UPSALA UNIVERSITET

UPTEC STS 24014

Examensarbete 30 hp

Juni 2024

Predicting EV charging Sessions Based on Time Series Clustering

A Case Study from a Parking Garage in Uppsala

Otto Palmlöf

Civilingenjörsprogrammet i system i teknik och samhälle

UPPSALA UNIVERSITET

Predicting EV Charging Sessions Based on Time Series Clustering

Otto Palmlöf

Abstract

Electric vehicles play a crucial role in the global transition towards sustainability, particularly highlighted in initiatives like the European Green Deal. With projections indicating a significant increase in electric vehicle adoption worldwide, including a notable surge in the EU and Sweden, the strain on existing electric infrastructure becomes a concern. Managed charging – the process of regulating the charging of electric vehicles in a coordinated manner – emerges as a promising strategy to mitigate this strain, optimizing charging schedules to alleviate peak loads, and reduce the need for extensive grid upgrades. However, naive peak shaving approaches may fall short in addressing systemic issues, prompting the need for smarter solutions based on predictive modeling. This paper focuses on Dansmästaren, a parking garage designed for mass electric vehicle charging, located in Uppsala, Sweden. Through load shifting techniques, one approach being explored at Dansmästaren aims to avoid grid capacity constraints by strategically scheduling EV charging during off-peak hours, utilizing predictive models for that scheduling.

This paper aims to aid such future predictive models by constructing a new feature for these models to train on, namely clusters. These clusters are made using time series clustering, a technique for assigning different time series to one of a set number of clusters. In this case, the study uses data collected during three months in the form of time series, split by charging sessions, to construct the clusters. The performance of these clusters are then tested using deep learning as a predictive model to evaluate whether or not, and to which degree, the construction of clusters helped the predictive model achieve better results. Different approaches and algorithms are tested and evaluated for the time series clustering with the intention of getting the best possible performance, here meaning the specific construction of clusters resulting in the best performance increase for overall predictions. Different approaches were also tested and evaluated for the deep learning model, although not to the same extent, since the time series clustering is the focus of this paper. In the end, a predictive performance increase of roughly 15% was achieved by the predictive model using the constructed clusters as an additional feature, which can be used in the future.

Teknisk-naturvetenskapliga fakulteten

Uppsala universitet, Utgivningsort Uppsala

Handledare: Alexander Wallberg Ämnesgranskare: Valeria Castelluci

Examinator: Elísabet Andrésdóttir

Populärvetenskaplig sammanfattning

Eldrivna fordon spelar en viktig roll i den gröna omställningen och är en central del i många projekt (inte minst på EU-nivå) vars syfte är att motverka den globala uppvärmningen. I och med detta förutspås antalet eldrivna fordon öka markant de närmsta åren och decennierna – globalt, i EU, samt i Sverige. Detta ställer dock högre krav på elektrisk infrastruktur och nödvändiggör uppgraderingar och utbyggnad av befintlig infrastruktur. Till stor del beror det på att de elektriska fordonen ofta laddar under samma tider på dygnet (exempelvis när folk kommer hem från jobbet), vilket koncentrerar den ökade efterfrågan på el till kortare perioder. Ett sätt att hantera detta är att schemalägga laddningen av de elektriska fordonen till olika tider på dygnet, så att inte lika mycket el behöver tillföras samtidigt, vilket i sin tur gör att inte lika många dyra uppgraderingar av den elektriska infrastrukturen behövs. Detta förutsätter emellertid en vetskap om hur länge ett visst eldrivet fordon kommer behöva ladda, om fordonet ska vara färdigladdat när föraren avser använda det.

På parkeringsgaraget Dansmästaren i Uppsala, som är designat för massladdning av elfordon, görs forskning om elfordonsladdning. Däribland gällande den tidigarenämnda schemaläggningen av laddning. En del av forskningen behandlar metoder för att förutspå hur länge olika elektriska fordon (a) är parkerade, och (b) behöver ladda för att vara färdigladdade. Detta kan göras med avancerade datamodeller som bygger på maskininlärning – program som kräver mycket data och datakraft, men kan prestera väl. Sådan maskininlärning kan i detta fall appliceras för att använda data den får tillgänglig när ett elektriskt fordon kopplas in för att med bästa möjliga träffsäkerhet förutspå när fordonet kommer (a) åka iväg och (b) vara färdigladdat. Kvaliteten av och kvantiteten data som maskininlärningsmodellen har tillgång till påverkar i hög grad hur bra den är på att förutspå detta.

Datan kan vara exempelvis vara information om vilken våning ett fordon står på, vilken tid den kopplades in, eller huruvida det är helgdag eller inte. Denna uppsats undersöker huruvida data från de elektriska fordonens laddningsbeteende kan få maskininlärningsmodeller att prestera bättre, genom att dela in de olika laddningsbeteendena i olika kategorier kallade "kluster". Med laddningsbeteende menas här uppmätta värden på ström och spänning vid olika tidpunkter på de laddstationer de elektriska fordonen laddar vid. Dessa värden utgör för ett laddningstillfälle sammantaget en tidsserie, där mönster och beteenden kan utrönas.

Denna uppsats utforskar metoder att placera dessa tidsserier (och därmed laddningstillfällen) i olika kluster efter en viss tid, i syfte att hjälpa maskininlärningsmodeller bättre förutse när de elektriska fordonen som utgör dessa laddningstillfällen (a) kommer lämna Dansmästren och (b) är färdigladdade. Här har tre månaders insamlad data använts för att konstruera dessa kluster så att påbörjade laddningstillfällen i framtiden kan placeras i de färdiga klustrerna, då det är mycket enklare och mindre tidskrävande att nyttja färdiga kluster än att bygga upp dem från grunden.

Olika tillvägagångsätt och algoritmer har prövats för att få ut bättre prestanda ur klustrerna och modellen som använder dem. Detta har gjorts genom en iterativ process som går ut på att konstruera kluster enligt existerande algoritmer (med olika inställningar och mindre modifikationer) som sen testas i en maskininlärningsmodell likt de som kan användas i framtiden. Maskininlärningsmodellens uppmätta prestanda, alltså hur bra den varit på att förutse (a) och (b) för olika samlingar kluster, avgör sedan hur inställningarna för klustringen (och maskininlärningsmodellen själv) kommer justeras till nästa iteration. Detta gör med tiden att prestandan ökar allt eftersom inställningarna och modifikationerna närmar sig de värden maskininlärningsmodellen presterar bäst med.

I slutändan uppvisade maskininlärningsmodellen en prestandaökning på cirka 15% när den använde sig av de konstruerade klustrerna, jämfört med att inte använda sig av dem. Detta innebär att användingen av kluster kan bidra till bättre prestanda för framtida modeller.

Contents

List of Figures

List of Abbreviations and Terms

API Application Programming Interface

BEV Battery Electric Vehicle

CSV Comma-Separated Values

DTW Dynamic Time Warping

EU European Union

EV Electric Vehicle

GAK Global Alignment Kernel

HTTP Hypertext Transfer Protocol

kWh Kilowatt-hour

MAE Mean Absolute Error

OCPP Open Charge Point Protocol

ReLU Rectified Linear Unit

RCBO Residual current Circuit Breaker with Over-current protection

RFID Radio Frequency Identification

RMSE Root Mean Squared Error

SOC State-Of-Charge

TSkmeans Time Series k-means

1 Introduction

1.1 Context

EVs are generally considered to be an integral part of the "green transition" – the transformation of the world economy towards climate neutrality, in light of accelerating climate change. EVs are addressed and treated directly in, for example, the European Union's (EU) set of proposals called "The European Green Deal", aiming to reduce greenhouse gas emissions by at least 55% by 2030 compared to 1990 levels[1].

The EV fleet is expected to balloon in the coming years, with predictions foretelling an expected worldwide sales increase of 21% and an 8% increase in the EU in 2024, with European EV sales predicted to hit 9.3 million units in 2030 [2, 3].

In Sweden, over 580 000 EVs were in use at the end of 2023. 290000 of these were Battery Electric Vehicles (BEVs) – thanks to an increase of 48% during 2023 alone. The share of EVs of newly registered personal vehicles has doubled from 30% to 60% since 2020. Additionally, the total number of EVs in Sweden is expected to increase further. Furthermore, the number of charging points in Sweden increased by over 75% during 2023 alone, numbering over 34000 in total[4].

A rapidly growing EV fleet would put heavy strain on existing electric grids and other electric infrastructure, with some system peaks possibly increasing twofold[5]. Accommodating this increase could require dramatic investments in electric infrastructure. Furthermore, the actual charging itself could be costly to consumers as prices of electricity increase substantially during peaks. Managed charging (the process of regulating the charging of electric vehicles in a coordinated manner) however, could make sure Electric Vehicle's (EV) are charged outside peak hours, lowering the price of charging for consumers and lessening the need for substantial, otherwise unnecessary, grid upgrades. This would mean that the EVs could use the headroom of off-peak hours to charge, hence not requiring an increase in grid capacity. Or in technical terms, to decrease the load factor (the ratio of the average power demand over a specific period to the maximum power demand during that period) by load shifting. Although some, largely local, grid upgrades would be necessary to handle the increasing need for EV charging, managed charging is a promising technique to reduce unnecessary, costly investments to existing electrical infrastructure as a result of excessive peak loads[6].

Simply "shaving" (implementing measures to decrease electricity consumption during peak periods or shift it to off-peak times) any peaks naively, however, comes with its own issues, as the system perspective is not taken into consideration, and peaks may exceed the grid capacity due to local peak shaving – there are numerous levels of interconnected grids, each one of which can reach capacity. Smarter solutions with larger perspectives are desirable, then. One such approach is to predict when, where, and how much, energy will be drawn.

These predictions can be based on variables that correlate with electricity consumption, such as temperature and wind speed, not to mention historical electricity data and load demand. By scheduling the flow of current, the current draw of EV chargers can be controlled to fit the demands and predictions by load shifting[7].

1.2 Background

This paper concerns one parking garage in specially designed for mass charging of EVs in particular, Dansmästaren (seen in Figure 1). Dansmästaren is the largest parking garage in the city of Uppsala, accommodating 461 parking spaces, of which 60 provide charging for EVs. It is open all day, every day of the year for cars paying a small fee, with an additional fee for charging, regardless of the price of electricity at that moment. Additionally, long-term parking is also available with an additional flat fee for EV charging[8].

Figure 1: Dansmästaren[8].

Dansmästaren was built by two municipal-run companies: Uppsala Parkering and Uppsalahem, a parking company and housing company, respectively. The parking garage's roof is lined with solar panels, which (in tandem with a battery facility) generates electricity that can help reduce the peak loads of the facility. Despite this, managed charging is still beneficial for the parking garage itself, not to mention that Dansmästaren also doubles as a research object for Uppsala University where work on this topic is applicable to other parking garages in Uppsala and beyond. The parking garage is divided into different sections, as can be seen in Figure 2. Here we see the public parking spaces on floors 1 and 2, as well as the long-term parking on floors 3 through 5. note also that floors 4 and 5 have fewer charging points than floors 1 through 3[8].

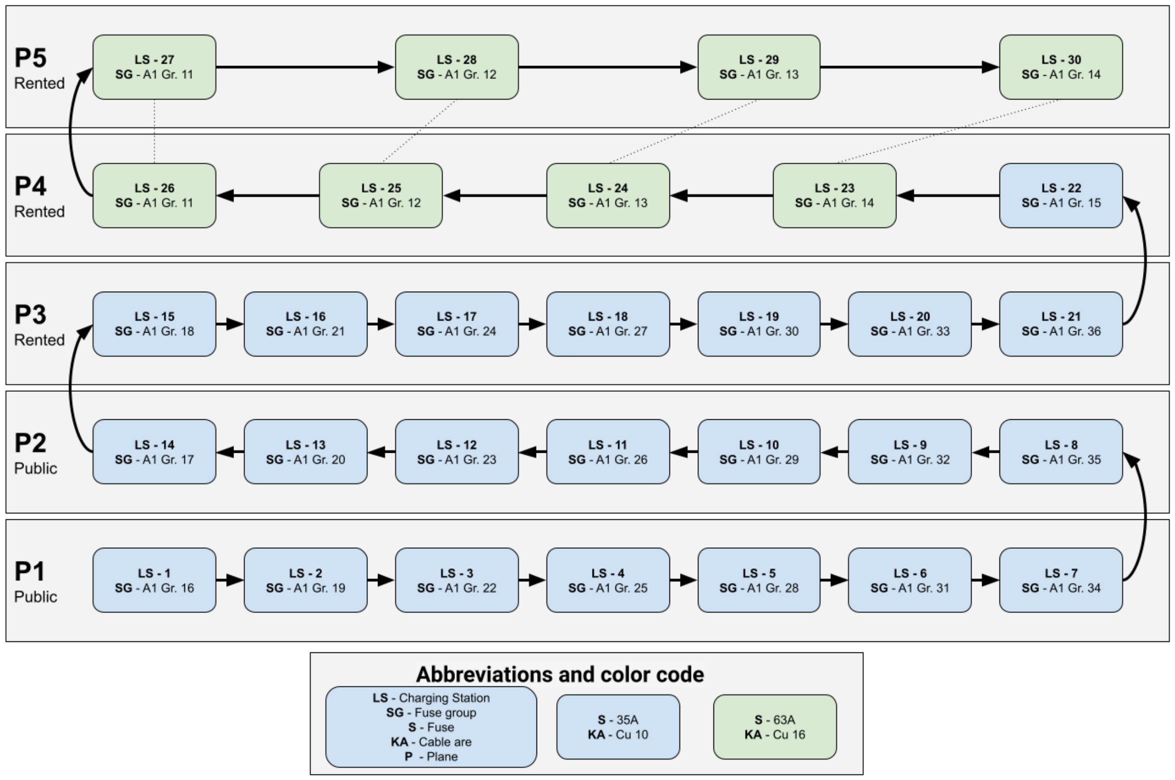


Figure 2: Dansmästaren layout.

1.3 The Bigger Picture

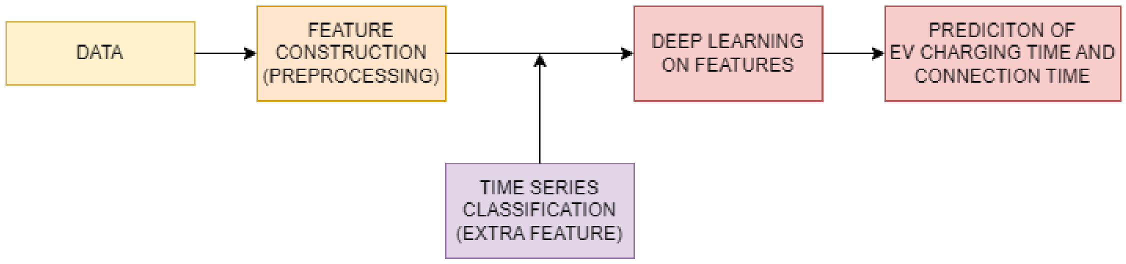
Since this paper deals with Dansmästaren specifically, an overarching goal of the research at Dansmästaren is to load shift to avoid reaching capacity. It is therefore desirable to wait with providing current to EVs that connect during peak hours to later off-peak hours when those EVs are still connected. In this way, any given EV can connect as normal, and be fully charged when disconnected as usual. The actual charging, however, need not have taken place immediately upon connecting, but rather during more off-peak hours. That is to say that the withholding of current to any EV with the goal of overall load shifting for Dansmästaren should only be done if that EV can reliably be predicted to be connected for a long enough time for it to be fully charged regardless of when it is charged during its connection. Or, in plain language, that no EV user will disconnect their EV and find that it has not been charged enough due to misapplied load shifting.

This type of control requires accurate and reliable predictions of how long any EV will (a) be connected, and (b) be charging, so the required time for charging can reliably be accommodated into the actual connection duration. The definitions and quantifications of what "accurate" and "reliable" might be, and how it is to be applied, is beyond the scope of this paper. Rather, this paper focuses on making predictions of how long any EV is charging as accurately as possible, though the same approach can be taken to predict how long any EV is connected.

An often used method of making predictions like these is using deep learning, which takes a number of dimensions of data (called "features"), usually all of substantial size, and "learns" from that by the answer, being a dimension itself, being hidden during learning, so it can predict using future data that is not yet available (or even existent). Since different EVs have batteries of different sizes, and therefore charge for different durations and possibly behave differently with regards to charging, identifying different types of EVs by charging behavior could be beneficial when using deep learning to predict charging durations. A common method of doing this without pre-defined categories is clustering. If the input data is time series data, which it would be if charging behavior is what is considered (since charging is a process which takes place over time), then using a specialized form of clustering called "time series clustering" is the standard approach for this task. It differentiates different time series from each other, grouping similar time series together, and places them into different categories. These categories can then be used as a feature for deep learning to use.

As deep learning processes and learns from different features of the data, more features can potentially mean more usable information to allow for the deep learning model to perform better. This depends on how usable the features are, since noise, for example, does at best not improve the model performance, and can potentially cause it to perform worse. The hope is, therefore, that the time series clustering can, by assigning a cluster number to each charging session, provide meaningful information for the deep learning model. This is to say that the cluster number of a particular charging session is a new feature of the meta data that the deep learning model can process and learn on. Since time series clustering is done on time series that are already finished and available, and is very time consuming, time series classification would rather be used in future use cases. In other words, time-consuming and cumbersome time series clustering is used beforehand (in this paper) so as to allow for fast, elegant Time Series Classification for future use, since time series classification requires already defined clusters. This future workflow is illustrated in Figure 3.

To summarize, the goal of this paper is to answer the questions (i): can using time series clustering aid a deep learning model in better predicting charging durations; and (ii): what can a set of settings, approaches, and hyperparameters well suited for that be? This is to be done while exploring and presenting the data so as to place the goal in perspective.



DATA
FEATURE CONSTRUCTION (PREPROCESSING)
DEEP LEARNING ON FEATURES
PREDICITON OF EV CHARGING TIME AND CONNECTION TIME
TIME SERIES CLASSIFICATION (EXTRA FEATURE)

Figure 3: Desirable future workflow.

2 Theory

2.1 Choice of methods and algorithms

Though there (to the author's best knowledge) has been no previous studies done combining time series clustering and deep learning in the way outlined above, previous studies have been done in similar areas, concerning several parts touched upon in this paper. Using state-of-charge data, k-means clustering has been performed on groups of EVs to predict various behaviors, for the purpose of managed charging[9]. Additionally, machine learning in general and deep learning in particular has shown promise for analysis of EV data and EV behavior prediction[10] [11].

Time series clustering has been performed on EV charging data in previous studies, using methods such as DTW further explored below[12]. Time series clustering for EVs has also been done for group of EVs, taken as a whole and not on an individual level[13].

Combination of clustering and deep learning methods for predictions has occurred in previous studies, though the clustering is not specifically time series clustering, as is the case in this paper. Using a sampling rate of 0.1 Hz, over 220000 charging records were collected during a full calendar year to perform predictions using clustering and deep learning in tandem[14].

In these aforementioned studies, the methods outlined in the rest of Section 2 have occurred and shown promise. The time series clustering methods have however not been paired with deep learning in the way performed in this study. This study contains therefore, to the author's best knowledge, a novel approach to EV charging duration prediction.

2.2 Deep Learning and Activation Functions

Representation learning, particularly within the domain of deep learning, refers to the automatic discovery of patterns inherent in raw data, usually for classification- or regression tasks. Deep learning models are characterized by multiple layers of representation, where each layer progressively transforms the input into a more abstract form. These transformations can amplify crucial features while dampening irrelevant variations. In other words, deep learning is a subset of machine learning methods utilizing multiple layers of neural networks, which are models of connected neurons arranged in layers. These networks autonomously identify patterns in data, often in an unsupervised manner, through the iterative processing of input data. By iteratively adjusting the parameters (weights and biases) through stochastic gradient descent (an iterative method for optimizing an objective function), deep learning systems minimize errors and enhance their ability to generalize to unseen data. This iterative optimization process involves computing gradients to determine how adjustments to each weight affect the overall error, thereby aligning the model toward convergence on an optimal solution[15].

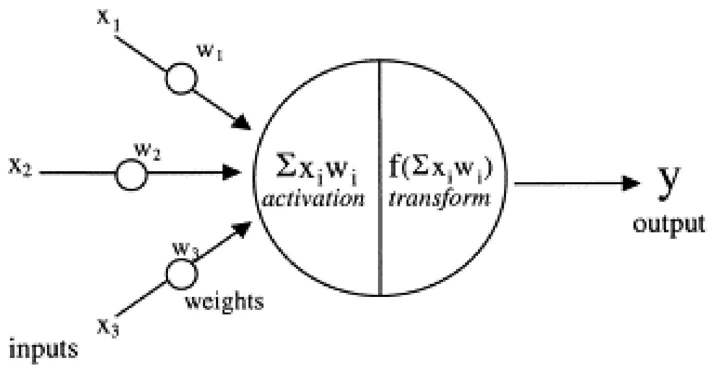


Figure 4: The workings of an activation function[16].

Deep Learning models have a number of adjustable hyperparameters (variables of the neural network) that are set prior to to optimization. One of these is the choice of activation function(s), which virtually every deep learning model uses one or more of by design, as they are a standard practice of improving performance of neural networks and deep learning models. Usually, an activation function is used after every dense layer of neurons except for the input layer, taking the outputs from the layer and applying a the function on them as a means of normalization, before outputting to the next layer (or model output), as visualized in Figure 4. Three activation functions are treated in this paper by virtue of being common. These are the Rectified Linear Unit (ReLU) activation function defined as

$$f(x) = \max(0, x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases} \tag{1}$$

, the sigmoid activation function defined as

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

, and the tanh (hyperbolic tangent) activation function defined as

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3}$$

[17].

2.3 Time Series Clustering

In the realm of data analysis, classification and clustering are two fundamental techniques with distinct methodologies and applications. Classification involves assigning predefined labels to data points based on their features. On the other hand, clustering aims to uncover natural groupings within data without prior labels, enabling exploratory analysis and pattern discovery. In this way, classification requires predefined labels, whereas clustering (while not necessarily naming the labels as such), creates its own categories, the number of which is usually determined in advance. Conventionally, any given data point belongs to one and only one cluster, as will implicitly be the case in this paper. Clustering is usually performed by in some way measuring the distances between data points and/or "barycenters" (also called "centroids" in the realm of general clustering) – a definition of a center for one or more clusters. It is used in both normal clustering –where it is a representative location for the cluster as a whole (a mean, for example)– and in time series clustering, where it represents the average behavior or characteristic pattern of time series within a cluster[18].

With time series however, like the one in Figure 5, there is no immediately obvious way to measure distance. While most time series data is discrete, the way to measure distances will have to be decided, in addition to determining which distances should be measured at all, and to which degree and in what way they should contribute to the clustering. In other words, there is no obvious similarity measure. This opens for opportunities and challenges within the field of clustering time series – time series clustering.

Time-series clustering has been employed across various scientific disciplines to uncover patterns, enabling data analysts to glean valuable insights from intricate and vast datasets. This can be challenging however, due to time series being naturally temporal (that is to say, ordered), high dimensional and large in data size. Additionally, a time series can be seen as a number of data points, or as a single object. Furthermore, if different data is part of the same time series at the same point in time (that is, data with dimensionality higher than 1), it is not obvious how this should be handled, or what to do in the case of time series of varying length. The time series clustering algorithms that are to follow were chosen because they have previously shown to perform well[19][20].

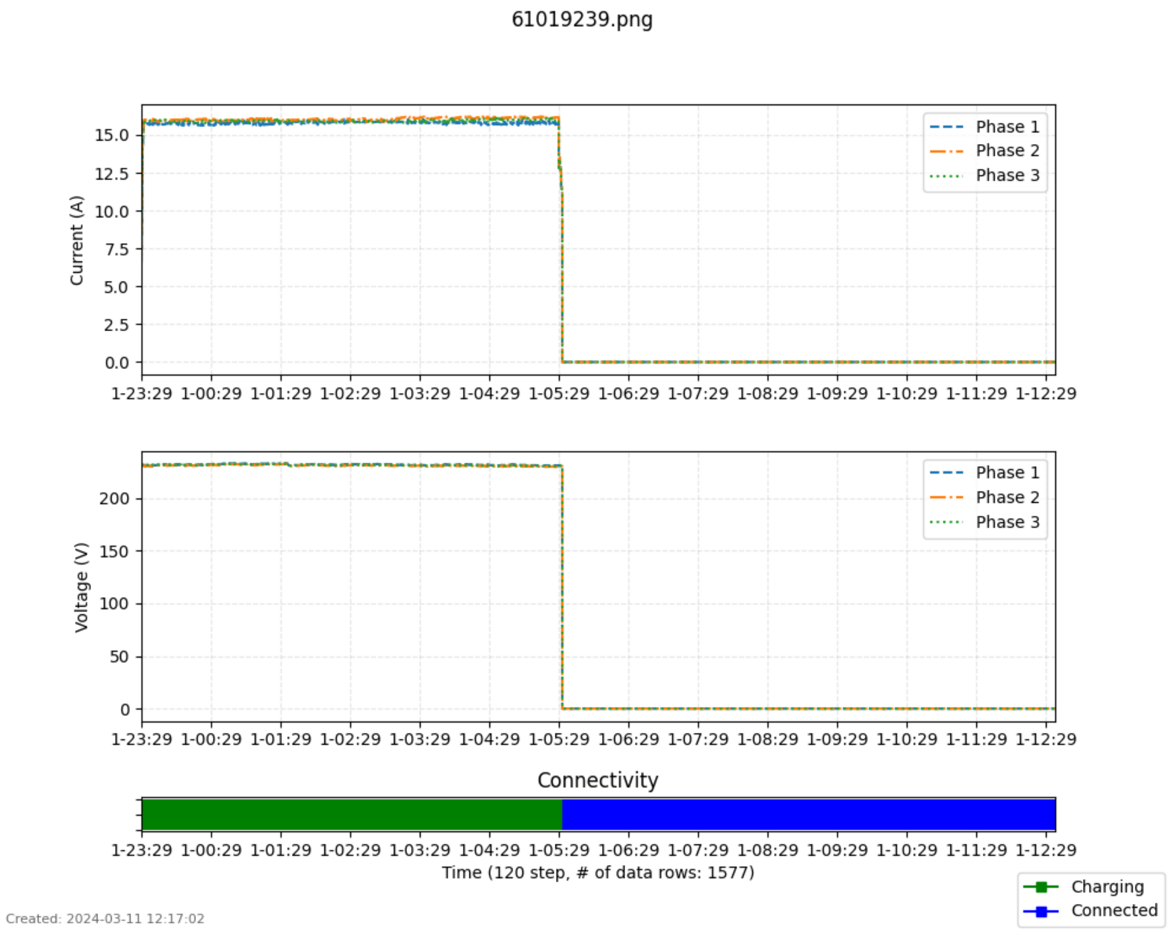


Figure 5: Example of how a time series can look at Dansmästaren.

2.4 Dynamic Time Warping

DTW is an algorithm first introduced back in the 1960s, and is today often used for time series clustering. It is a time series similarity measure that utilizes elasticity to detect similarities between time series despite possible shifts and distortions[21].

Suppose there are two sequences $\mathbf{X} = \{x_0, x_1, \ldots, x_{n-1}\}$ and $\mathbf{Y} = \{y_0, y_1, \ldots, y_{m-1}\}$ that we want to compare. Here, all elements x_i and y_j are assumed to lie in the same d-dimensional space and the absolute times at which observations occur are irrelevant: only their ordering matters. Essentially, DTW minimizes the Euclidean distance between time series under all allowed temporal assignments (meaning matches between points in the two time series). It is also possible to use other distance metrics within DTW, though these not used in this paper. DTW for two points x and y can be formulated as (when using euclidean distance) solving the optimization problem

$$DTW(x, y) = \min_{\pi \in \mathcal{A}(x,y)} \sqrt{\sum_{(i,j)\in\pi} \|(x_{(i)} - y_{(j)})\|^2} \tag{4}$$

[22] where, an alignment path $\pi = [\pi_0, \ldots, \pi_{K-1}]$ is a list of index pairs $\pi = ((i_k, j_k))$ with $0 \leq i_k < n$ and $0 \leq j_k < m$ that satisfies the following conditions:

1. The starts and ends of time series are matched: $\pi_0 = (0, 0)$, $\pi_{K-1} = (n-1, m-1)$.
2. The sequence is monotonically increasing in both i and j, and all time series indexes have to appear at least once:

$$i_{k-1} \leq i_k \leq i_{k-1} + 1, \quad j_{k-1} \leq j_k \leq j_{k-1} + 1.$$

In this way, a path π can be seen as a set of matching points in two time series that are matched in such a way that the sum of the Euclidean distances between matched points (each point having to be matched to at least one other point) is minimized. This matching is visualized in Figure 6 together with a simple Euclidean matching of the points x_i and y_j by setting $i = j$[23].

Applying DTW for computing time series barycenters μ for all points D using sum of squared distances corresponds to the optimization problem

$$\min_{\mu} \sum_{x \in D} \mathrm{DTW}(\mu, x)^2$$

[22].

The optimization for DTW can be solved by an $O(mn)$-algorithm (here indexed from 1):

```
1: procedure DTW($x, y$)
```

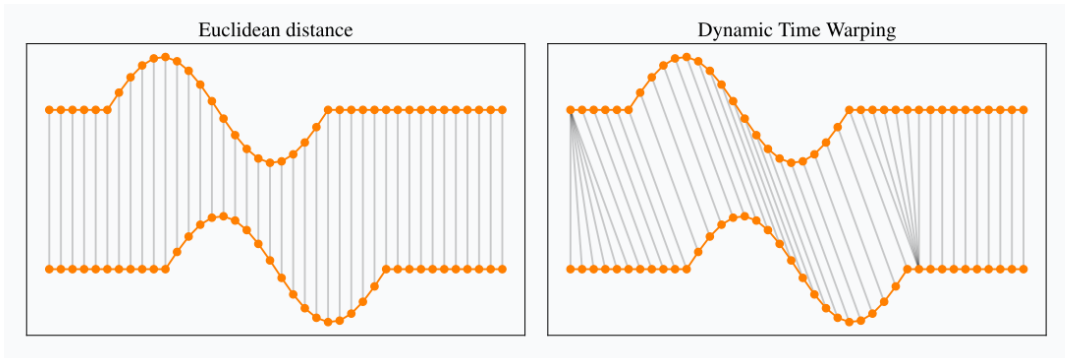


Figure 6: DTW visualization[23].

```
2:      Initialization
3:      for $i = 1$ to $n$ do
4:          for $j = 1$ to $m$ do
5:              $C[i, j] \leftarrow \infty$
6:          end for
7:      end for
8:      $C[0, 0] \leftarrow 0$
9:
10:     Main loop
11:     for $i = 1$ to $n$ do
12:         for $j = 1$ to $m$ do
13:             dist $\leftarrow d(x_i, y_j)^2$
14:             $C[i, j] \leftarrow$ dist $+ \min(C[i-1, j], C[i, j-1], C[i-1, j-1])$
15:         end for
16:     end for
17:
18:     return $\sqrt{C[n, m]}$
19: end procedure
```

[22].

A modification of DTW called "soft-DTW" has been developed which unlike DTW is differentiable. The differentiability of the soft-DTW function allows for use of the chain rule, which makes for better optimization in machine learning models utilizing gradient-based optimization techniques. Furthermore, barycenters for soft-DTW can be estimated through gradient descent, thanks to this[24].

The non-differentiable part of DTW is the min-function, which has been replaced in soft-DTW with the newly formulated soft-min function

$$\text{soft-min}_\gamma(a_1, \ldots, a_n) = -\gamma \log \sum_i e^{-\frac{a_i}{\gamma}} \tag{5}$$

[22],

where γ is a hyper-parameter that controls the smoothing of the resulting metric. This smoothing, an inherent feature in soft-DTW, has been hypothesized to be one of the reasons why soft-DTW has been shown to outperform DTW: for smaller values of γ soft-DTW smoothes out local minima – which the algorithm is less likely to get stuck in, then – and provides a better (although different) optimization landscape[24].

2.5 Time Series k-means

One common time series clustering algorithm is Time Series k-means (TSkmeans). Amongst other things, it assigns different weights to different time stamps depending on the importance of the time span within the time series it is a part of. These weights are constrained to so as to induce a smooth subspace in the clustering process. TSkmeans iteratively discovers subspaces of the entire time series, and clusters the time series based on the subspaces instead of the entire time series. Furthermore, the algorithm smooths the weight of adjacent time stamps so as to use sequence information regarding their chronological order.

It works as follows. Let $X = \{X_1, X_2, \ldots, X_n\}$ denote a set of n time series objects. Each object $X_i = \{x_{i1}, x_{i2}, \ldots, x_{im}\}$ is characterized by m values corresponding to m timestamps. The membership matrix U is an $n \times k$ binary matrix, where $u_{ip} = 1$ indicates that time series object i is assigned to cluster p, otherwise, it is not assigned to cluster p. The barycenters of the k clusters are represented by a set of k vectors $Z = \{Z_1, Z_2, \ldots, Z_k\}$. Similarly, $W = \{W_1, W_2, \ldots, W_k\}$ denotes a set of k vectors representing the weights of the timestamps for each cluster. The value of element w_{pj} indicates the weight of the jth timestamp for the pth cluster. The objective function of TSkmeans is formulated as

$$P(U, Z, W) = \sum_{p=1}^{k} \sum_{i=1}^{n} \sum_{j=1}^{m} u_{ip} w_{pj} (x_{ij} - z_{pj})^2 + \frac{1}{2}\alpha \sum_{p=1}^{k} \sum_{j=1}^{m-1} (w_{pj} - w_{pj+1})^2, \tag{6}$$

,subject to the constraints

$$\begin{cases} \sum_{p=1}^{k} u_{ip} = 1, \quad u_{ip} \in \{0, 1\} \\ \\ \sum_{j=1}^{m} w_{pj} = 1, \quad 0 \leq w_{pj} \leq 1 \end{cases}$$

[25].

The parameter α is used to balance the effects between the scatter of objects within clusters and the smoothness of the weights of timestamps. The smoothness of the weights among adjacent timestamps increases with the increment of α's value. The first term of the objective function aims at minimizing the sum of scatters of all the clusters. The second term of the objective function smoothes the weights of the adjacent timestamps. In the clustering process, this objective function simultaneously minimizes the within-cluster scatter and smooths the weights of the adjacent timestamps[25].

Similarly to traditional k-means, TSkmeans is an iterative algorithm that updates the membership matrix U, the barycenters Z, and the weights W. TSkmeans minimizes the objective function by iteratively fixing two of the variables of U, Z, and W, while obtaining the values of the other variables. Which is to say that using the derivatives of the objective function with respect to U, Z, and W respectively, they are set to zero, and the values for the non-set variables are acquired. This process continues until convergence. Experimentally, this algorithm has shown to outperform other common time series clustering algorithms[25].

In this formulation of the objective function, the distance between any timestamp value x_{ij} and barycenter value z_{pj} is measured using the euclidean distance. The distance can however also be measured using the DTW and soft-DTW algorithms outlined above. Furthermore, the barycenters Z can also be measured using DTW, a technique called "DTW barycenter averaging". It aligns the time samples to each other as DTW would before measuring the distance, increasing accuracy of the overall algorithm[26].

2.6 Kernel k-means

Another method for time series clustering is done with a weighed kernel k-means using a Global Alignment Kernel (GAK). Kernel k-means is so called because they utilize so-called "kernel functions", which transform linearly inseparable data into linearly separable data by mapping them to higher-dimensional space. This is done by using the "kernel trick", allowing for computation of the inner product of two non-linearly transformed points $\phi(a)$ and $\phi(b)$ without explicitly transforming the transformation of each point. Examples of such a kernel functions are the Polynomial Kernel $\kappa(a, b) = (a \cdot b + c)^d$, and the Gaussian Kernel $\kappa(a, b) = e^{-\frac{\|a-b\|^2}{2\sigma^2}}$. In this way, kernel functions defines similarity measures between the vectors a and b corresponding to their inner product. These are then saved in a kernel matrix K[27].

Additionally, the kernel k-means algorithm can be generalized by adding a weight $w(\mathbf{a})$ for each point $\mathbf{a}$. Taken together, denoting clusters by π_j, a partitioning of points as $\{\pi_j\}_{j=1}^{k}$ (k being the number of clusters), and the best cluster representative as $\mathbf{m}_j$ the objective function of weighed kernel k-means is defined as

$$D\left(\{\pi_j\}_{j=1}^{k}\right) = \sum_{j=1}^{k} \sum_{\mathbf{a}\in\pi_j} w(\mathbf{a})\|\phi(\mathbf{a}) - \mathbf{m}_j\|^2 \tag{7}$$

, where

$$\mathbf{m}_j = \frac{\sum_{\mathbf{b}\in\pi_j} w(\mathbf{b})\phi(\mathbf{b})}{\sum_{\mathbf{b}\in\pi_j} w(\mathbf{b})}$$

[27] , where the Euclidean distance from $\phi(\mathbf{a})$to center $\mathbf{m}_j$ is given by

$$\left\|\phi(\mathbf{a}) - \frac{\sum_{\mathbf{b}\in\pi_j} w(\mathbf{b})\phi(\mathbf{b})}{\sum_{\mathbf{b}\in\pi_j} w(\mathbf{b})}\right\|^2 = \phi(\mathbf{a})\cdot\phi(\mathbf{a}) - \frac{2\sum_{\mathbf{b}\in\pi_j} w(\mathbf{b})\phi(a)\cdot\phi(\mathbf{b})}{\sum_{\mathbf{b}\in\pi_j} w(\mathbf{b})} + \frac{\sum_{\mathbf{b},\mathbf{c}\in\pi_j} w(\mathbf{b})w(\mathbf{c})\phi(\mathbf{b})\cdot\phi(\mathbf{c})}{(\sum_{\mathbf{b},\mathbf{c}\in\pi_j} w(\mathbf{b}))^2} \tag{8}$$

[27].

Using this objective function, the pseudocode for the weighted Kernel k-means looks like

Algorithm 1 Weighted Kernel K-Means

```
1: procedure WeightedKernelKMeans($K, k, w, C_1, \ldots, C_k$)
2:     Input: $K$: kernel matrix, $k$: number of clusters, $w$: weights for each point
3:     Output: $C_1, \ldots, C_k$: partitioning of the points
4:     Initialize the $k$ clusters: $C_1^{(0)}, \ldots, C_k^{(0)}$.
5:     Set $t = 0$.
6:     repeat
7:         for each point $a$ do
8:             Find its new cluster index as $j^*(a) = \arg\min_j \|\phi(a) - m_j\|^2$ using 8
9:         end for
10:        for $j = 1$ to $k$ do
11:            Compute the updated cluster $C_{t+1}^{j} = \{a : j^*(a) = j\}$.
12:        end for
13:        $t = t + 1$
14:    until Convergence
15: end procedure
```

[27].

GAK uses this and applies it to time series by using DTW as a distance measurement for the kernel. It is defined as the exponentiated soft-minimum of all alignment distances(recall notation in 2.4)

$$k_{GA}(\mathbf{x}, \mathbf{y}) \overset{\text{def}}{=} \sum_{\pi\in A(n,m)} e^{-D_{x,y}(\pi)} \tag{9}$$

, where $D_{\mathbf{x}}, \mathbf{y}$ is defined as the cost

$$D_{x,y}(\pi) \overset{\text{def}}{=} \sum_{i=1}^{|\pi|} \|(x_{\pi_1(i)} - y_{\pi_2(i)})\|^2 \tag{10}$$

, which is to say 4 without the minimizing function and square root. GAK is the kernel function used in the kernel k-means clustering in this paper[28][29].

2.7 kShape

kShape is a centroid-based clustering algorithm that can preserve the shapes of time-series sequences. Its distance measure is based on a cross-correlation measure, which helps compute the centroids. The algorithm then iteratively refines the centroids and their clusters. Since other well-performing distance measures for time series clustering that can handle distortions of and amplitude and phase are computationally expensive, such as DTW, kShape instead circumvents this efficiency limitation by adopting a normalized version of the cross-correlation measure.

Cross-correlation is a measure of similarity that compare one-to-one points between signals, as opposed to DTW which compares one-to-many and one-to-none points. It does this by shifting one time series ($\vec{x}$) over another ($\vec{y}$), so that the inner product of the matching points (samples) are maximized. So a cross-correlation measure

$$\mathrm{CC}(\vec{x},\vec{y}) = \max_{k}\left\{\sum_{n=1}^{N}\vec{x}(n)\cdot\vec{y}(n-k)\right\} \tag{11}$$

yields a sequence $CC_w(\vec{x},\vec{y}) = (c_1, ..., c_w)$ of the shifts producing the highest inner products, which is possible thanks to the sequences $\vec{x}$ and $\vec{y}$ being padded with zeroes when necessary. After a standardization of the time series to deal with scale invariance, the shape-based distance measure of kShape is defined as

$$SBD(\vec{x},\vec{y}) = 1 - \max_{w}\left(\frac{\mathrm{CC}_w(\vec{x},\vec{y})}{\sqrt{R_0(\vec{x},\vec{x})\cdot R_0(\vec{y},\vec{y})}}\right) \tag{12}$$

, where $R_0(\vec{x},\vec{y})$ is the the geometric mean of autocorrelation of the individual sequences

$$R_0(\vec{x},\vec{y}) = \sum_{l=1}^{m} x_l \cdot y_l \tag{13}$$

. Then, by using this distance measure, the computation of centroids is done by minimizing the sum of squared distances to all other time series sequences[30].

3 Method

3.1 Data

As preparation for the project, the data was explored firstly by simple overview of the size and shape of the data, and later by plotting all the time series data available per charging session, as seen in Figure 5. As can be seen from the raw data as well as the plots, there are differences in the current and voltage values of the different time series, as well as different shapes of the current- and voltage curves in the plots. These differences show promise that the different time series can be categorized non-arbitrarily, which is to say that they can preliminarily be deemed fit for potential time series clustering. The data used consisted of roughly three months worth of data collected from Dansmästaren between 2023-12-18 to 2024-02-14, stored in 1878 Comma-Separated Values (CSV)-files, one per charging session.

When the predictions of charging times take place a few factors are of importance, given that varying amounts of information are available at different times. Examples of when predictions can be made and what information they require are:

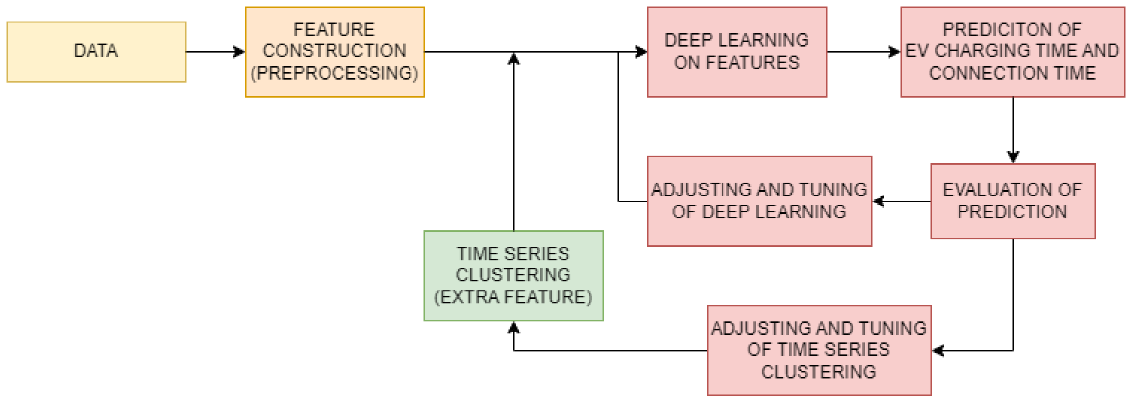
- **Day** – a prediction made for every day predicting Dansmästaren occupancy and load.
- **Immediate** – a prediction made for a specific charging session when an EV connects.
- **Intermediate** – a prediction made for a specific charging session after an EV has been connected for a set amount of time.

These are not exclusive, and a succeeding prediction done when more data is available need not entirely replace an earlier prediction. A more fitting approach would be to merely revise it. This is particularly clear when it comes to the scope of potential predictions. Whereas the "day"-prediction take all charging stations as part of a whole (namely Dansmästaren as an entity), the "immediate" and "intermediate" predictions are specific for that charging session, and consequently EV and charging port. As such, one approach could be to make "day"-prediction for every day, which can depend on things like the degree of occupancy at Dansmästaren and ambient temperature. This, in turn, can be revised whenever a charging session is initiated, which can depend on things like which charging point is used and the time of connection. Finally, this can in turn be revised after a set amount of time, at which point specifics regarding the behavior of the charging is available, for example how many Kilowatt-hours (kWhs) the connected EV accumulates per minute. This last revision can of course be made repeatedly at different times if so desired. It shall be noted that the approach and structure of this is beyond the scope of this paper, and the preceding section is merely an example of how charging session predictions can be made and used, so as to place this paper into perspective.

Using this new terminology, and the assumption that the necessary work has been carried out, question (i) can be rephrased as: did using time series clustering in addition to the intermediate features significantly improve predictions compared to only using the intermediate features?

3.2 Project Workflow

To achieve the future workflow outlined in the introduction in general and in Figure 3 in particular, the time series classification must have ready-made clusters to classify the different time series as, which is the goal of this paper. This is to say that the deep learning on features done to predict EV charging- and connection time that is done in this paper is only done preliminarily as a means to measure the performance of the time series clustering and not as some sort of final prediction model. The reasoning behind this is that the normally used various measures for the performance of time series clustering are not specialized for the usage of these time series clusters to the extent a dummy for a future prediction model is, as it is designed with that very intent. Furthermore, it is not clear what constitutes a good cluster in and of itself. What makes a cluster good for the purpose of this paper is an ability to make a prediction model output more accurate results compared to not using the clusters, and so what even defines good cluster is dependent on such a prediction model. With this in mind, the workflow used in this project is outlined in Figure 7.



DATA
FEATURE CONSTRUCTION (PREPROCESSING)
DEEP LEARNING ON FEATURES
PREDICITON OF EV CHARGING TIME AND CONNECTION TIME
ADJUSTING AND TUNING OF DEEP LEARNING
EVALUATION OF PREDICTION
TIME SERIES CLUSTERING (EXTRA FEATURE)
ADJUSTING AND TUNING OF TIME SERIES CLUSTERING

Figure 7: Project workflow.

The "adjusting and tuning of deep learning" in Figure 7 is therefore only done so as to make the prediction model perform reasonably well, without delving too deep into the different possible architectures, designs, and parameters that a model good enough for actual prediction of EV charging- and connection would have. Despite this, reasonable adjustments to hyperparameters and smaller basic architecture changes are explored in a limited fashion. This is not to say that the prediction model is necessarily subpar at predicting charging- and connection times – the evaluation of its performance is simply beyond the scope of this paper. The exception to this is comparison within the model. That is, a comparison between the performance of the deep learning prediction model when using clusters and when not, as a means to the end of evaluating the performance of the time series clustering.

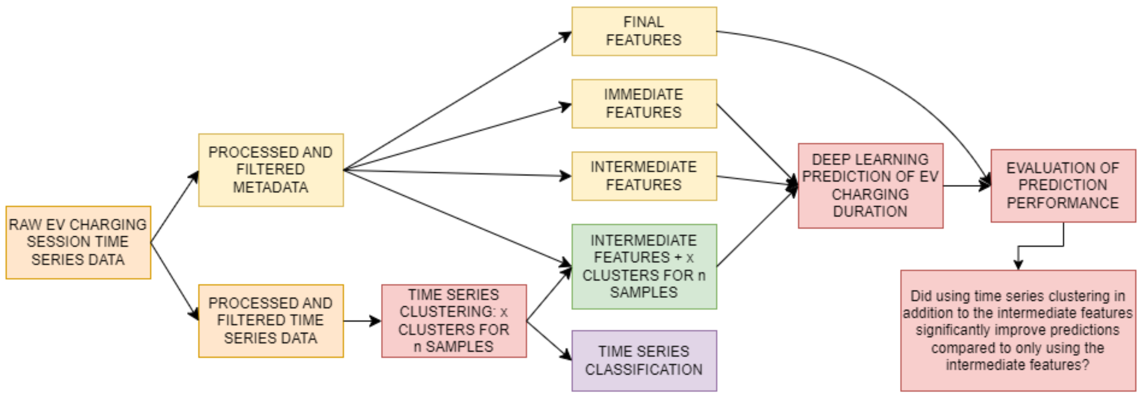

RAW EV CHARGING SESSION TIME SERIES DATA
PROCESSED AND FILTERED METADATA
PROCESSED AND FILTERED TIME SERIES DATA
TIME SERIES CLUSTERING: x CLUSTERS FOR n SAMPLES
FINAL FEATURES
IMMEDIATE FEATURES
INTERMEDIATE FEATURES
INTERMEDIATE FEATURES + x CLUSTERS FOR n SAMPLES
TIME SERIES CLASSIFICATION
DEEP LEARNING PREDICTION OF EV CHARGING DURATION
EVALUATION OF PREDICTION PERFORMANCE
Did using time series clustering in addition to the intermediate features significantly improve predictions compared to only using the intermediate features?

Figure 8: Project graphic overview.

How the actual data is handled in this project is outlined in Figure 8, though explored in more detail in section 3.3. Essentially, the raw time series data is processed and filtered to only use appropriate data in relevant formats. In addition to this, "metadata" is constructed out of the time series data, where data on each time series is contained, rather than the time series itself (for example, how many kWh were accumulated during the session). The metadata is linked to the time series data by ID, so they can be identified and associated.

This metadata is then piece-wise copied into different dataframes such that data in one of the resulting dataframes may, but need not, contain the same data as another dataframe. All these dataframes are also linked by the same ID as has been previously used. These dataframes are differentiated on the grounds of when the data contained in them can be constructed during a charging session. In other words, the features that are available at the respective times thusly:

- "**Immediate Features**" contains the data of the features that are available upon the first sample of a charging session, for example the time at which the session started.
- "**Intermediate Features**" contains (in addition to all data available in **Immediate Features**) the data of features that are available after a few samples, for example how many kWh have been accumulated thus far.
- "**Final Features**" contains (in addition to all data available in **Intermediate Features**) the data of features that are available when the session has finished, for example if the EV is to be considered fully charged when disconnected.

In parallel to this split of the metadata, the time series data is prepared for time series clustering by extraction of the n first samples. One reason for this is that time series clustering is very time consuming, and longer time series simply means longer time series clustering, which can become unwieldy. Furthermore, though DTW can be applied to time series of different length, Euclidean measures cannot. The main reason, however, for extracting the first n samples, however, is that the proposed time series classification is proposed to (1.3) take place some time after (= a number of samples after) a charging session has been initiated, and the amount of samples n available at time t is of course equal to $\frac{t}{\omega}$ at sample frequency ω. In this paper, time series clustering is explored at $t = 15, t = 30, t = 45$, and $t = 60$ minutes, which at the sample rate used at Dansmästaren at the moment, $\omega = 0.5$ samples/minute, equals $n = 30, n = 60, n = 90$, and $n = 120$ samples. These numbers were chosen arbitrarily, though with the reasoning that too high a number would mean that a large share of all considered EVs have already disconnected by then, and that the clustering would presumably not get much better. A number too low on the other hand, would presumably not contain enough data to perform well.

Although that was the reasoning behind the number of sample values initially chosen, in the possibility of good predictions using the lowest value $n = 30$, further similar values would be explored. Though these values for n were chosen, the same approach (and indeed, codebase) used to achieve the results for these amount of samples can be done for any value $2 \leq n \leq l$, where l is the number of samples contained in the longest time series under consideration.

The two relevant features to predict both concern durations. One how long an EV is connected, and the other how long an EV is charging. Both can be relevant with respect to the goals of this paper, though adjusting the deep learning and time series clustering for both simultaneously would be cumbersome and time consuming. Additionally, a greater focus and easier overview of the results is made possible when only focusing on one of these two features. For these reasons, prediction of charging duration is the one made in this paper, unless explicitly stated otherwise. Brief results of the same model instead predicting EV connection time will also be presented. The same approach this paper uses, and indeed codebase, could be used for adjusting the settings for instead predicting how long an EV is connected. Given the high correlation between these two features, however, the adjusting would likely not be major, or perhaps not required at all, since a deep learning model is supposed to perform regression again for any given EV, and the feature to be predicted is set in that stage.

3.3 Data Acquisition and Processing

3.3.1 IDE (Integrated Development Environment)

As it is a common programming language with widely used and available libraries for data preprocessing, time series clustering, and deep learning; Python was chosen to be used for this paper. Microsoft's editor "Visual Studio Code" was used for programming as it has native Python support, and is a popular and versatile editor. The developer platform Github was used to backup and handle files and for version control. The Python libraries "Pandas" and "NumPy" were used for preprocessing and various data handling, -reading, and -saving. Additionally, the Python libraries "scikit-learn" and "TensorFlow" were used for data processing and deep learning as they are commonly used and well documented, and "tslearn"[31] was used for time series processing and -clustering, as it is specifically designed for those purposes. The data was downloaded as CSV-files, as that is a common format for data storage, and supported by Pandas and NumPy.

3.3.2 Chargers

In the parking garage Dansmästaren, the 60 charging ports for EVs consists of 30 "Aura" charging stations, seen in Figure 9, manufactured by the company "Charge Amps". It is suitable to Dansmästaren as it is equipped with two 22 kW sockets (and so 30 charging stations means 60 charging ports), and the current draw can be configured remotely using the ISO 15118 standard via Open Charge Point Protocol (OCPP)-compatible cloud services. This allows for the aforementioned scheduling and load balancing[32, 33, pp. 24-25].

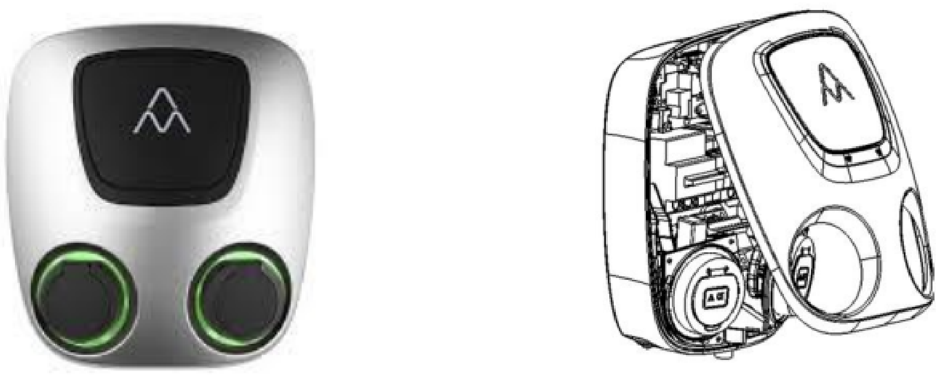


Figure 9: Aura charger[32, p. 24][33].

Feature	**Details**
Part number	101010 and 1309009
Charging current	6 A to 32 A, 1 or 3-phase per socket
Voltage	230 V and 400 V
Charging standard	Mode 3
Standard	EN 61851-1:2011
Operating temperature	−30 °C to 45 °C
Identification	RFID for both outlets
Internet connection	WiFi, LAN and 4G* (applies to certain models)
Communication protocol	OCPP 1.6J
Connection	Wi-Fi (2.4 GHz), LAN 2-port switch (IEEE 802.3 IOBASET / IOOBASE-TX) for network connection. (Possibility of 4G connection via 4G router)
Fault current protection	Internal RCBO type A + 6 mA DC monitoring (corresponds to JFB type B)
Metering	3-phase voltage, current and output power (equivalent to the Measuring Instruments Directive 2014/32/EU)
IP rating	IP 55
IK rating	IK 10
Sockets	Type 2, 2 × 22 kW (32 A) simultaneously
Dimensions (W x D x H)	$367x159x40$ mm
Weight	10 kg
Cable area connection	2.5 mm^2 to 15 mm^2
Mounting method	Wall or post
Lock	Mechanical lock for access

Table 1: Specification Details[32, p. 25][33].

3.3.3 Loading

The collection of the charging data is done, as per previous research on Dansmästaren, using the OCPP-compatible cloud service "My Charging Space", a multi-plant management portal for extraction of information from EV charging stations. In this, total charged energy, status of the chargers and their socket, charging session measurements can be accessed, and remote-control actions can be taken. This is implemented using the REST Application Programming Interface (API) "Swagger API", with which the data is fetched using Hypertext Transfer Protocol (HTTP) requests. The data is then written to, and stored in, CSV-files. Each row in a CSV-file is data taken from one sampling, which were taken every 30 seconds, meaning that there is approximately 30 seconds difference between each row[**?** , p. 25]NIKOLOPOULOS2022

In the loaded CSV-files, whose names are of the format "yyyy-MM-dd_ChargingPointID" (where "ChargingPointID" corresponds to a series of numbers and letters identifying a unique charging port), the following data is contained: "Timestamp", the time of the sampling of the format "yyyy-MM-dd-HH:mm:ss.SSS; "Charger Status", the status of the Charger (being "Online" for all rows used in this project, as other values means that no EV is connected, and thus appropriately treated as empty rows); the three phases of current measurements in amperes; the three phases of voltage measurements in volts; "Charging Status", with the possible values "Charging" and "Connected" (as the value "None" not in any treated data, as they are not present in data where "Charger Status" is "Online"); and "Charging Session ID", a unique ID given to the charging session, where a charging session is defined as a continuous streak of samples where the "Charging Status" is "Charging" or "Connected", as opposed to "None". Two example rows can be seen in Table 2 and (continuing) in Table 3

Timestamp	**Charger Status**	**Phase 1 Current (A)**	**Phase 2 Current (A)**	**Phase 3 Current (A)**
2024-01-14-00:16:32.779	Online	16.11	16.21	16.157
2024-01-14-00:17:04.139	Online	0.0	0.0	0.0

Table 2: Two succeeding example rows from a CSV-file (Part 1).

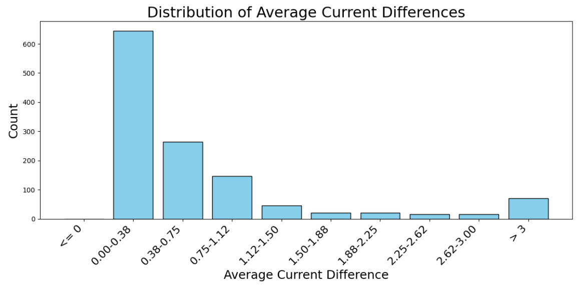
Phase 1 Voltage (V)	**Phase 2 Voltage (V)**	**Phase 3 Voltage (V)**	**Charging Status**	**Charging Session ID**
231.88	230.76	231.72	Charging	59310825
0.0	0.0	0.0	Connected	59310825

Table 3: Two succeeding example rows from a CSV-file (Part 2).

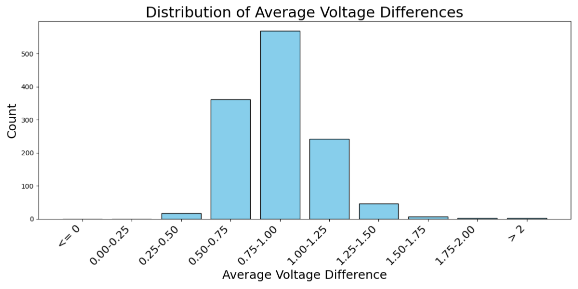
As a part of this project, the CSV-files were loaded into a Visual Studio Code environment (set up to work in tandem with Github), with a Python Script, in which they were all combined to a single CSV-file and then further processed in what can be deemed "preprocessing". First, erroneous and empty data was adjusted or removed. This could, for example, be negative values of Currents or Voltages (that excepting their negativity where reasonable), that were treated by taking their absolute values; files with only zeroes being removed; and filtering away all data from two specific charging ports as their data was known to be wildly erroneous.

To this time series data, the following was calculated and added:

- Charger IDs were converted to numbers between 1 and 60, representing the numbering of the charging ports in Dansmästaren, and named "**ChargingPoint**".
- The "**kW**s of each sample were calculated.
- Assuming 30 seconds of continuous equivalent charge, "**kWh**"s were added.
- "**VoltageDiff**" represents the total sum of distances between the phase voltages per sample.
- "**CurrentDiff**" represents the total sum of distances between the phase currents per sample. The addition of this and "VoltageDiff" rests on the intuition that the relative differences between phases may contain information not easily handled by models looking at the overall currents and voltages, and as Figure 10 shows, there are differences between the phases that can be discerned even when averaged.



(a) Differences in current between phases in amperes.



(b) Differences in voltage between phases in volts.

Figure 10: Distributions of current- and voltage differences between phases.

3.3.4 Preprocessing

The data was then split into two main bodies: the time series data proper, and "metadata" – where each charging session is condensed and only occupies one row, at the sacrifice of the time series data for voltages and currents, in favour of features explaining the charging session as a whole. These features were added as columns to a "meta dataframe" linked by ID to the time series dataframes, and are as follows:

- "**Filename**" is the filename in quotes, extracted from the CSV filename, for reference, if ever needed.

- "**Half_Minutes**" is for how many samples (that is, half minutes) the charging session took place (meaning how long the EV was connected).
- "**Charging_Half_Minutes**" is for how many samples (that is, half minutes) the "Charging Status" was "Charging", meaning how long the EV was charging.
- The "**Floor**" of Dansmästaren where the charging port is located was added.
- A boolean "**Weekend**" was added, representing whether or not the first sample of a Charging Session was received during a weekend (or holiday).
- "**Current Type**" was introduced, indicating either "1-Phase" or "3-Phase", depending on whether just one phase of the Currents and Voltages was ever non-zero.
- "**AverageVoltageDifference**" represents the average difference between the phase voltages, calculated from "VoltageDiff".
- "**AverageCurrentDifference**" represents the average difference between the phase currents, calculated from "CurrentDiff". This and "AverageVoltageDifference" are used as potential features of the meta data, if the "VoltageDiff" and "CurrentDiff" in themselves prove not useful enough and/or cumbersome enough not to use in the time series clustering, but the averages still may. These are the features plotted in 10.
- "**MaxVoltage**" is the highest voltage value recorded during the charging session.
- "**MaxCurrent**" is the highest current value recorded during the charging session.
- "**Energy_Uptake**", the accumulated kWhs of a charging session, calculated from "kWh".
- "**TimeConnected**" represents the timestamp of the first sample of the charging session.
- "**TimeDisconnected**" represents the timestamp of the last sample of the charging session.

A problem with using time in the standard format (in which it is expressed in "TimeConnected" and "TimeDisconnected"), is that points in time that are very similar seem to be very dissimilar to algorithms not accounting for the cyclical nature of the day/night-cycle. For instance, an EV connecting at 23:59 should reasonably not be treated all too differently (with regards to time of day) as an EV connecting at 00:00, although the hours and minutes they add up to if one simply counts the hours and minutes of the day, are considerably dissimilar (after normalization they would be as dissimilar as can be). A common way to deal with this is to encode cyclical features (in this case the time of day) into a pair of sine- and cosine functions, as can be sen in Figure 11[34][35].

In this way, "TimeConnected" and "TimeDisconnected" features were converted to two features each, which were also added as features to the data: "**TimeConnected_sin**", "**TimeConnected_cos**", "**TimeDisconnected_sin**", and "**TimeDisconnected_cos**".

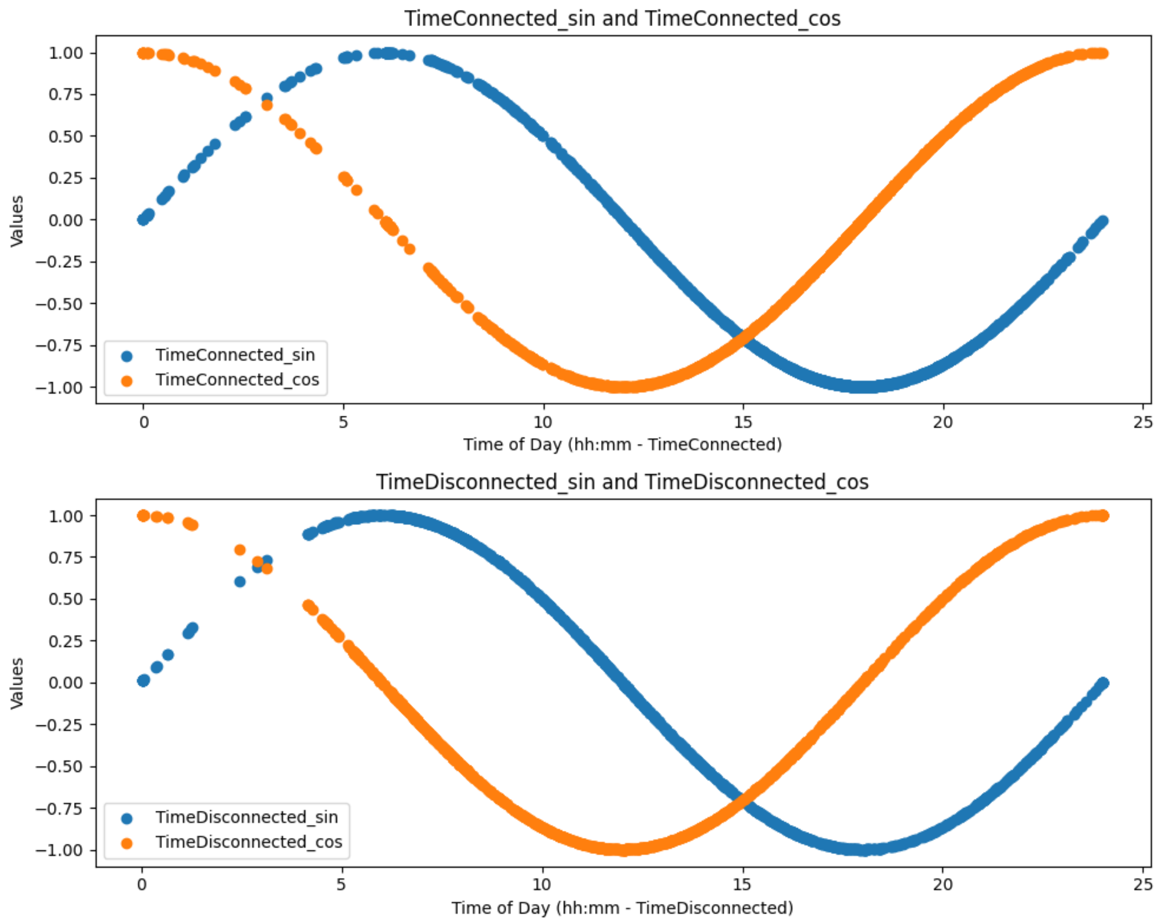
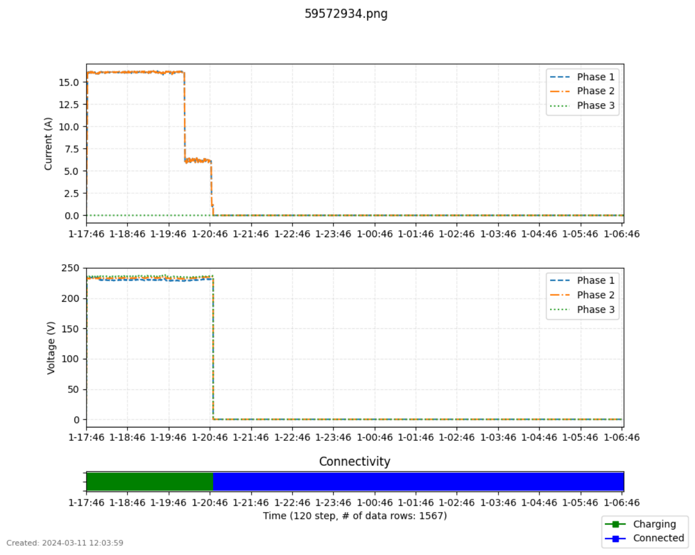


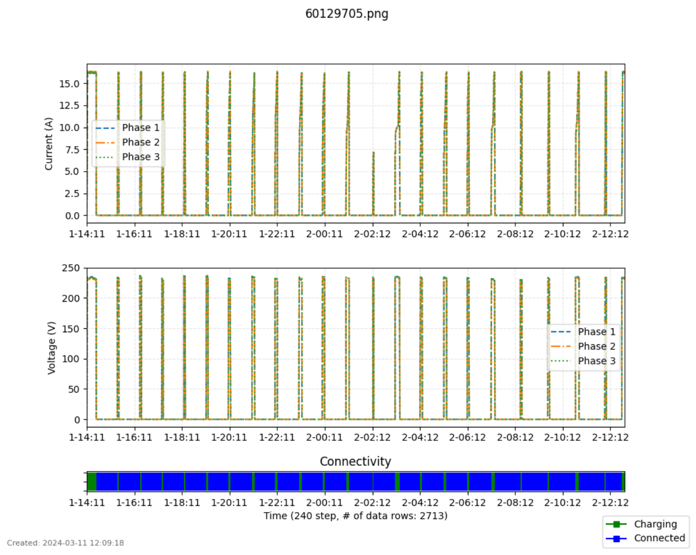
Figure 11: Cyclical time encoding using the actual times encoded from EVs at Dansmästaren. Note the scarcity of points in the hours following midnight.

3.3.5 Full Charge

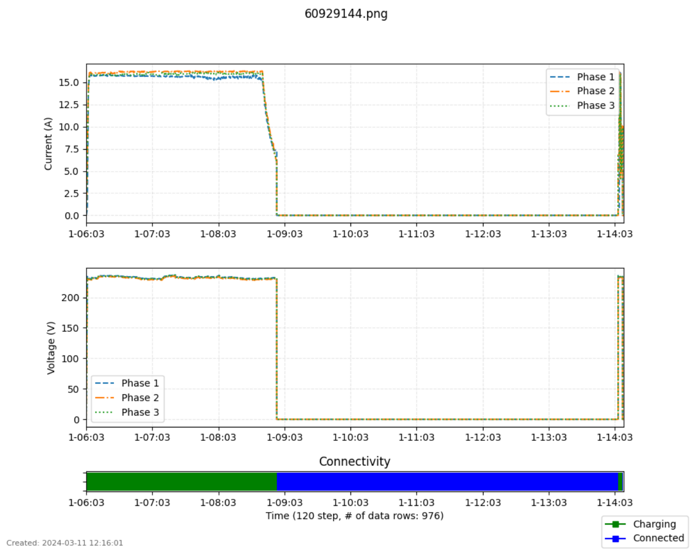
Furthermore, whether or not the EV of a charging session is considered to be fully charged was added to the boolean"**FullyCharged**". It shall here be noted that the State-Of-Charge (SOC) of the EVs is not available, and must therefore be surmised by whether or not the EVs are drawing current. Some cases are unambiguously fully charged, as determined that the EV has simply stopped charging (see Figure 12a). Determining the "FullyCharged" value was not done by simple using the last charging status of the charging sessions however, as some fully charged EVs charge for a single (or very few) sample(s) at regular intervals after the main actual charging has been done (see Figure 12b). Additionally, some EVs seem to charge during the last sample or two, after not charging for a single sample since the "main" real charging (see Figure 12c, Figure 12d). This might be some (very insignificant) charging that is done after the EV is unlocked remotely, for example. This was dealt with by defining a "Charging Streak" as the number of samples that continuously have the charging status "Charging", and then calculating if the number of samples in the last Charging Streak of the charging session constitutes a significant portion ("Streak Percentage") of the total number of samples with the charging status "Charging" for that specific charging session. This "Streak Percentage" was set to 20%, determined by visual inspection of a number of charging sessions. An example of this can be seen in Figure 13



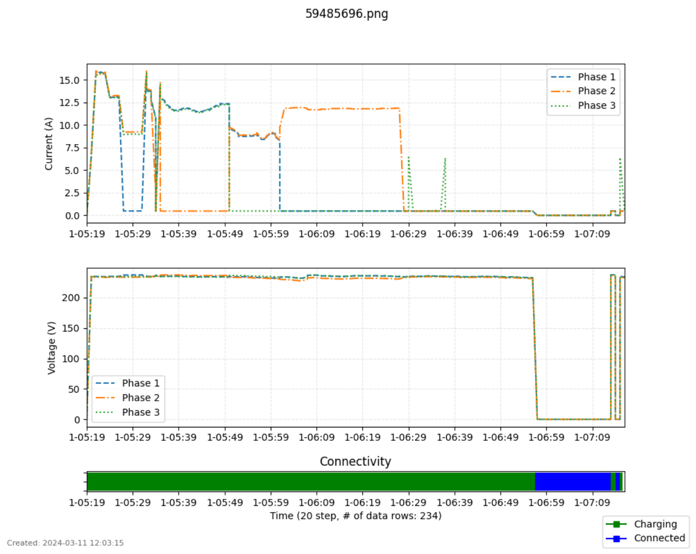
(a) Time series 1.



(b) Time series 2.

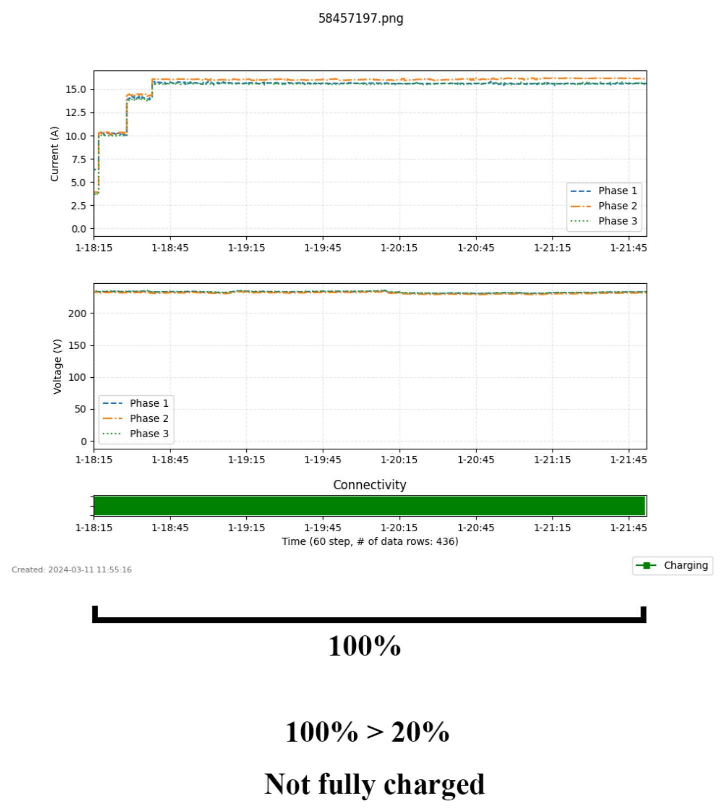


(c) Time series 3.



(d) Time series 4.

Figure 12: Four examples of different time series with different behavior.



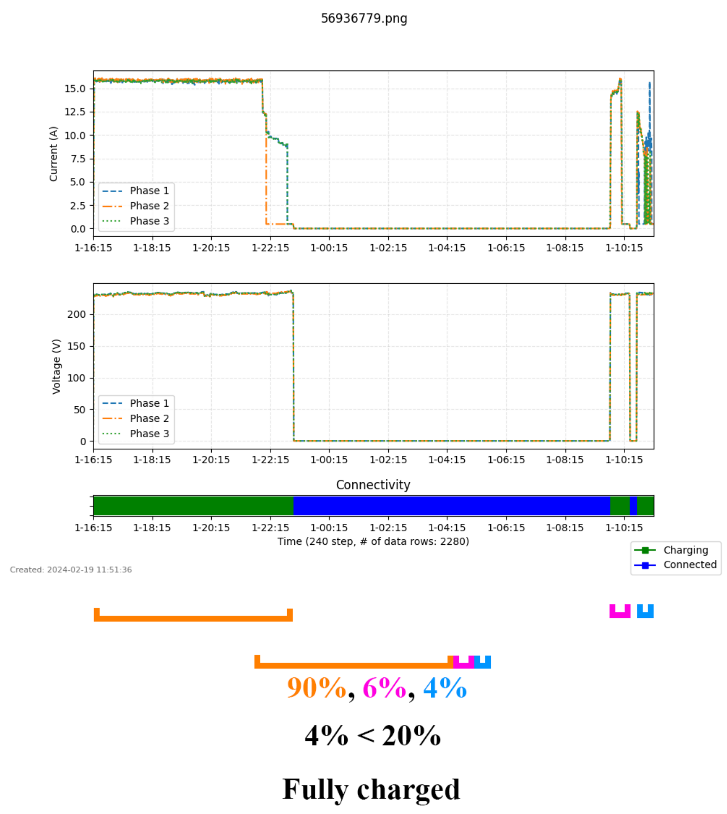


Figure 13: Two charging sessions, one deemed fully charged and one not (percentages visually approximated).

3.4 Settings

3.4.1 Deep Learning Settings

The exploration was done using grid search on a range of hyperparameters. After min-max normalizing all numerical features, the hyperparameters tested were as follows:

Hyperparameter	Values tested
Epochs	10, 25, 100, 150, 200, 250, 500, 1500
Batch size	1, 4, 8, 12, 16, 32
Layer 1 neurons	64, 128, 192, 256, 320
Layer 2 neurons	48, 64, 96, 128, 256
Layer 3 neurons	1, 4, 16, 32, 64
Layer 1 activation	tanh, relu, sigmoid, softmax
Layer 2 activation	tanh, relu, sigmoid, softmax
Layer 3 activation	tanh, relu, sigmoid, softmax
Dropout rate	0.3, 0.4, 0.5
Learning rate	0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1

Furthermore,

- a hyperparameter "ExcludedFeature", excluding one feature from the deep learning model was attempted for every feature.
- A hyperparameter "ShouldEmbed" determined whether or not an embedding layer was to be used.
- The model was tested with and without "one-hot encoding" of the boolean feature.

The number of neurons in layer 3 being 1 essentially means that we tested a layer depth of 2 and 3, and decided on 2, as 1 neuron in layer 3 is virtually equivalent to there not being a layer there. Furthermore, the dropout layer was placed after the first dense layer, and an output layer with linear activation was set. The optimizer used was "Adam" with the mean squared error as loss function.

3.4.2 Time Series Clustering Settings

Similarly to the deep learning model tuning, the hyperparameters of the time series clustering were chosen using grid search. The values chosen were chosen because they either (a) performed best, or (b) performed equally well as other values that were more time-consuming to compute with. "Performing best" here means yielding the lowest RMSE for the deep learning model. The hyperparameters were tested as follows:

Hyperparameter	Values tested
Algorithm	TSKmeans, KernelKmeans, K-shape
Max Iterations	50, 100, 250
Tolerance	1e-08, 1e-07, 5e-07, 7e-07, 1e-06
Number of initiations	1, 3, 5
Metric	Euclidean, DTW, softDTW
Max barycenter iterations	50, 100, 200
Use voltages	True, False
Use all 3 phases	True, False
Min cluster size	None, 5, 10, 25
Max cluster size	None, 100, 250, 500
Handle min clusters	Reassign, Merge, Outlier, Nothing
Handle max clusters	Reassign, Split, Nothing

The time series clustering algorithms outlined in Section 1.2 were variably tested with the distance metrics also outlined, in addition to Euclidean distance. Different values of the maximum amount of barycenter calculations was tested only for TSKmeans using DTW or softDTW, because that is where the setting is available in tslearn. Similarly, the different options for distance metrics were only tested for TSKmeans. The number of initiations corresponds to the setting "n_init" in tslearn, controlling the amount times the algorithms will be run with different centroid seeds, although this did not affect performance (other than taking longer). The tolerance corresponds to the setting "tol" in tslearn, being the inertia tolerance threshold determining when the algorithm is considered to have converged.

Additionally, additional hyperparameters were constructed. "Use voltages" controls whether or not the voltage data from the time series were used in the time series clustering. The rationale for testing without is that voltages are generally affected by factors external from the EVs themselves, such as the load of the grid. Including the voltage data proved more accurate, however. Only using one phase (phase 1) was also tried, to see if the same performance as using all three phases could be achieved using only on phase (requiring less computation time), although this was not the case.

3.4.3 Sizes of clusters

Often, out of hundreds of time series, some clusters could have less than 25 (and sometimes even less than 10) time series belonging to them. All the while other clusters contained more than 30% of all time series, even with the number of clusters exceeding 15. This prompted investigation into whether or not handling these clusters by splitting the large ones and merging the smaller ones could possibly improve performance. First, minimum and maximum cluster sizes were set preliminarily, then for clusters not reaching the minimum cluster size the following approaches were tried:

- **Reassign** – All time series in the cluster were assigned to other clusters individually.
- **Merge** – All clusters below the minimum threshold were merged to the nearest cluster, until the threshold was reached for all clusters.
- **Outlier** – All clusters below the minimum threshold were merged to one cluster (the outlier cluster designated "-1"), regardless of similarity to each other.
- **Nothing** – No action was taken.

For clusters exceeding the maximum cluster size, the following approaches were tried:

- **Reassign** – The process of the time series most distant from the cluster barycenter being assigned to the other nearest cluster was tried until the cluster was below the maximum cluster size threshold.
- **Split** – The cluster was split into two. This was repeated if any resulting cluster was still above the maximum cluster size threshold.
- **Nothing** – No action was taken.

Then, different minimum and maximum cluster sizes were tested, in terms of absolute members and not as percentages. Meaning any clusters with less time series than the minimum or more than the maximum were handled in one of their respective ways outlined above. In the end, simply not handling clusters below the minimum or above the maximum, regardless of the minimum- and maximum thresholds set, and the handling method, provided the best results. For all time series clustering the random state "random_state" was set to 42 for replicability, since the time series clustering is meant to be reproducible, as opposed to the deep learning which only act like a dummy for a future model.

4 Results

4.1 Deep Learning Settings

Of the settings discussed in section 3.4, the best performing ones (meaning the ones yielding the lowest RMSE) were chosen for the final results, as they performed the best, or equally well as other settings that were more time-consuming. The deep learning model settings were finally set to:

Hyperparameter	Value chosen
Epochs	150
Batch size	16
Layer 1 neurons	256
Layer 2 neurons	64
Layer 3 neurons	1
Layer 1 activation	tanh
Layer 2 activation	relu
Layer 3 activation	relu
Dropout rate	0.4
Learning rate	0.005

Furthermore, the deep learning model performed best with

- "ExcludedFeature" set to "None" (meaning that the model performed best when not excluding any feature),
- "ShouldEmbed" set to false,
- "one-hot-encoding" not being explicitly set as the same performance was gained without it (possibly due to built-in one-hot-encoding of boolean features)

Sometimes the deep learning model would perform poorly, presumably being stuck in local minima, as there were two clear groups of RMSE results: those below 325 and those above 450. Only around 500 runs out of over 33000 had a RMSE for the clusters model between 325 and 450, with the rest being either above or below this range. This was partly dealt with by running the same deep learning settings multiple times with different values of "random_state". This is a setting that controls the random number generator used to shuffle the data before splitting it. When splitting, the size of the test set was always set to 0.3, meaning 30%. The model using cluster data was less likely to get stuck in this local minima compared to the intermediate model, regardless of other settings such as learning rate (though the best learning rate made either model less likely to get stuck). Even when not stuck in such a sub-optimal local minima as > 450, the performance using different random states varied. As such, multiple runs with the same data and same settings were run multiple times with different random states for better results.

4.2 Time Series Clustering Settings

The time series clustering settings were finally set to:

Hyperparameter	Value chosen
Algorithm	TSKmeans
Max Iterations	250
Tolerance	7e-07
Number of initiations	1
Metric	softDTW
Max barycenter iterations	100
Use voltages	True
Use all 3 phases	True
Min cluster size	None
Max cluster size	None
Handle min clusters	Nothing
Handle max clusters	Nothing

Time series clustering without voltages at all was tested, as well as testing with only one phase (for voltage and current, as well as only current). Both of these approaches were quickly abandoned as the performance with them was significantly worse.

4.3 Data Visualization

As stated in section 3.2, time series clustering is performed for time series of length $t = 15, t = 30, t = 45$, and $t = 60$ minutes, which at the sample frequency $\omega = 0.5$ samples/minute equals $n = 30, n = 60, n = 90$, and $n = 120$ samples. This is done by extracting the relevant time series data from the preprocessed time series, meaning that all time series of length $l < n$ are discarded. Then the first n samples from the remaining time-series are extracted. These extracted time series is what the time series clustering is performed on. This means that all extracted time series are of the same length, for every chosen value of n. The duration of how long the EVs are connected and how long they are charging is shown in Figure 14.

Prior to the actual time series clustering, a slight exploration of the data with respect to EV behavior was conducted, in no small part by visualization. Partly to get an idea of what the charging sessions look like from a larger perspective, and to serve as a sort of sanity-check, so as to see that preliminary results were reasonable and not caused by an error of method and/or application. This exploration and visualization could also validate, in earlier stages of research, intuitions on different behaviors of EV charging. One example of this is the fact that almost all EVs that are connected longer than a certain period are considered fully charged, as can be seen in Figure 15 and Figure 16, with the few not deemed so are insignificant in number, and probably resulting from faulty battery behavior and/or incorrect measurements. Exempting these, more detail on the types of EV charging amongst those charging for a

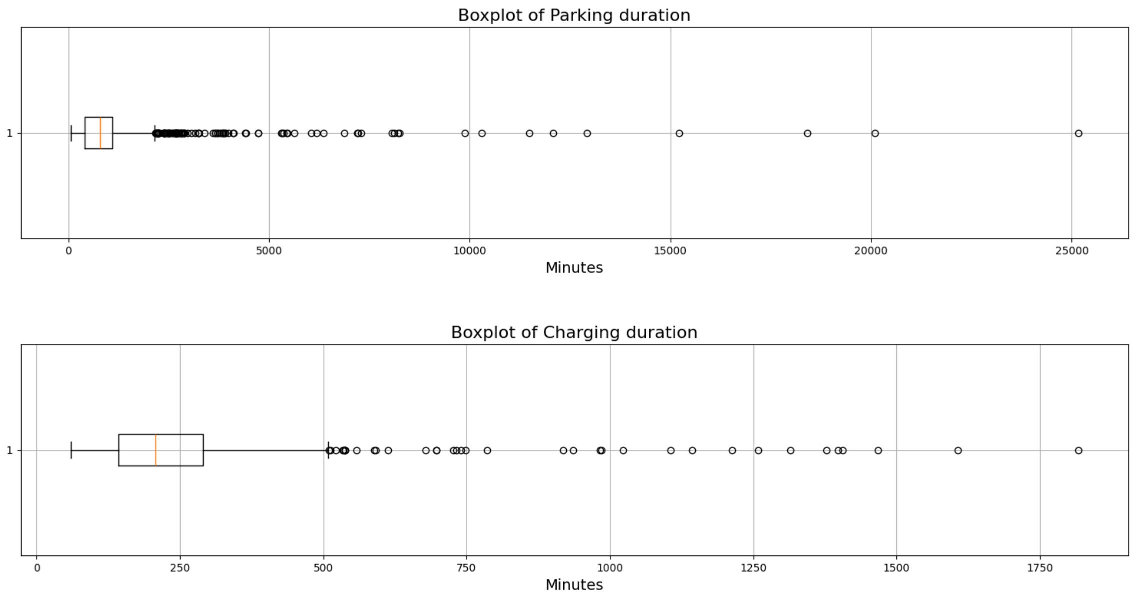


Figure 14: Boxplots of parking- and charging durations.

maximum of 24 hours can be seen in Figure 17. The spike between 0 and 30 minutes could be people parking briefly to buy groceries in the shop below, for example.

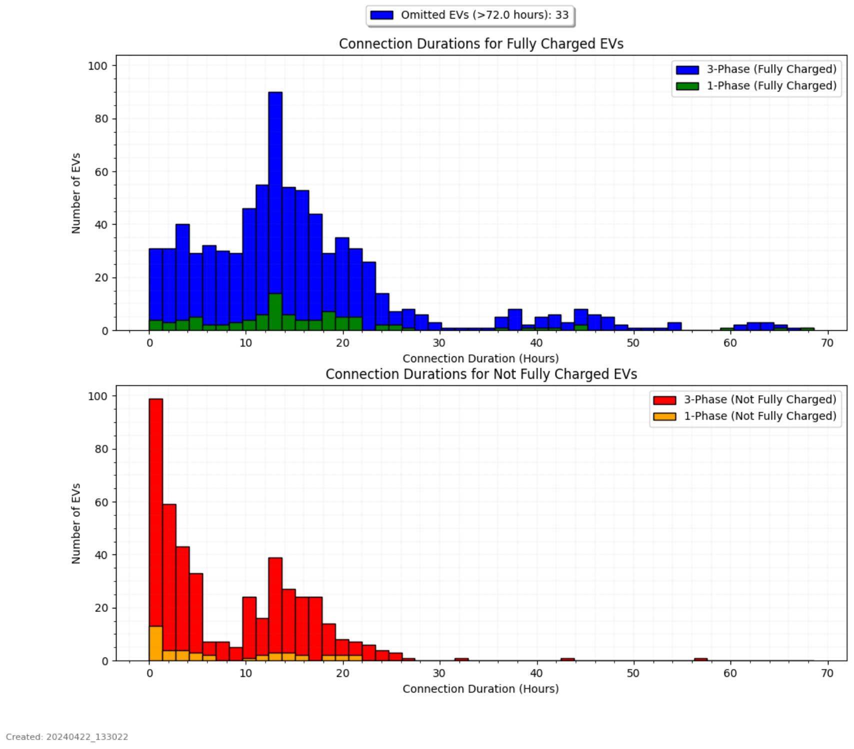


Figure 15: Different phase charging separated (first 72 hours).

Another informational visualization regards the hour of the day that EVs connect and

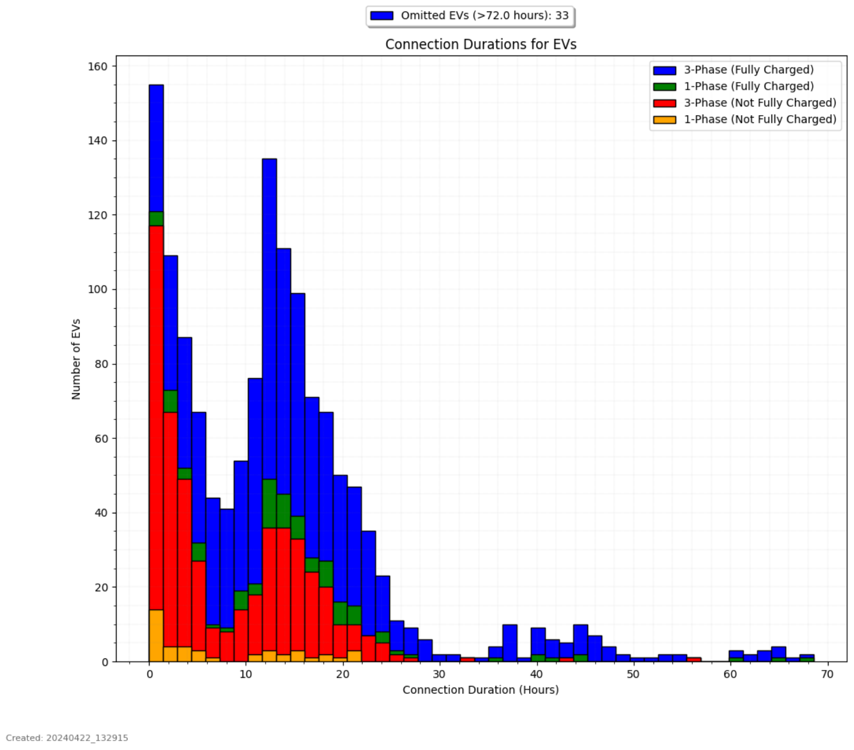


Figure 16: Different phase charging joined (first 72 hours).

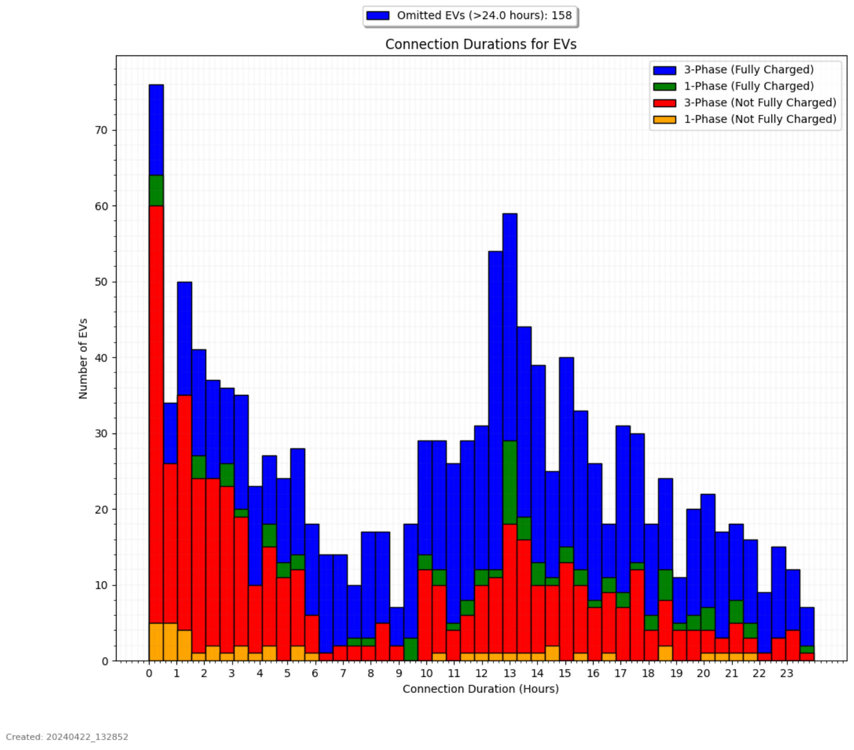


Figure 17: Different phase charging joined (first 24 hours).

disconnect, as can be seen in Figure 18. Here the x-axis is the hours of the day that an EV was either connected or disconnected. The height of the columns show the median charging duration of EVs that connected or disconnected during that hour. Finally, the alpha value (darkness) of the columns depend on how many EVs connected during this hour. The left subplots show weekday data and the right subplots show weekend data, and the top subplots show the time of EV connection and the bottom subplots show the time of EV disconnection. A single EV can be present in multiple subplots, by for example connecting on 18 on a Friday (contributing to the upper left subplot) and disconnecting a Saturday at 10 (contributing to the bottom right subplot). Note that the data present in the visualization is not for one period of 24-hours, but for all data collected.

Some basic behaviors can be seen here. The spikes in connections around hours 16 to 18 and disconnections around hours 06 to 07 are entirely absent from the visualization of weekend data, further suggesting that the spikes result from people coming home and going to work. The charging sessions last for longer during the weekends, and are more more uniformly distributed, though not as many EVs are connected. Moreover, the median charging duration in hours increases sharply at 14 for connecting weekday cars, suggesting that people start returning home from work. A similar change, though a drop, can be sen in the weekday disconnection subplot, where the median charging duration drops sharply after 17, suggesting that the visits disconnecting after this hour are only stopping by for shorter visits, be it for buying groceries at the shop below or visiting friends and family.

Since Dansmästaren contains 5 different floors, it can also be informational to visualize each floor separately, as they may show different behaviors. Not in the least because floors 1 and 2 are public, whereas floors 3, 4, and 5 are rented (long term-parking for residents). It is expected, then, that different behaviors are exhibited by EV owners depending on floor. As can be seen in Figure 19 and Figure 20, the spikes in EV connections and disconnections around the normal workday that were present in Figure 18 are almost absent.

As can be seen in Figure 21 and Figure 22, floors 3 and 4 show behavior very similar to that of Dansmästaren taken as a whole, albeit with even stronger tendencies. As can be seen in Figure 23, floor 5 shows the same behavior, but with weaker tendencies, perhaps due to it being the furthest away from the entrance (at the bottom floor) of them all.

4.4 Time Series Clustering

Since every time series belongs to one and only one cluster, all time series can be plotted to a graph showing all the time series assigned to that cluster. An example clustering can be seen in Figure 24 using $n = 120$. For easier visualization, only one phase per charging session is shown in the plot. The results presented in this paper deal only with the best performing run of every unique combination of samples $n = 30$, $n = 60$, $n = 90$, $n = 120$, and the number of

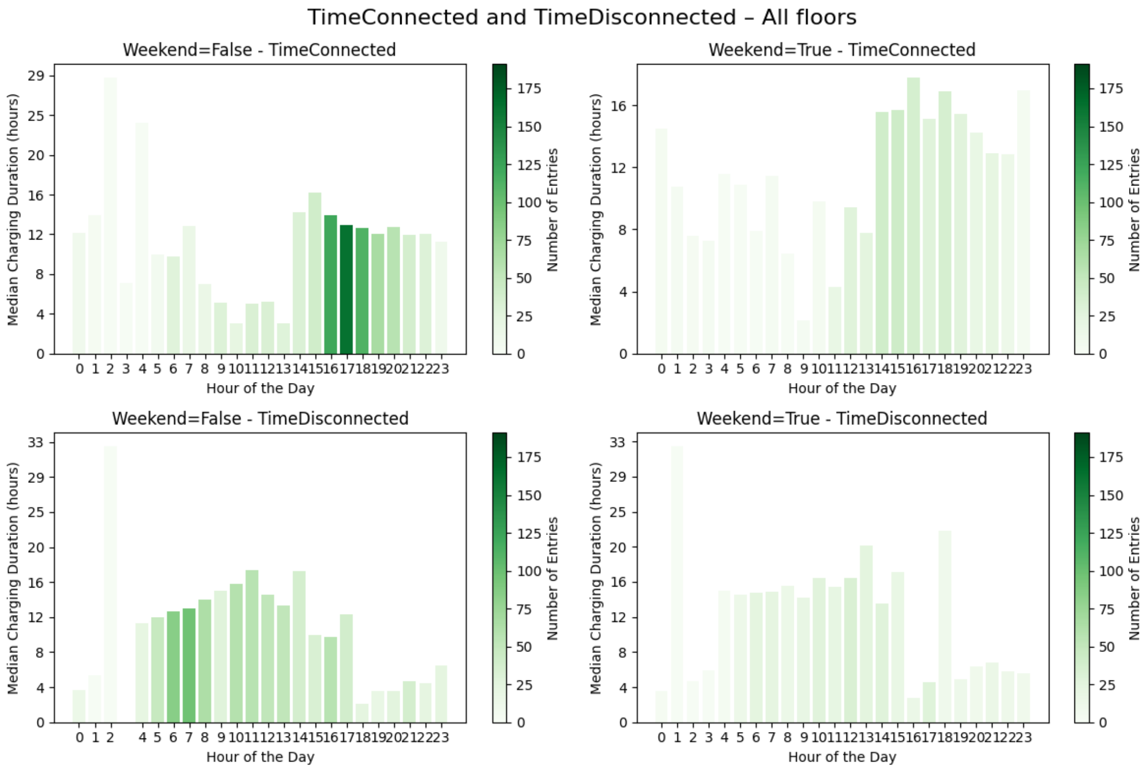


Figure 18: The times that EVs connected and disconnected at Dansmästaren.

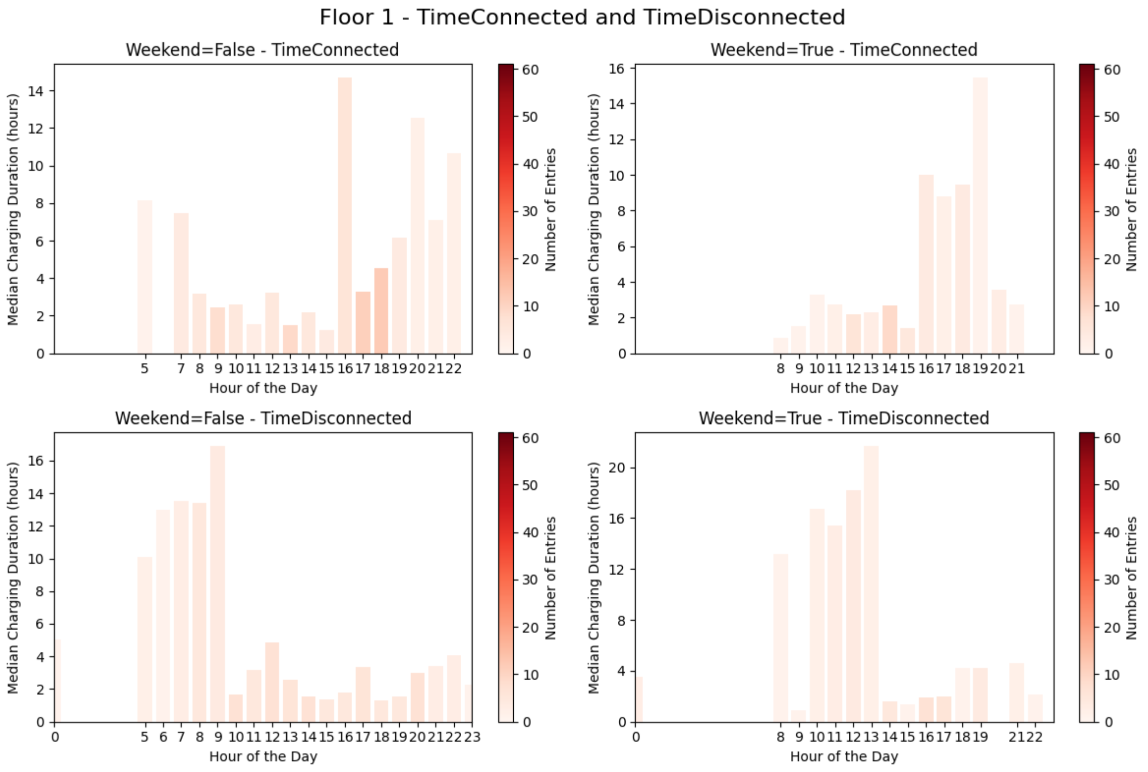


Figure 19: The times that EVs connected and disconnected at floor 1 of Dansmästaren.

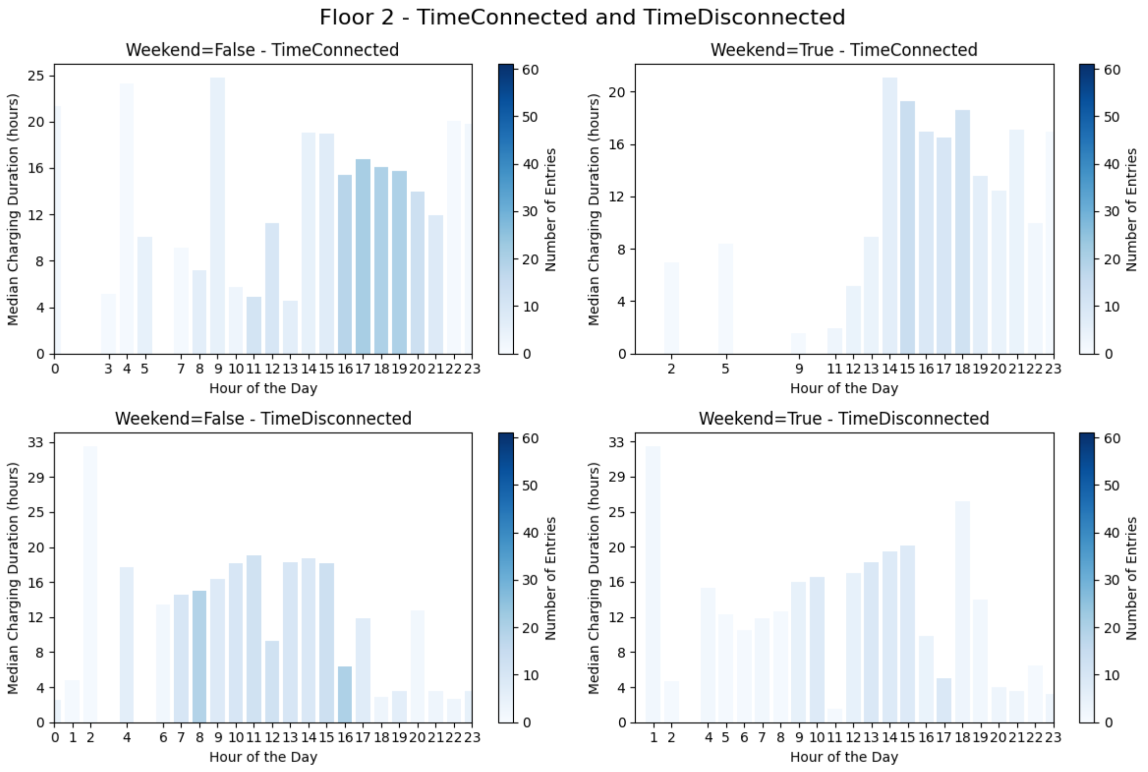


Figure 20: The times that EVs connected and disconnected at floor 2 of Dansmästaren.

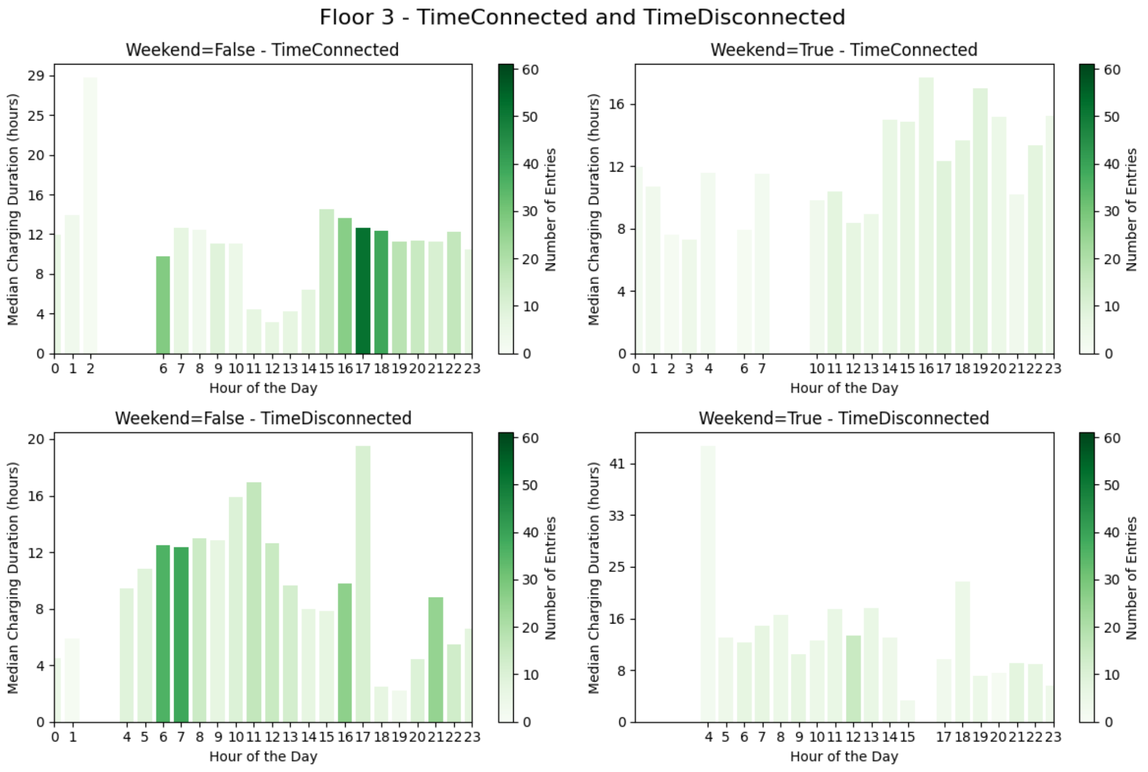


Figure 21: The times that EVs connected and disconnected at floor 3 of Dansmästaren.

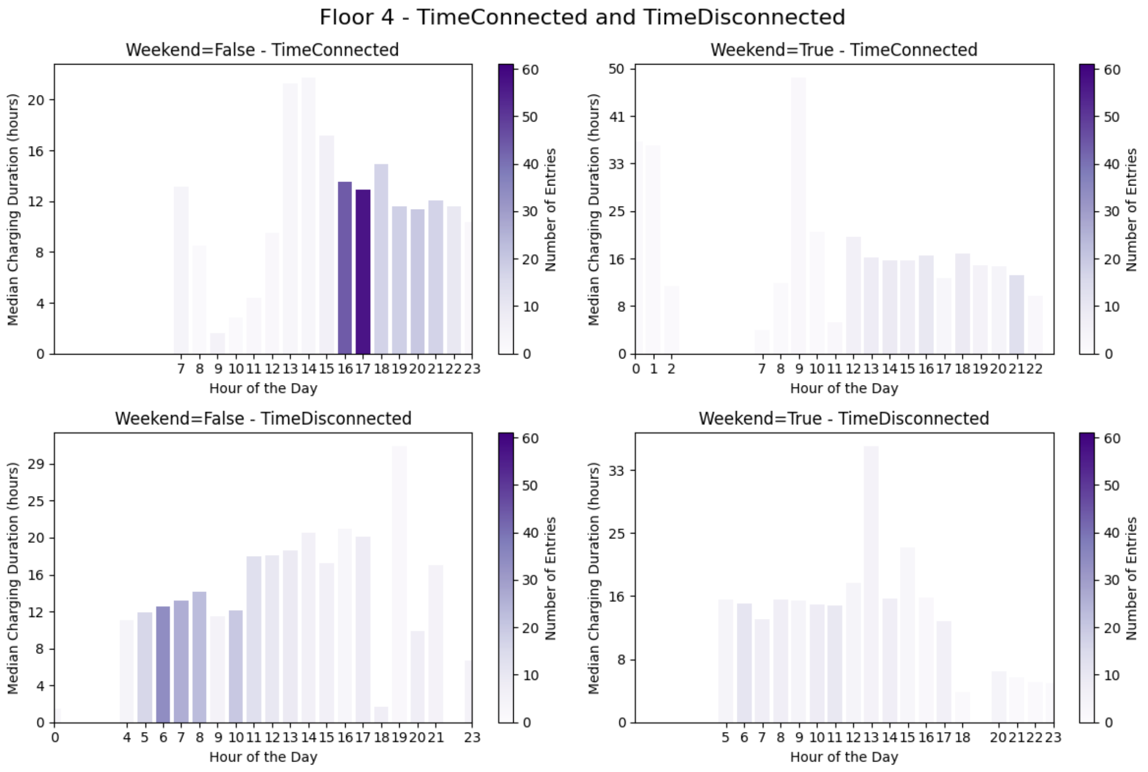


Figure 22: The times that EVs connected and disconnected at floor 4 of Dansmästaren.

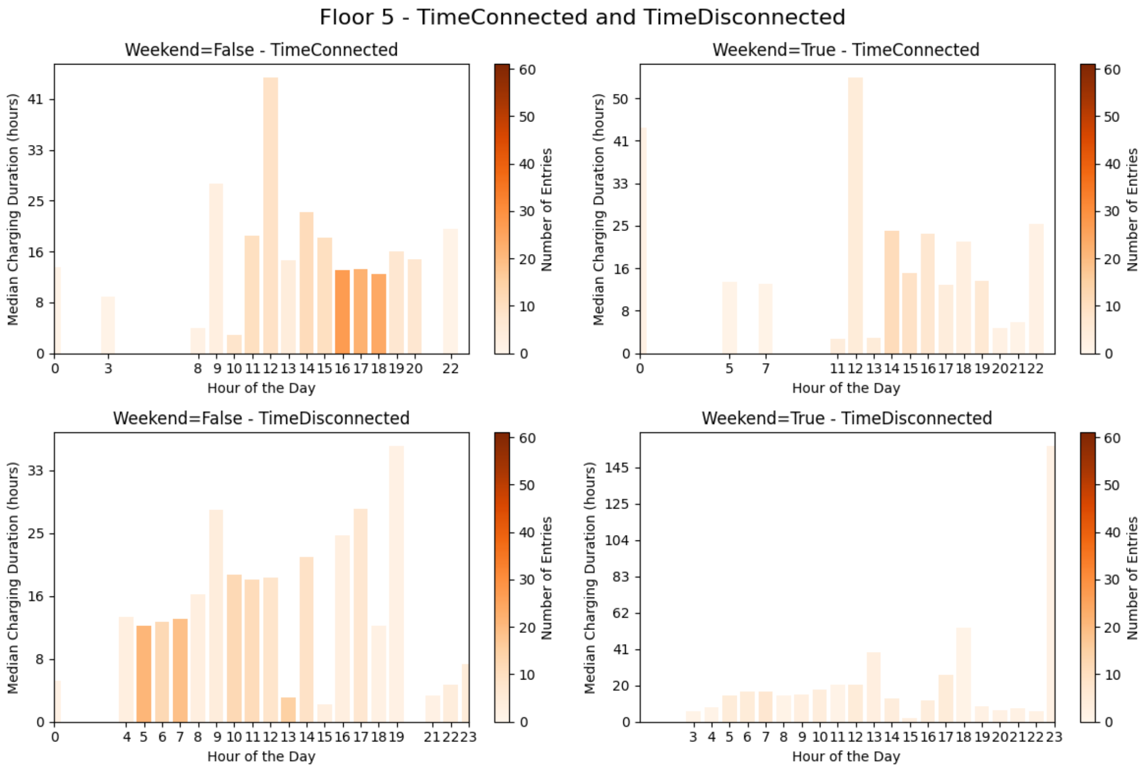


Figure 23: The times that EVs connected and disconnected at floor 5 of Dansmästaren.

clusters, ranging from 2 to 20; unless otherwise specified.

4.5 Predictions

For the intermediate model, the best results achieved overall are displayed. For the Clusters models, the best results achieved for that value of "TS_Samples" (i.e. n) are displayed, for the value of "num_clusters" that performed the best.

Model	RMSE (minutes)	RMSE decrease	MAE (minutes)	MAE decrease	Clusters
Intermediate	90	0	50	0	0
$n = 30$	82	8%	46	7%	14
$n = 60$	88	2%	47	8%	12
$n = 90$	76	15%	45	11%	17
$n = 120$	74	18%	42	17%	13

Every set of setting combinations for the deep learning and time series clustering predicted every relevant time series (meaning at least of length n, see 4.3). This enables prediction data of every session to be compared with the "real" data, meaning the actual charging times for the EVs the time series belong to. This is done for both the model only using the intermediate data, and for the model also using the cluster data. In other terms, this shows when the model predicts a time series to end given its circumstances, versus its circumstances in addition to its appearance. As can be seen in Figure 25, which sorts the charging sessions by duration on the x-axis, the "Clusters" model performs better than the "Intermediate" model, though both overestimate the durations of the shortest charging sessions, and underestimate the durations of the longest charging sessions.

For the four values of n tested, different RMSE values ("true" value versus predicted value) were achieved, see Figure 26. Overall, the different cluster models performed quite similarly with respect to this metric. Around 7-9 clusters, as well as around 15-17 clusters, most models seem to perform best.

Another way to show the model predictions is to plot the MAE of the different cluster models, as in Figure 27. Here the relative dips around 7-9 clusters is yet more clearly visible.

A succinct way to display the results is to show by how many percent the error decreased when using the clusters model compared to the intermediate model, as can be seen in Figure 28. The decrease in error varies depending on the number of clusters, though it shall be noted that since the best performing amount of clusters is best chosen for future models, the increase for the best performing amount of clusters is the one that should be considered.

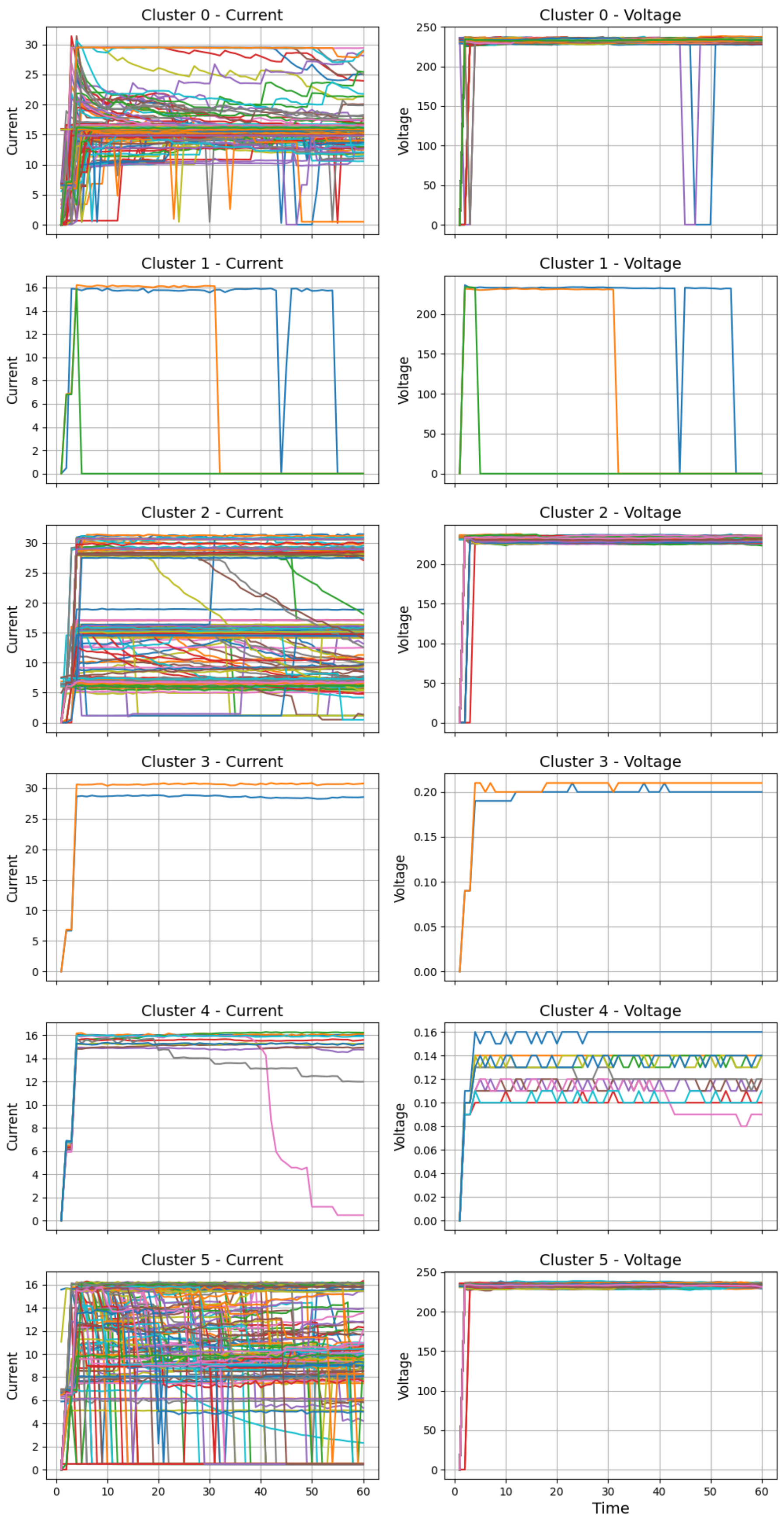


Figure 24: An example of time series belonging to different clusters (Phase 1).

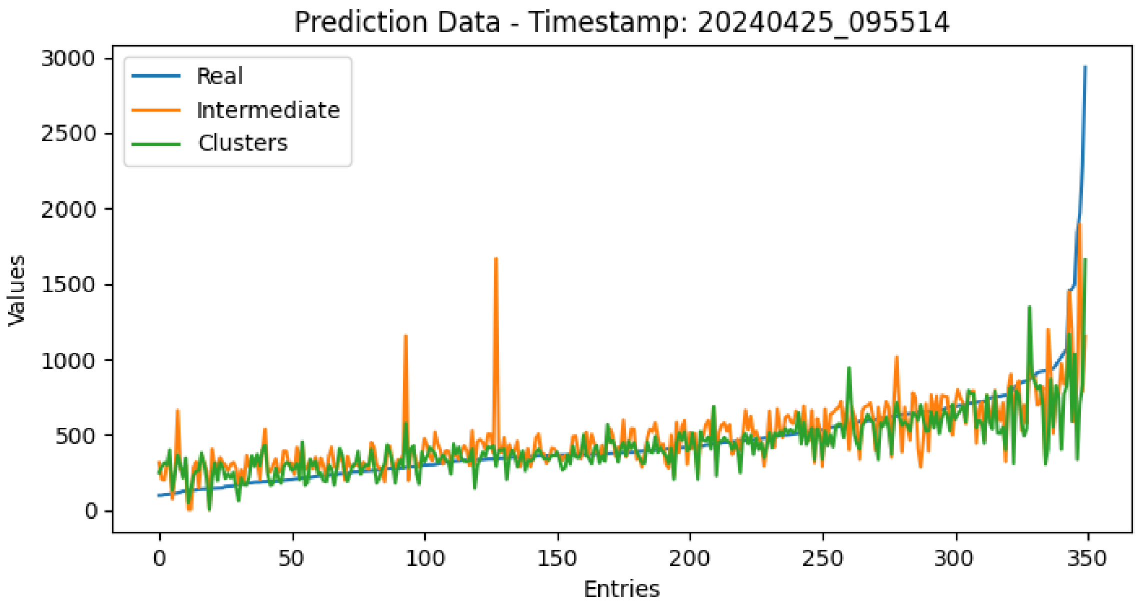


Figure 25: Predictions for charging half-minutes (y-axis) made with and without clustering data.

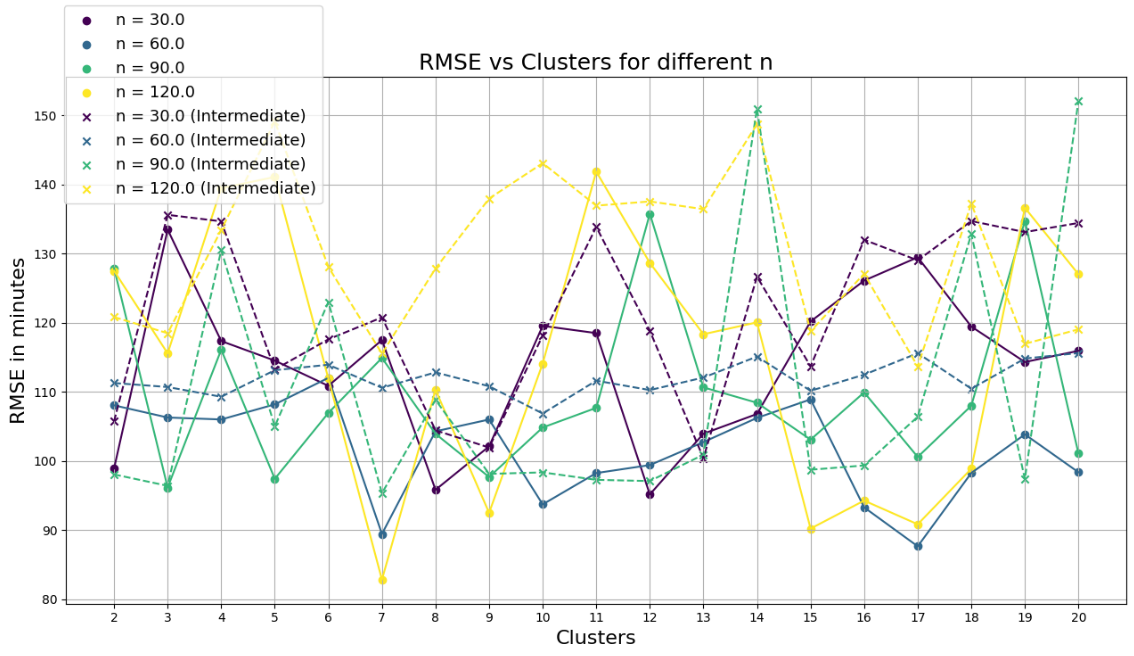


Figure 26: The RMSE of the prediction model. Some runs of the deep learning get stuck at local minima, which is rarer when using the cluster data.

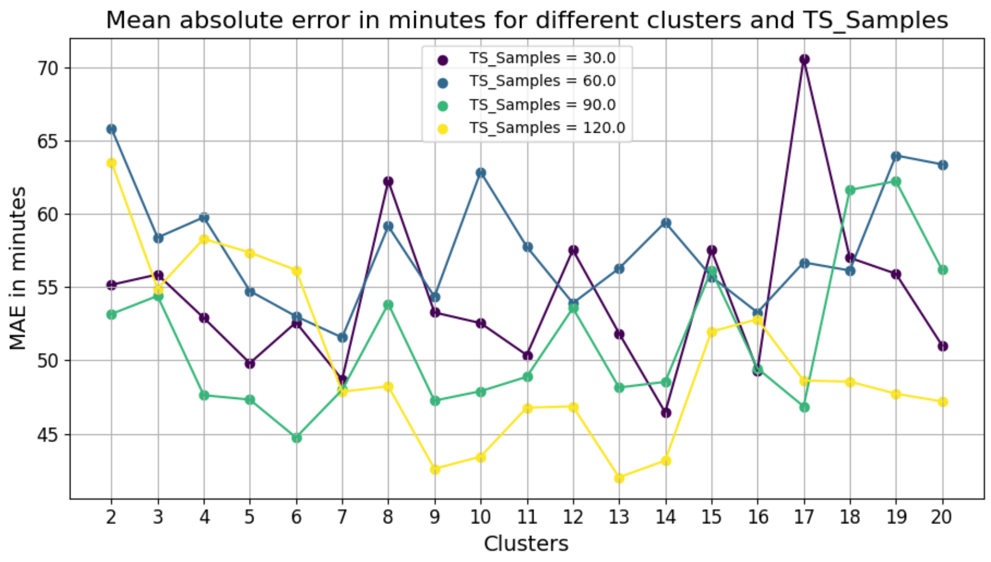


Figure 27: The MAE of the prediction model in minutes.

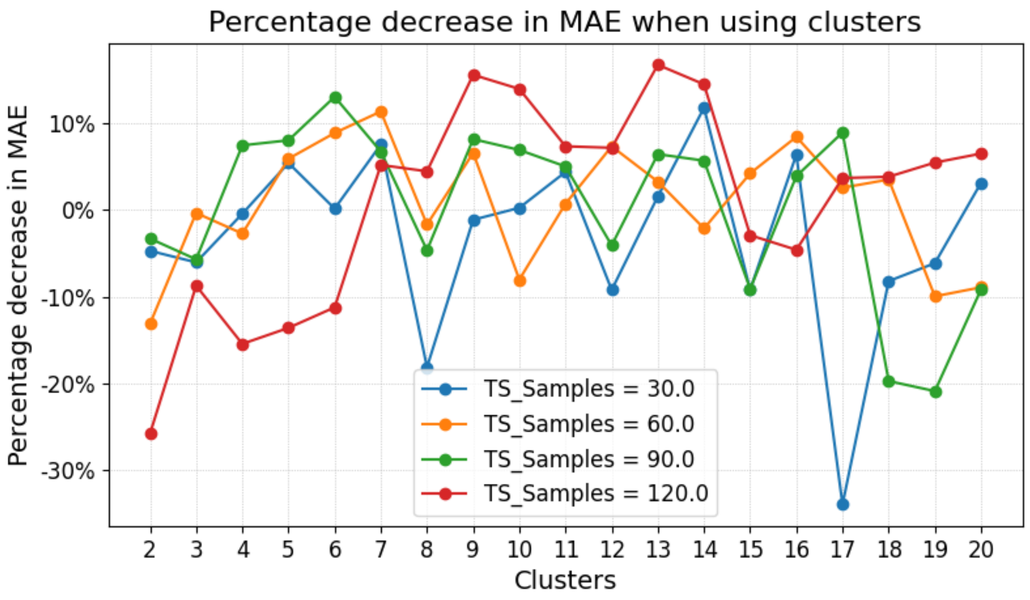


Figure 28: Decrease in error in percent. Negative values mean an increased error.

5 Discussion

5.1 Evaluation and Comparison

It was the aim of the work presented in this paper to improve future prediction models predict the durations that EVs were parked or charging. The one that was performed was on charging, though the same approach and codebase can be used to predict parking duration. There was no stated limit to what would constitute an improvement, not to mention what would constitute a significant one. The metric used in this paper for the sake of minimization was RMSE, though since MAE is a more intuitive and easier to place in real-life contexts as changes in MAE are linear, it shall be the metric we use when discussing the results. In other words, RMSE is more suited for adjusting the algorithms, whereas MAE is more easily placed into a real-life context.

The improvement made when using the clusters depends on the value of n, meaning the amount of samples from the time series that were used for clustering. This translates to being the amount of samples in to a charging session when the deep learning model is to be predicting the charging duration. In other words, if an EV has been connected for 60 samples (= 30 minutes), then the deep learning model has been predicting from that point on, as a real-time prediction model would. Although the equivalent RMSE of the intermediate model depends only on the set random state, the clusters model depends on the number of clusters as well, since the clusters feature affects how the deep learning model behaves. The cluster model also depends on the random state set during the deep learning. It did not, however, also depend on a random state set during the time series clustering, because all time series clustering was done with the random state set to 42 for replicability.

The improvement acquired, in terms of decrease in error, was around 20% for the $n = 30$ and $n = 60$ models, while it was around 30% for the $n = 90$ and $n = 120$ models. The reasons for the better relative performance of the two latter models is not immediately obvious. It could be simply because fewer charging sessions are included, and the time series clustering need not extend to include more behaviors that have been excluded in the preprocessing stage for being too short.

For some number of clusters, the cluster model actually performed worse than the intermediate model. The reasons for this are not clear, suffice to say that the clusters created do not seem to correspond to any difference in charging duration, rather they seem to confuse the deep learning model. Although it is worth noting here again that what matters is the result for the best performing amount of clusters for the relevant n, since that is what future models will use.

Using other data for the time series clustering and deep learning model could affect the optimal number of clusters used. Additionally, some adjustments to hyperparameters might as well, since they also affect the results differently depending on the data used for them. Overfitting has been reduced in this paper due thanks to the 30-70 split of test- and training data, but the predictions could still be heavily dependent on the specific behavior of EVs at Dansmästaren. This is a problem that will not go away by simply using more data from Dansmästaren, since that data will have the same bias too.

To explicitly compare the performance of the clusters model against the intermediate model, we can see the data in Table 4.5. Here we see that for the bet number of clusters, the clusters model performed between 7% to 17% better (with regards to MAE). A general trend is that the more samples that are used for the time series clustering, the bigger the improvement in accuracy of the deep learning model. This can be explained by the intuition that the time series clustering model has more data and therefore behaviors to distinguish. The lower MAE compared to RMSE is explained by the fact that outliers affect RMSE to a much higher extent, since errors are squared. We can see in Figure 25 that the shortest charging sessions are off by about 100 minutes from the predictions, while the longest are off by up to 1000 minutes. These highest values, however, are unlikely to be caused by EVs functioning normally, as it would mean charging for over 48 hours. The fact that the clustering model manages to predict significantly longer charging times for these few cases, as opposed to the intermediate model also doing it incorrectly, is alone grounds for showing that quirks in the charging sessions can aid in detecting these outliers to some extent without giving wild false positives.

As TSKmeans is capable of finding patterns that are vertically and horizontally shifted, we can see in Figure 24 that some patterns are matched that are not in the same space of the plot, for instance the sharp declines in current in cluster 5 that occur during various times of the charging sessions. Visualized are time series of length $n = 120$. The clusters model using $n = 60$ would not "see" the sharp declines in the second half of the plot, hence possibly missing out on similar behaviors from EVs that could behave, and therefore charge, similarly. This is one explanation for higher values of n performing better. TSKmeans can also detect similarities between phases of the charging, meaning for example that phase 1 in one charging session can resemble phase 3 in another. This is why some lines in Figure 24 seem to not align, as only one phase is visualized per charging session.

5.2 Limitations

One approach to deal with the large amount of data needed to process with time series clustering is to reduce the dimensionality of the data. This was not done in this project, as there were not too many points to process, given that the maximum amount of samples used for any time series was 120, and that each time series had at most 8 features. Of these 8 time features in the time series data –the voltages of the three phases, the currents of the three phases, plus "**VoltageDiff**" and "**CurrentDiff**– only the original current and voltage data was used in the final clustering, further reducing the need for dimensionality reduction. The reason for the eventual exclusion of "**VoltageDiff**" and "**CurrentDiff** is that they did not improve the model performance, while at the same time making the time series clustering take even longer to complete. Additionally, as outlined in Section 3.2, this time-consuming time clustering is not meant to be done on-the-fly, but rather as a foundation for a future real-time classification to work on. So to generate new clusters may be time-consuming, but easy, since dimensionality reduction has not been performed.

Since optimizing the deep learning model was not the aim of this paper, the different designs and settings were only briefly explored. The increase in performance when using the best performing clusters show that the time series clustering has indeed fulfilled its job of assisting the deep learning model in more accurately predicting EV charging duration. An increase of 7-17% is significant here because it (a) is not likely caused by smaller, random variations of the deep learning model, but rather a consistent out-performance; (b) substantial with regards to the actual accuracy gain in minutes (from a 50 to 42 minute error for the best models), increasing the usefulness of future predictive models, potentially helping them be used in real-world scenarios. Future models are likely to perform better, given that the deep learning in this paper was not deeply explored. The accuracy gain achieved could still remain, however, possibly enabling such models to be viable.

5.3 Future Work

Given the limitations of the project time frame, more clusters than 20 was not tested, despite results with 18 to 20 clusters not being significantly subpar compared to when using fewer clusters. This was motivated by the time and resources needed for exploring yet more clusters, and the fact that no significant performance increase was observed as the number of clusters grew. Despite this, it cannot be ruled out that using even more clusters could yield better performance. One potential approach not tested was splitting the data into different groups before time series clustering. For example, splitting the 1-Phase and 3-Phase data into different time series before applying time series clustering to the groups separately could potentially increase performance. A further possible approach would be to use a so called Sakoe-Chiba band or Itakura Parallelogram to constrain the DTW-algorithm, as this has sometimes shown to produce better results.[36]

Another limitation is the lack of data. Though 1878 charging sessions stored in as many CSV-files can be considered quite a lot, it is not a huge number with respect to time series clustering, and quite a low number with respect to the field of deep learning. Furthermore, the data available is limited to the winter season, only having being collected in the relevant format between 2023-12-18 and 2024-02-14. This means that other factors, such as ambient temperature and seasonal variation of commuting method for individuals, is not represented in the data. This is not to mention the overfitting of the models to the winter season, as it cannot be ruled out that EV batteries behave differently during other seasons and weather conditions. Meaning that new time series measured during other seasons and conditions may not fit the clusters constructed during the winter season, and therefore be classified in a way that is strenuous. This could also mean that great variations between different time series could be classified as the same clusters, simply because no cluster behaves similarly. Future work could benefit from using more data and include the features discussed.

A way to possibly increase performance is to include features of external variables, such as temperature, humidity, and hours of sunshine. These could potentially increase the accuracy of the models. Additionally, if the actual data collection was to be altered to include more data (such as more samples per minute) the time series clustering, although more time-consuming, could potentially perform better. This is not to mention that further technologies could installed to gather more information about the parked EVs, such as cameras or microphones. Though privacy concerns may be an obstacle to this.

To mention the strengths of the approach taken and of the results achieved here, however, the gain in accuracy of the prediction model when using clusters is significant. The constructed clusters are a result that can be built on – future time series can easily be classified into one of them, depending on the desired value of n. With this an increased accuracy could be gained by future predictions models, which is of course the very thing that is desired with such models. Another strength is the array of algorithms tested. Though by no means exhaustive of the current possible time series classification toolbox, it covers some of the most popular ones. Most settings available out-of-the-box in the used library (tslearn) were adjusted according to grid-search – a common practice. Additionally, approaches not available out-of-the-box were employed, such as the handling of clusters that were too big or small, or the various inclusions and exclusions of phases and data from the time series, or the various inclusions and exclusions of features for the deep learning. A further concrete strength is the insight into the collected data that has been shown in graphs and plots, providing useful information for any future future work EVs on Dansmästaren. As a final upside, the code written for this paper can easily be built on, if more data is collected or other features added, for instance.

6 Conclusions

In this study, the aim was to enhance prediction models for estimating the durations of EVs parked or charging. While the focus was primarily on charging duration prediction, the methodology presented can be adapted for parking duration prediction as well. The evaluation centered on improving model accuracy, employing RMSE as the metric for minimization, although MAE was deemed more intuitive for practical application.

The utilization of clusters significantly improved prediction accuracy, with enhancements ranging from 7% to 17% compared to models not assisted by time series clustering. The dependence of improvement on the number of samples (n) used for clustering suggests a nuanced relationship, where deeper insights into charging behaviors may be gained from longer time series.

Despite the promising results, several limitations and avenues for future exploration were identified. The study's scope was constrained by time and resources as it was performed by one person during one semester as is standard for master's theses, limiting the exploration of a larger number of clusters and different approaches. Additionally, the dataset's size and seasonal bias towards winter months raise concerns about generalizability to other seasons and weather conditions. Integrating external variables such as temperature and humidity could enhance model accuracy and robustness. Furthermore, the inclusion of additional data sources, such as high-resolution sensor data or external environmental factors may provide deeper insights into EV charging behaviors and their charging session time series, possibly paving the way for more reliable results.

These limitations notwithstanding, the study offers valuable insights into the usefulness of time series clustering for improving EV charging duration predictions. The developed clustering provides a foundation for future research, allowing for the classification of new time series data into pre-defined clusters. Moreover, the comprehensive evaluation of various algorithms and settings underscores the versatility and adaptability of the approach as a whole. And so the question (i): "can using time series clustering aid a deep learning model in better predicting charging durations" and its paraphrased corollary from 3.1 "did using time series clustering in addition to the intermediate features significantly improve predictions compared to only using the intermediate features?" can be answered affirmatively, and the question (ii): "what can a set of settings, approaches, and hyperparameters well suited for that be?" is outlined in Section 3 and Section 4 in general and by Section 4.1 and Section 4.2 in particular.

Moving forward, efforts to expand the dataset, incorporate external variables, and refine clustering algorithms show promise for further enhancing the accuracy of prediction models. By addressing these challenges and building upon the strengths of the approach taken here, future predictive models can offer more reliable estimations, increasing the practical utility of EV charging prediction in real-world scenarios.

References

[1] European Commission (n.d.), The European Green Deal. Available online: https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/european-green-deal_en (2024-03-11),

[2] Winton, N. (2023), European EV Sales Growth Slows, But 2030 Forecasts Remain Ambitious. Available online: https://www.forbes.com/sites/neilwinton/2023/11/02/european-ev-sales-growth-slows-but-2030-forecasts-remain-ambitious/ (2024-03-11)

[3] Bloomberg (2024), Electric Vehicle Market Looks Headed for 22% Growth This Year. Available online: https://www.bloomberg.com/news/newsletters/2024-01-09/electric-vehicle-market-looks-headed-for-22-growth-this-year?embedded-checkout=true (2024-03-11)

[4] Power Circle (2024), Elbilsåret 2023. Available online: https://infogram.com/elbilsaret-2023-1hnq4107zjevk23?live (2024-04-09), meta-information: https://powercircle.org/elbilsaret-2023/ (2024-04-09)

[5] Wargers, A., Kula, J., Ortiz de Obregon, F., Rubio, D., (2018), Smart Charging: Integrating a Large Widespread of Electric Cars in Electricity Distribution Grids, European Distribution System Operators for Smart Grids: Brussels, Belgium. Available online: https://web.archive.org/web/20220121052715/https://www.edsoforsmartgrids.eu/wp-content/uploads/EDSO-paper-on-electro-mobility-2.pdf (2024-05-02)

[6] Hildermeier, J., Kolokathis, C., Rosenow, J., Hogan, M., Wiese, C., & Jahn, A., (2019), Smart EV Charging: A Global Review of Promising Practices, World Electric Vehicle Journal, 10(4), p. 82. Available online: https://doi.org/10.3390/wevj10040080 (2024-03-25)

[7] Flygare, C., Wallberg, A., Jonasson, E., Castellucci, V., & Waters, R. (2024), Correlation as a method to assess electricity users' contributions to grid peak loads: A case study, Energy, 288, 129805, pp-1-2. Available online: https://doi.org/10.1016/j.energy.2023.129805 (2024-03-26)

[8] Uppsala parkering (n.d.), Dansmästaren. Available online: https://uppsalaparkering.se/vara-parkeringar-test/dansmastaren/ (2024-03-25)

[9] Miyazaki, K., Uchiba, T., & Tanaka, K. (2020), Clustering to Predict Electric Vehicle Behaviors using State of Charge data, 2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe: Madrid, Spain, 2020, pp. 1-6

[10] Shahriar, S., Al-Ali, AR., Osman, AH., Dhou, S., Nijim, M. (2020), Machine learning approaches for EV charging behavior: A review. IEEE Access, 2020 Sep 11, 8, pp.168980-93.

[11] Shahriar, S., Al-Ali, AR., Osman, AH., Dhou, S., Nijim, M. (2021), Prediction of EV charging behavior using machine learning. IEEE Access. 2021 Aug 6, 9, pp. 111576-86.

[12] He, Z., Gao, M., Ma, G., Liu, Y., Tang, L. (2016), Battery grouping with time series clustering based on affinity propagation. Energies, 2016 Jul 19, 9, p. 561.

[13] Mohammadi M, Thornburg J. Strategic EV charging and renewable integration in Texas. In2024 IEEE Texas Power and Energy Conference (TPEC) 2024 Feb 12 (pp. 1-6). IEEE.

[14] Cui, D., Wang, Z., Liu, P., Wang, S., Zhao, Y., Zhan, W. (2023), Stacking regression technology with event profile for electric vehicle fast charging behavior prediction. Applied Energy, 2023 Apr 15, 336, p. 120798.

[15] LeCun, Y., Bengio, Y., & Hinton, G., (2015), Deep Learning. Nature, 521(7553), pp. 436–444.

[16] Agatonovic-Kustrin, S., & Beresford, R. (2000), Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research, Journal of Pharmaceutical and Biomedical Analysis, 22(5), pp. 717-727.

[17] Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018), Activation functions: Comparison of trends in practice and research for deep learning, arXiv preprint.

[18] Saxena, A., Prasad, M., Gupta, A., Bharill, N., Patel, O. P., Tiwari, A., Er, M. J., Ding, W., & Lin, C. T. (2017), A review of clustering techniques and developments, Neurocomputing, 267, pp. 664-681.

[19] Aghabozorgi, S., Shirkhorshidi, A. S., & Wah, T. Y. (2015). Time-series clustering–a decade review., Information Systems, 53, pp. 16-38.

[20] Alqahtani, A., Ali, M., Xie, X., & Jones, M. W. (2021). Deep time-series clustering: A review, Electronics, 10(23), p. 3001.

[21] Müller, M. (2007). Dynamic time warping, Information retrieval for music and motion, pp. 69-84.

[22] Tavenard, R. (2017), Dynamic Time Warping. Available online: https://tslearn.readthedocs.io/en/stable/user_guide/dtw.html (2024-04-16)

[23] Tavenard, R. (n.d.), An introduction to Dynamic Time Warping. Available online: https://rtavenar.github.io/blog/dtw.html (2024-04-16)

[24] Cuturi, M., & Blondel, M. (2017), Soft-DTW: a Differentiable Loss Function for Time-Series. Proceedings of the 34th International Conference on Machine Learning, in Proceedings of Machine Learning Research, 70, pp. 894-903. Available online: https://proceedings.mlr.press/v70/cuturi17a.html (2024-04-16)

[25] Huang, X., Ye, Y., Xiong, L., Lau, R. Y., Jiang, N., & Wang, S. (2016), Time series k-means: A new k-means type smooth subspace clustering for time series data, Information Sciences, 367, pp. 1-3.

[26] Tavenard, R. (2017), TimeSeriesKMeans. Available online: https://tslearn.readthedocs.io/en/stable/gen_modules/clustering/tslearn.clustering.TimeSeriesKMeans.html (2024-04-19)

[27] Dhillon, I. S., Guan, Y., & Kulis, B. (2004), Kernel k-means: spectral clustering and normalized cuts. Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 551-556.

[28] Cuturi, M. (2011), Fast global alignment kernels. Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 929-936.

[29] Tavenard, R. (2017), KernelKMeans. Available online: https://tslearn.readthedocs.io/en/stable/auto_examples/clustering/plot_kernel_kmeans.html (2024-04-20)

[30] Paparrizos J, Gravano L. (2015), k-shape: Efficient and accurate clustering of time series. Proceedings of the 2015 ACM SIGMOD international conference on management of data 2015 May 27, pp. 1855-1870.

[31] Tavenard, R., Faouzi, J., Vandewiele, G., Divo, F., Androz, G., Holtz, C., Payne, M., Yurchak, R., Rußwurm, M., Kolar, K., & Woods, E. (2020), Tslearn, A Machine Learning Toolkit for Time Series Data. Journal of Machine Learning Research, 21(118), pp. 1-6. Available online: http://jmlr.org/papers/v21/20-091.html (2024-04-16)

[32] Nikolopoulos, A. (2022), Designing a platform for smart electric vehicle charging - a case study in Uppsala, Sweden. (ELEKTRO-MFE). Available online: https://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-481721 (2024-03-27)

[33] Charge Amps (n.d.), EV CHARGER Charge Amps Aura. Available online: https://www.chargeamps.com/product/charge-amps-aura/ (2024-03-27)

[34] Kud, A. (2023), Why we need encoding cyclical features. Available online: https://medium.com/@axelazara6/why-we-need-encoding-cyclical-features-79ecc3531232 (2024-03-31)

[35] Van Wyk, A. (2022), Encoding Cyclical Features for Deep Learning. Available online: https://www.kaggle.com/code/avanwyk/encoding-cyclical-features-for-deep-learning (2024-03-31)

[36] Z. Geler, V. Kurbalija, M. Ivanović, M. Radovanović & W. Dai. (2019), Dynamic Time Warping: Itakura vs Sakoe-Chiba, 2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA): Sofia Bulgaria, pp. 1-6.