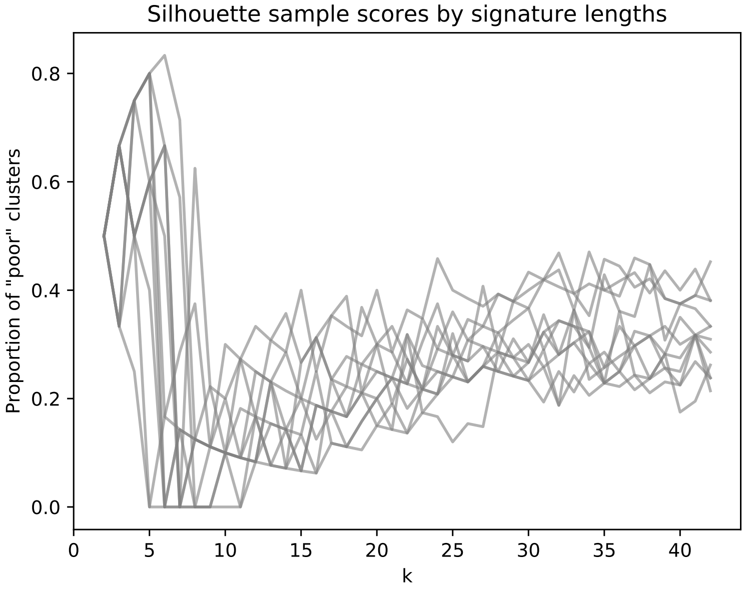
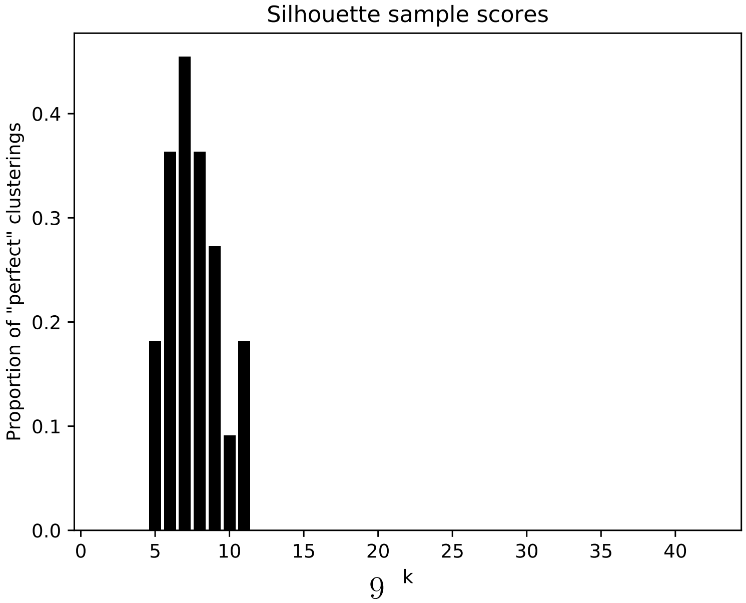


Figure 1.8: The proportion of signature lengths which yield clusterings in which all clusters contain points above the global silhouette score average. The x-axis is the number of clusters in the clustering.

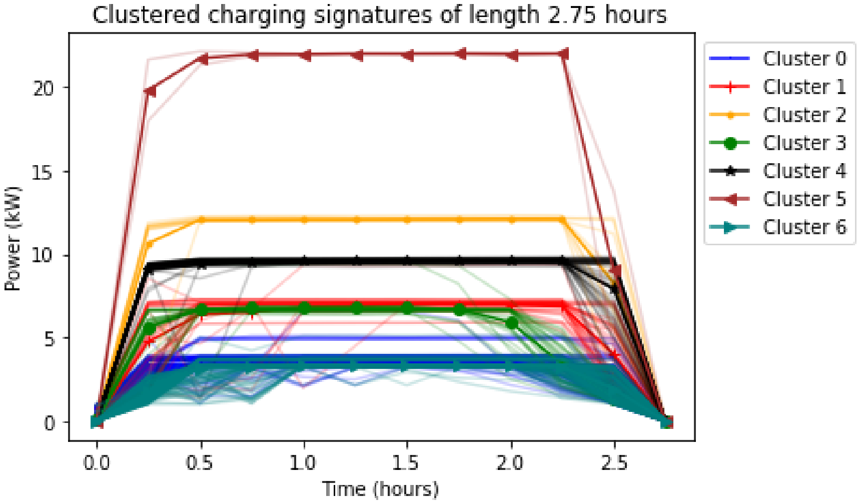


A pragmatic approach to choosing k for our purposes (crucially, choosing a single k for multiple signature lengths) can thus consist of two measures: The highest k with silhouette score above some deviation from the mean for a sufficiently large range of k, and the k with the largest number of "perfect" clusterings. In this case, the two measures coincide and yield $k = 7$.

1.5 Finding Templates

Everything is now in place to extract typical charging signatures from the data. Figure 9 plots signatures of a given length by cluster with $k = 7$, showing the cluster means as heavy lines with markers.

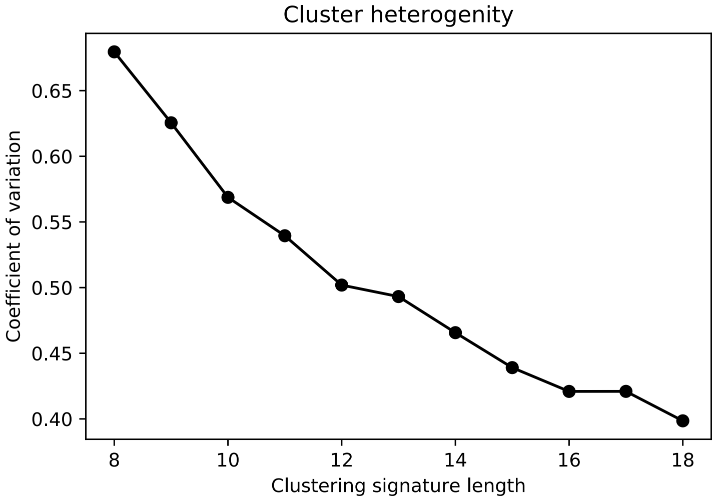
Figure 1.9: The resultant clusters with $k = 7$ for signatures of length 2 hours 45 minutes. The heavy lines indicate cluster means, while the thin lines indicate individual signatures color-coded by cluster.



The clustering seems mostly to be based on average power drain, but some clusters, like 3 and 1, display subtler differences still apparent in the figure.The clustering thus seems reasonable at least for this sample signature length. Figure 1.10 plots the average coefficient of variation (standard deviation normalized by mean) for each signature length, showing that longer signatures lead to more coherent clusters.

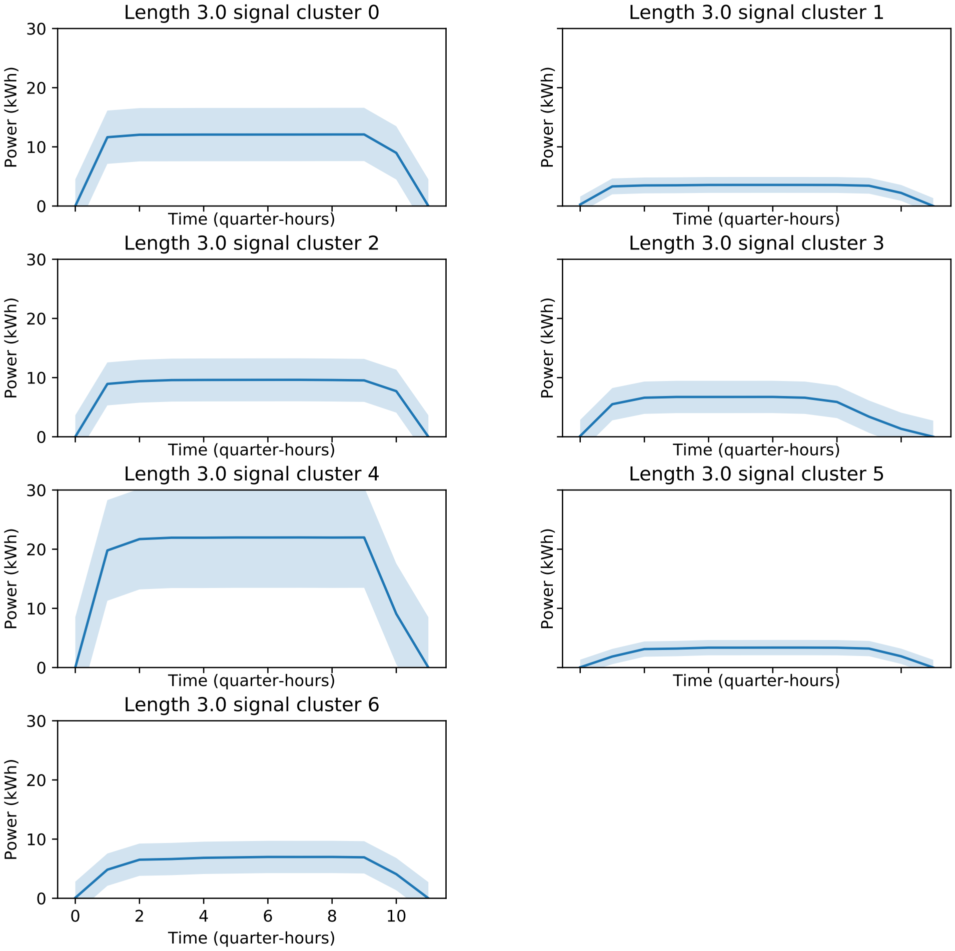
Since these clusterings ideally capture the distinct structures in the data, their means should provide good insight into the shape of typical charging

Figure 1.10: The average coefficient of variation for each signature length; that is, the average standard deviation within each cluster normalized by mean signature value.



signatures. Choosing k remains the primary bottleneck, and while this example led to an unequivocal choice, automating the selection for less obvious cases may provide a challenge.

Figure 1.11: Sample resulting signature templates. Cluster 3 and 4 display an exponential decline, as seen clearly in the signatures of some households at higher resolution. The rest appear to be clustered by mean power drain.



Chapter 2

Signal Discovery: Neural networks

2.1 Introduction

Given the impressive results achieved in data analysis and prediction tasks by neural networks in recent years, attempting to adopt this discovery for signal discovery seemed an obvious path to take. Some research in this area has already been done, but it remains to be fitted to this particular problem.

I have used three different techniques, all attempting, like with the matched filters, to discover charging cycles within an aggregate power use signal from the entire year of 2017. As with the matched filters, the models are trained on 43 houses and tested on the remaining, incomplete 12 houses. All models were trained on windows of data, and so do not take into account anything but the signal itself. While considering variables such as time of day may yield higher test scores, this approach would be less general, and thus more prone to ignoring non-typical charging cycles.

2.2 Training Data

Unlike the matched filters, which were assembled by using an unsupervised clustering method, the neural nets have to be supervised in training. This means providing examples of labeled input data from which the nets can learn. The more training data, the better. Extracting all windows consisting of charging signals of length 1.5 hours or longer from the EV-charger signal as well as the corresponding windows in the aggregate signal provides a few thousand training samples. This may not be enough for a model to properly generalize, or may trick a model into recognizing certain common related features which are not really related to EV-charging. For example, if enough car owners come home after work, and immediately plug in their EV and start preparing dinner, the model may learn to assign a higher score to peaks near stove-like signals. This may sound like an assumption I could be happy to let the model make, but really I will want it to find EV signals even in houses where the occupants never cook.

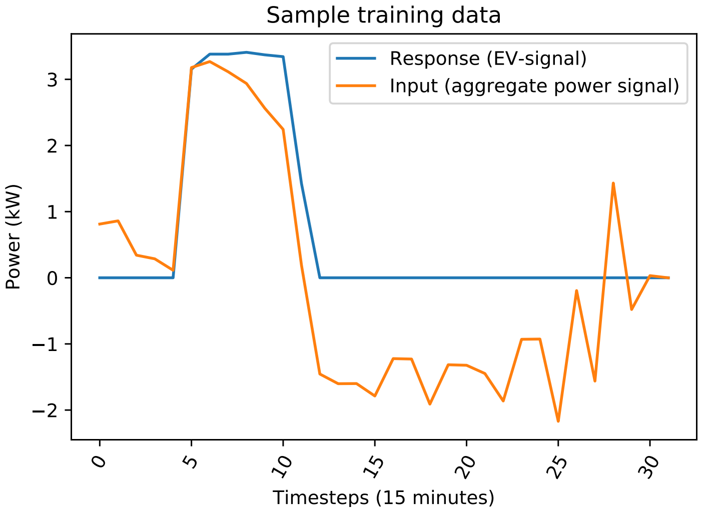
Luckily, I can make data. Since I have the actual charging signatures from the charger power signal,I can use these to create more data by adding charging signatures to random windows not overlapping with any charging event from the aggregate signals. I also want to have samples of data which does not contain a charging signal, which are simply the random aggregate windows with no signature added. Using this method, I created a set of 400 000 samples of training data, half of which had a response of zero.

2.3 Denoising Autoencoder

A denoiser attempts to remove noise from data, extracting an underlying signal. Commonly used for audio or image enhancement, I will attempt to use denoising to recover the EV-charging signal from an aggregate. The inputs to this model will consist of aggregate signal windows, of which about half contain an embedded car-charging signature. The outputs will either be the embedded signature, or a zero signal.

Autoencoders are a class of neural networks designed to find optimal lower-dimension representations of input data. Structurally, they consist of at least a high dimensional input layer, a low dimensional lower layer, and an output layer of the same size as the input layer. By using the input as the response, the model is trained to create a lower-dimensional representation of the signal in the middle layer, then reconstruct the original signal at the output. The closer the output is to the input, the better is necessarily the hidden representation.
A variation of autoencoders called denoising autoencoders (dAE) attempts to reconstruct a clean response from a noisy input. These can be used to create

Figure 2.1: The training data for the denoiser. Given the input signal, the model will attempt to reconstruct the response.



more robust autoencoders, but in my case, removing the "noise" of non-EV can be viewed as an objective in itself. The goal is thus to create a model which, given an aggregate power signal, attempts to recreate underlying EV-signals.

Figure 2.2: Sample denoiser predictions given the aggregate signal as input. Notice that the denoiser is prone to being fooled by certain high-consumption signals, such as in the beginning of the rightmost example.

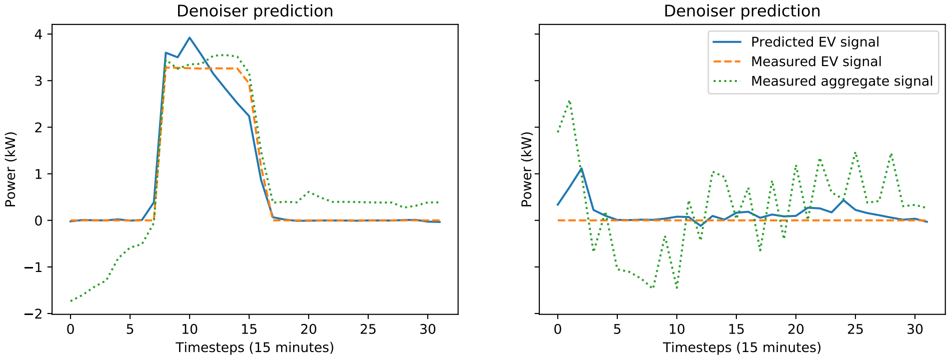
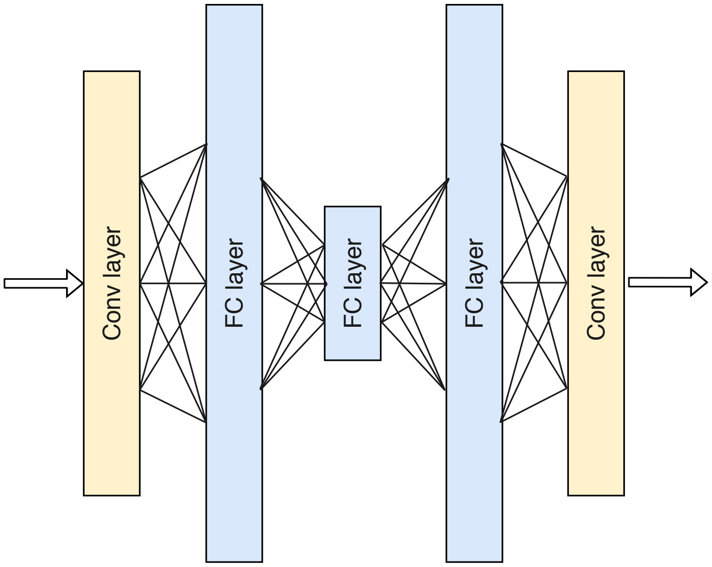


Figure 2.3: The architecture of the denoiser. The weights between the layers do not represent the real structure of the model. The number of actual trainable parameters is 33049.


Conv layer
FC layer
FC layer
FC layer
Conv layer