CHALMERS
UNIVERSITY OF TECHNOLOGY

Energy Prediction of Electric Trucks' Auxiliaries

-A Data-driven Approach Using Field Test Data

Master's thesis in Systems, Control and Mechatronics

MARTIN CLAESSON
ERIK NORHEIM

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2022
www.chalmers.se

Master's thesis 2022

Energy Prediction of Electric Trucks' Auxiliaries

-A Data-driven Approach Using Field Test Data

MARTIN CLAESSON
ERIK NORHEIM

Department of Electrical Engineering
Systems and Control
Chalmers University of Technology
Gothenburg, Sweden 2022

Energy Prediction of Electric Trucks' Auxiliaries
-A Data-driven Approach Using Field Test Data
Martin Claesson and Erik Norheim

Supervisors:
Victor Olsson, Volvo Group
Balázs Kulcsár, Chalmers University of Technology, Department of Electrical Engineering

Examiner:
Balázs Kulcsár, Chalmers University of Technology, Department of Electrical Engineering

Master's Thesis 2022
Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Electric medium-duty truck driving.

Typeset in $\LaTeX$
Gothenburg, Sweden 2022

Energy Prediction of Electric Trucks' Auxiliaries
-A Data-driven Approach Using Field Test Data
Martin Claesson and Erik Norheim
Department of Electrical Engineering
Chalmers University of Technology

Abstract

In today's society, a large proportion of transportation is carried out on land by fossil-fueled vehicles but there is an increasing trend towards electrifying vehicles. A fundamental disadvantage of Electric Vehicles (EVs) is the limited range. An often overlooked aspect of the energy consumption is the auxiliaries, which especially for trucks can be a substantial proportion of the total energy consumed. This thesis investigates data-driven methods to predict the energy of electric trucks' auxiliaries using historical data from Volvo Group.

The analysis and predictions were done on preprocessed data to ensure that the results are derived from feasible values of the signals measured. The analysis laid the groundwork for determining the quality of the data and which methods were applicable to the problem. Results indicate that the energy consumption of auxiliaries is difficult to predict with the inputs available and does not always follow a typical nor expected pattern, despite a significant correlation with the ambient temperature and time. Furthermore, preprocessing of data proved to be a fundamental process in enabling accurate predictions.

Testing models of different complexity and types, the thesis found significant improvements in the energy prediction compared to algorithms found in relevant research papers when applied to the data. Machine Learning (ML) models performed well considering the complexity of the problem, the available signals, and the large amount of data. Lastly, important future work is presented that can further improve the prediction of auxiliaries and thereby contribute to more accurate range estimations.

Keywords: energy prediction, electric vehicles, auxiliaries, machine learning, MLP, XGBoost

Acknowledgements

First of all, we would like to thank the team at Volvo Group Electromobility for having us as thesis workers and showing great interest in our work. We would especially like to thank our supervisor at Volvo Group, Victor Olsson, for his continuous support and guidance with all aspects of the thesis. Finally, we want to thank our academic supervisor and examiner Balázs Kulcsár for helping us tackle difficulties along the way with valuable advice and ideas.

Martin Claesson and Erik Norheim, Gothenburg, June 2022

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AC	Air-Conditioning
BEV	Battery Electric Vehicle
CART	Classification And Regression Tree
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
ePTO	electrical Power Take-Off
EU	European Union
EV	Electric Vehicle
HVAC	Heating, Ventilation and Air-Conditioning
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MBE	Mean Bias Error
MI	Mutual Information
ML	Machine Learning
MLP	Multi-Layer Perceptron
PCA	Principal Component Analysis
RES	Renewable Energy Source
SGD	Stochastic Gradient Descent
XGBoost	eXtreme Gradient Boosting

Contents

List of Figures

List of Tables

1 Introduction

Transportation of goods is an important part of today's society and trucks are an integral part of it. In the European Union (EU) there are about 6.2 million trucks circulating and 73.1% of all freight transportation on land is carried out by trucks [1]. There is an increasing trend in the EU to decarbonize vehicles as 0.24% of the trucks on road in January of 2022 have a powertrain with zero-emission compared to 0.04% in 2019 [2], and the adoption of electric trucks has been projected to surpass 30% by 2030 [3]. The fleet of buses in the EU consists of 0.9% Electric Vehicles (EVs) and 1.4% hybrid electric, but in the Netherlands, the share of electric buses is 12.4% [2]. Following improvements in battery capacity and lower costs of batteries, the trend of electrifying vehicle fleets is expected to continue. Electric trucks can either have electric energy on-board, for instance, inside a battery, or receive energy externally, like a trolley truck that is given electricity by overhead lines along its path. This thesis will however only cover EVs with batteries on-board, sometimes named Battery Electric Vehicle (BEV).

A study made in 2012 presents that EVs powered by the European electricity mix and with an assumed lifetime of 150,000 km for the vehicle and batteries, can achieve a 10% to 24% decrease in global warming potential compared to conventional diesel or gasoline vehicles [4]. The lifetime was set by the authors to 150,000 km but they found the lifetime at the time to range between 150,000 and 300,000 km. A longer lifetime typically benefits EVs more than fossil-fueled vehicles due to the greater production impact of EVs. The decrease in global warming potential is projected to grow larger in the EU since the Renewable Energy Sources' (RES) share in the gross final energy consumption has increased steadily from 10 to 21.3% between the years 2005 and 2020, and the proposed RES share target in 2030 from the 'Fit for 55 package' is 40% [5]. There is also a pledge to achieve climate neutrality by 2050 in the EU [5].

The potential of electrifying trucks for transporting goods differs greatly when comparing countries. In a study from 2019 EVs can cover up to 71% of the freight transport in Switzerland in comparison with 38% in Finland, greatly due to differences in average trip length and gross vehicle weight [6]. Increased electrification of vehicles however introduces considerable strains on the electrical grid. The total energy needed in society increases and having numerous trucks charging simultaneously at a local charging station can require power in the magnitude of megawatts. The work of [6] further states that the potential of EVs is significantly affected by the available fast recharging infrastructure. In particular overnight charging is expected to be the dominant choice of charging as it is viewed as the cheapest option and the technology for it is already available [3].

BEVs today have a limited range due to the battery capacity on-board making it important to accurately predict the energy consumption for a given route. In the EU there are also regulations on driving, introducing further constraints on when and where the vehicle can charge. For instance, a driver must take a break of at least 45 minutes after driving for four and a half hours according to regulation 561/2006 [7]. Accurate energy predictions can assist in finding a suitable charging station to charge at during the required break. Precise predictions are also very useful for battery sizing and planning routes more efficiently. These factors are very important for a successful implementation of BEVs in society, which in turn can contribute to lowering the impact on the environment compared to fossil-fueled vehicles.

1.1 Background

The energy consumption of trucks can be divided into two main categories; propulsion of the vehicle and auxiliaries. The auxiliaries account for the extra energy needed to operate the vehicle, such as the Air-Conditioning (AC), cabin displays, temperature control of batteries, and in some cases add-ons to the exterior of the vehicle. Examples of such add-ons are a fridge unit, a crane, or a tail-lift, and are called electrical Power Take-Off (ePTO).

Predicting the energy consumption of the auxiliaries can be more difficult to perform accurately on trucks compared to passenger cars due to the increased number, size, and complexity of sources drawing power from the batteries. The major power sources for a car are typically the propulsion of the vehicle, AC, and heating/cooling of the cabin. A passenger car usually has a similar weight and purpose of each trip, to transport people between destinations, unlike trucks which can have additional applications added to the exterior of the vehicle depending on the mission. The energy consumption of the add-ons can be difficult to predict as they can have an energy consumption of various types and depend on the route as well as driver behavior. For instance, a fridge unit typically consumes energy depending mainly on time and temperature, while a crane has sudden power spikes when lifting heavy loads. As for the increase in size compared to cars, an example is that air compressors have a wider field of application for trucks as it is used for braking, suspension, and controlling the height of the chassis.

Several studies have shown that it is possible to accurately predict the energy consumption of hybrid and electric vehicles with various methods. Mean absolute percentage errors of 5.9% with an instantaneous energy consumption model including auxiliaries [8] and 5.76% or 3.56% (down- or uphill) for an electric bus with a probabilistic Machine Learning (ML) approach excluding auxiliaries [9] have been achieved previously for their respective choice of validation. Less research has however been conducted on how well the auxiliaries' energy consumption of electric trucks can be predicted.

The auxiliaries typically consume less energy than the propulsion and are often considered to be linearly time-dependent or even omitted. Examples include an electric bus and medium-duty truck in [9], electric medium-duty truck in [10], and small cars as well as small trucks in [11]. However, several factors such as the ambient temperature and usage of add-ons can increase the energy consumption of the auxiliaries. This can make the auxiliaries hold a significant share of the total energy consumed making it an important factor to accurately predict when estimating the range of the vehicle.

1.2 Related Work

In [12], data from EVs has been collected in Japan indicating an asymmetrical 'U' shape of a relationship between the total energy efficiency (kWh/km) and ambient temperature. When they fitted second and third-order polynomials the minimum was approximately 17.5 °C. The preferred temperature of the batteries and cabin play an important role regarding which ambient temperature results in the lowest auxiliary consumption. In [13] linear regression was used for the auxiliary energy E stated as

$$\Delta E_{auxiliaries} = B|20 - T| \frac{\text{Duration of auxiliaries switched on}}{\text{Total duration of trip}} \Delta t \tag{1.1}$$

with regression coefficient B, ambient temperature T in °C, and time t in seconds. In [14], (1.1) is extended with a constant energy term $E_{Aux,const}$, the minimum energy consumed by some auxiliary systems at 20°C, multiplied with Δt. For regular vehicle use, it has been shown that nearly no heating or cooling is needed when the ambient temperature is 20°C, and for the total energy consumption, the parameter with the largest uncertainty is the average auxiliary power [15]. Additionally, [16] presents an EV fleet simulation in which the efficiency (km/kWh) is maximized at an ambient temperature near 20 °C.

The definition of auxiliaries can differ and the modeled parameters of them can vary greatly, including which temperature ranges the power of auxiliaries are defined within. For a Nissan Leaf model, 300 W is used as a baseline in [17] while [18] has a minimum of 200 W at 20°C and a maximum of 6000W at -15°C. In [8] a higher baseline of 700 W, 850W at 25°C, 1200 W at 35°C and 2200 W at -5°C was used. Again for a Nissan leaf, the auxiliary energy consumption at -7°C during the WLTC driving cycle when simulating was about 33% of the total consumption in [14], highlighting the importance of predicting the auxiliary load.

The predictions in [9] are done on links, which is also done in this work but only for the auxiliaries. The definition of a link is the distance traveled on a road between two nodes, for example, two road intersections, lacking any intersection between them. A route in this work is defined differently than in [9]. One of the main reasons is that the vehicle does not always drive in a loop in the logged data. In this work, a route is defined as a sequence of links for which the time difference between the links does not exceed 5 minutes. Effectively, a new route is created whenever the vehicle starts to log data again after it has been turned off for at least 5 minutes before logging data. Furthermore, the predictions made in this work are average power per link instead of energy per link as in [9]. By doing this the possible range of target values is restricted to in between zero and the maximum power of the auxiliaries, which also makes it easier to interpret the results. The time duration for each link is however known, meaning the predicted energy is simply the known time duration multiplied by the predicted power. The final prediction and evaluation will be done both per link and on entire routes by aggregating the energy per link of each route.

1.3 Scope

The thesis aims to create a model that can accurately predict the auxiliary energy consumption for different types of a Volvo medium-duty electric truck. The main focus will be the auxiliaries' energy consumption excluding ePTO and the intention is to apply ML using collected data from electric trucks, primarily field test data from Volvo Group. The field test data in this work is primarily data collected from electric trucks driving at test tracks and in scenarios that reflect the intended use. Due to the secrecy of the data, the publicly available thesis will not include all results. From here on the term auxiliaries will exclude the ePTO such that they are separated from each other since the focus is not on the ePTO. Due to using field test data that has not been preprocessed, a considerable amount of effort is required into using data engineering methods to make sure the data is viable to use for predicting the energy accurately. The thesis seeks to contribute with new knowledge to the prediction of trucks auxiliaries using a data-driven approach by answering the following questions:

- What is the research front regarding the prediction of energy consumption for electric trucks' auxiliaries?
- Which ML model is optimal for predicting the energy consumption of an electric truck's auxiliaries?
- How well can an ML model predict the auxiliary energy consumption of an electric truck offline?

The predictions in this work will be done before a truck starts driving and evaluated after the truck has finished driving. Therefore, the input variables of the model can only consist of parameters that are available before a truck starts its trip. Only historical data will be used to train the models and the data used will be limited to a subset of data of the medium-duty electric truck. Furthermore, the data will be used in a way that ensures the drivers' privacy. The algorithms developed for predicting the auxiliary power will thus not have access to any personal data, and the predictions can in no way discriminate any driver.

1.4 Delimitations

Analyzing and training algorithms on data from only one truck manufacturer, on one type of model, might not generalize well to other trucks. Moreover, the algorithms used were improved continuously during the implementation phase, but not optimized to the greatest extent as it was not the focus of the thesis. The same argument is applied to the componentwise analysis. The number of algorithms and approaches have also been limited meaning that there is more work necessary to get the full picture of how well the auxiliary energy can be predicted. As previously mentioned, ePTO is excluded which presents a considerable limitation for vehicles with significant energy consumed by the ePTOs.

1.5 Thesis Outline

Chapter 2 presents the theoretical knowledge necessary for understanding the methods used later. First off is the definition of what an auxiliary component is along with how they can be grouped is described. Thereafter the theory of selecting features as input to the models, the field of ML, and the chosen ML models.

Following the theory is how the chosen methods are implemented in Chapter 3, including the essential procedures of processing and analyzing the data before making predictions. In Chapter 4 the results from the data analysis and models' predictions are presented and compared to each other. The baseline is a simple linear regression algorithm that has been used in several research papers. Conclusions from the thesis are lastly presented in Chapter 5 along with recommendations for further work.

2
Theory

This chapter describes the necessary theory to understand and apply the implementation presented in Chapter 3. At first, the auxiliaries and grouping of them are defined. Thereafter methods for selecting features as input to ML models, and lastly basic theory of the ML field and ML models applied are explained.

2.1 Auxiliaries

The electrical system of an electric truck is typically divided into multiple subsystems. A common approach is to separate the energy consumed by the components directly propelling the vehicle into a subsystem for propulsion, and group the remainder of the components as auxiliaries. This work considers all energy consumption that is not directly connected to the vehicle's propulsion as auxiliary energy consumption. Some papers also group the Heating, Ventilation, and Air-Conditioning (HVAC) as a subsystem. Typical auxiliaries in a truck include the HVAC system, steering pumps, and components for heating/cooling batteries.

The truck model analyzed has three different types depending on the field of application; distribution, refuse, and fridge. They all have a similar electrical system in which the auxiliaries can be divided into low and high-voltage subsystems. The auxiliary components consume power in various patterns, such as linearly with time or per usage. Several components consume energy only when deviating from a specified metric, for instance, heating or cooling of the cabin and batteries.

2.1.1 Cabin

Examples of components considered to be connected to the cabin include the HVAC system, displays, and windshield wipers. Several of these auxiliaries depend on both the individual driver's preferences, for instance, the cabin temperature, and also external settings, such as the ambient weather condition. The temperature and solar irradiation affect the energy consumption of the HVAC system, introducing variance in the energy consumption of trips despite driving similarly. When comparing plug-in electric vehicles in different states in the USA the highest AC load to heater load ratio was 6.3 in Arizona, and the lowest was 0.48 in Alaska [19]. Preconditioning via heating or cooling using off-board powered thermal preconditioning, however, can significantly decrease the energy expenditure of the climate control system [20]. In Phoenix, Arizona, an EV has been computed to have a seasonally averaged maximum of 8.75% increase in range when preconditioning the cabin [21].

2.1.2 Driveline

During driving, there are several subsystems required to be turned on for optimal performance. Examples include the air compressor, steering assistance for the driver, and heating/cooling of batteries. The batteries are components that have an optimal temperature range for operating which introduces a significant correlation between the ambient temperature and auxiliary energy consumption. They can be preconditioned to reduce the auxiliary energy consumption during driving by lowering the need for heating/cooling of them [20]. Battery thermal management system usage and cabin preconditioning when parked before driving increased an EV fleet's median efficiency (km/kWh) by 8% and 9% respectively when the ambient temperature is at -10 °C and +40 °C [16]. Furthermore, temperatures too low can decrease the energy possible to extract from the batteries while too high temperatures can increase battery degradation and self-discharging [16].

2.1.3 ePTO

Some types of trucks have external add-ons which consume auxiliary energy. Examples include a tail-lift for a distribution truck, a compactor for a refuse truck, and a fridge unit for a fridge truck. These add-ons vary greatly in how often they are used and how much energy they consume. The ePTO for the truck is considered to be part of the high-voltage system and is separated from the other auxiliaries which have a lower voltage.

2.2 Feature Selection

A feature is a measurable variable describing a part of the observed process, and the process can consist of hundreds of features [22]. Before selecting features as inputs to a model it is important to consider transforming features for compatibility and performance reasons, including scaling of features. Thereafter the process of feature selection is to select the relevant features to choose as input to the models and the process of selection can include several different methods in combination. The methods can be derived from statistics, information theory, manifold, and rough set [23]. Broadly, methods can be divided into two categories that either use the features' importance themselves based on the data, filter methods, or the predictor's performance to evaluate the features' importance, wrapper methods [24].

The performance of any model is affected by what input it is given. Given more features as input, the accuracy can increase but requires more computations making the training process slower [22]. If not given enough information as input, the accuracy will be sub-optimal, resulting in a trade-off regarding the number of features and computational complexity. Having fewer input features can even result in some models generalizing better since it has fewer adaptive parameters and can be trained quicker [25]. Features that do not increase the performance are deemed redundant and should not be included. Distinct performance benefits of including a variable can often be directly determined by investigating the association of variables.

2.2.1 Association of Variables

There exist several types of measures to define how well variables are associated with each other, for instance, the correlation between variables. Examples of correlation metrics include Pearson's r (product-moment correlation) and Spearman's ρ [26]. Pearson's r can be written as

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2}\sqrt{\sum(x_i - \bar{x})^2}} \tag{2.1}$$

where x, y are variables and $\bar{x}, \bar{y}$ are the sample means. The expression in (2.1) has limits of ± 1 for r and can only handle correlation between two continuous variables at a time. Pearson's r evaluates the linear relationship between two numerical variables and is sensitive to outliers, while Spearman's ρ is robust against outliers and determines monotonic relationships between two numerical variables [26]. Spearman's ρ can thus determine relationships that are nonlinear. To calculate the association of categorical values with other categorical or numerical values, other methods must be used. A practical correlation coefficient ϕ_K for categorical values is presented in [26]. It is based on the Pearson's χ^2 contingency test that can capture nonlinear dependency and enables correlation of all variable types.

Another available metric of two variables' association is Mutual Information (MI) [22]. It can be used for both categorical and numerical values. Using Shannon's definition for entropy, $H(y)$ corresponding to the information content in the output Y, and observing a variable X, the MI can be written as

$$MI(Y,X) = H(Y) - H(Y|X) = -\sum_{y} p(y)log(p(y)) + \sum_{x}\sum_{y} p(x,y)log(p(y|x)) \tag{2.2}$$

with probabilities p and $H(Y|X)$ being the conditional entropy [22]. The expression in (2.2) will be equal to zero if they are independent, or above zero if they are dependent. It can also be applied to continuous variables by exchanging the summations for integrals.

2.2.2 Filter Methods

Filter methods utilize a variable ranking technique using a criterion, for instance, a threshold, to determine which variables to include based on the values used for ranking [22]. The method is straightforward to implement and it is easy to choose features from the ranking. There exist several techniques to rank the variables with regard to their association to the target variable, such as correlation and MI as explained in Section 2.2.1. A drawback is that both correlation and MI do not take the used variables' correlation to other variables into consideration, resulting in the possibility of having redundant features [22]. In addition, these methods do not take into account features that individually can have a low association with the target variable, but in combination be more informative.

2.2.3 Wrapper Methods

Wrapper methods use the performance of the predictor to evaluate the different variable subsets, enabling comparison of how features in combination can improve the performance, unlike filter methods [22]. In turn, this means that for optimal results the wrapper must be rerun for each new learning algorithm used since they function differently. When using an exhaustive search procedure to check all possible combinations of features, the number of possible feature subsets is 2^d, d being the number of possible features [25]. Testing all subsets when having a high-dimensional space becomes computationally costly and there exist several sub-optimal methods such as the genetic algorithm that can be applied [22], [23]. One possible approach to reduce the cost is to only search for subsets of features that have already been filtered by a filtering method. This two-stage implementation can be called a hybrid feature selection method [23]. Additionally, subsets of features can be selected with wrapper methods such as forward selection and backward elimination. In forward selection, it is common to start with no features selected and then add features sequentially. In backward elimination, one starts with all features selected and removes one feature at a time such that redundant features are removed [27].

2.2.4 Transformation of Features

Features such as the length of the trip can be used directly as a numerical input to certain models while a timestamp can be decomposed into multiple features such as month and time of the day. Various models might only be capable of handling certain types of inputs, for example, numerical values, and some numerical data have no natural ordering resulting in a need to be categorized. Thus, certain features need to be transformed to make use of all the information correctly before they can be used. Examples of necessary transformations can include encoding categorical variables to numeric values, transforming numerical values into categories, and scaling.

Feature scaling is a type of transformation that standardizes the range of values the features can take on. Larger values do not necessarily correspond to a larger importance of a feature. To exemplify, the values decrease significantly when changing the metric of weight from kilogram to tonne, but the importance remains the same. Scaling the input before training ensures that the parameters of a model can be given a suitable random initialization [25]. The variables can also be made more interpretable by scaling the features and can improve performance to a varying degree depending on the values of the feature and the model at hand. Scaling can furthermore significantly improve the rate of convergence when training a model with gradient descent, as with Batch Normalization [28]. Caution however must be taken before transforming the features since it can change the distribution of the feature. Standardization using the mean and variance does not retain the input distribution if the input is not Gaussian, resulting in loss of information. Methods such as standardization and min-max scaling are also sensitive to outliers.

2.3 Machine Learning

An ML model consists of an algorithm that learns to perform a task from the data it is given, with the central challenge being to perform well on new, unseen data [29]. Typically, an ML model has a loss function that it tries to minimize during training using optimization on the training set. How well the model learns, therefore, depends greatly on the quality of the data fed to it since the algorithm tries to infer patterns from the used data on unseen data. ML models are furthermore suited differently well to different types of problems and data, for example, if there exists a target variable or not.

ML problems are commonly divided into supervised and unsupervised learning problems [30]. In supervised learning, each training input $\mathbf{x}$ has a corresponding target $\mathbf{y}$. The training dataset can thus be written as $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_i^N$, where N is the number of samples in the training data with indexing of samples i [30]. The aim of the supervised learning problem is to train a model to predict $\mathbf{y}$, given $\mathbf{x}$. The target can be either categorical or real-valued, making it a classification or a regression problem respectively [30]. In the unsupervised learning problem, the training inputs in the data have no targets. Instead, the objective is to learn important properties of the training samples $\mathbf{x}$ [30]. The target variable for this thesis is a measured real-valued variable, the average auxiliary power of a road link, making the prediction a supervised regression problem.

Typically the algorithms have settings called hyperparameters that are determined externally from the learning algorithm that change the behavior of the model [29]. Some of the hyperparameters external to the architecture of the model include the learning rate, number of epochs trained, and the choice of an optimizer. An epoch is the number of passes of the entire test dataset. When creating an ML model it is important to determine the model's hyperparameters which control the capacity of the model and the ability to fit a variety of functions. Examples of model hyperparameters include the topology and size of a network. A too small capacity tends to underfit the data, so it cannot fit the data well, while a too large capacity tends to overfit the data. When overfitting the data the model can find patterns that exist in the training data, but not in the test data. An approach to reduce the overfitting is to add regularization to the model, for instance, weight decay which adds a preference for smaller weights of the network. In Figure 2.1 the errors of training and testing are compared when overfitting.

2.3.1 Optimization of Machine Learning Models

There are many differences between the optimization methods used for ML models and traditional optimization. One important aspect is that ML models use an indirect way of minimizing the loss [29]. That is, the objective is to minimize the loss based on the test set but the problem is approached by minimizing the training loss instead [29]. However, it is often not wise to only focus on minimizing the training loss as this comes with a great risk of overfitting. A common approach to prevent

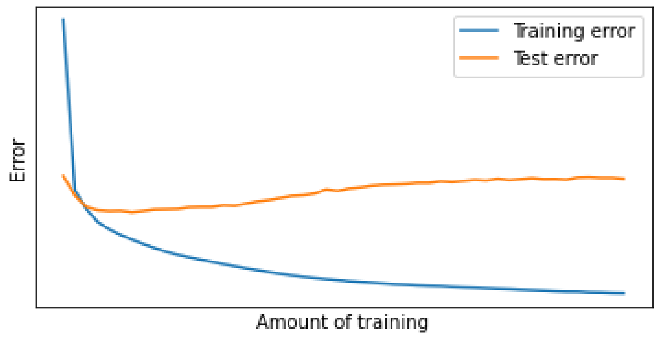


Figure 2.1: Prediction error versus the amount of training of an ML model. When the training error decreases while the testing error increases consecutively, overfitting is occurring.

overfitting is to split the data into three different sets and implement early stopping [29]. One set is used for training the model, the training set, one for validating the training, the validation set, and the last one for testing the model after the training is complete, the test set. Early stopping is implemented by evaluating the model on a validation set after each iteration of training and stopping the training process when the validation loss consecutively increases [29]. Referring to Figure 2.1 and having a validation set used instead of a test set, the training should be stopped when the orange curve starts to increase.

ML models are often optimized with methods applying gradient descent, which is an optimization algorithm updating the model's weight vector $\mathbf{W}$ as

$$\mathbf{W}^{(\tau+1)} = \mathbf{W}^{(\tau)} - \eta \nabla L(\mathbf{W}^{(\tau)}) \tag{2.3}$$

with τ as the iteration step, η as the learning rate, and ∇L as the gradient of the loss function [31]. Furthermore, it is common to use mini-batch gradient descent, which means that only a randomly selected subset of the data is used to evaluate the loss function of which the gradient is calculated. This approximation is a lot less computationally costly compared to computing the exact gradient of the entire training set and can often result in faster convergence [29]. Moreover, gradient descent can also converge faster if momentum is used. Momentum uses a velocity term that is calculated based on previous gradients. This reduces the oscillations of the gradient trajectory and accelerates the training process [29].

2.3.2 Loss Functions

Optimization of ML models requires that a suitable loss function is chosen. Outliers can cause a significant decrease in the model's performance if not dealt with correctly [32]. For instance, using the squared error loss r^2 in the presence of outliers can result in the outliers having a large impact on the model weights [32]. This effect can be reduced by using other loss functions that are more robust regarding outliers. One possible solution is to use the absolute loss $|r|$, which is robust in the sense that

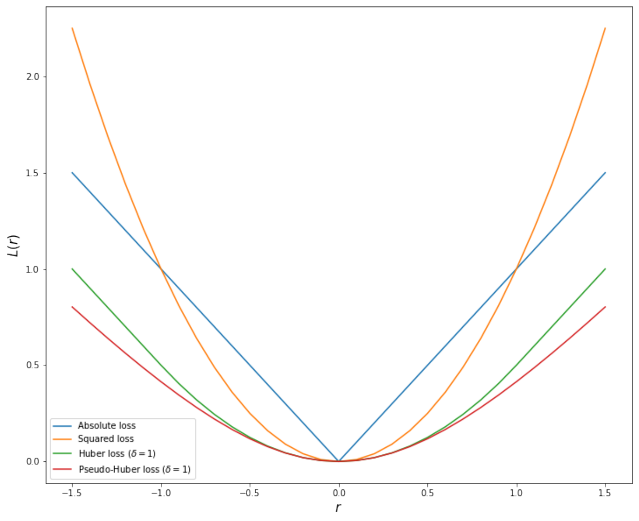


Figure 2.2: Comparison of the absolute, squared, Huber, and Pseudo-Huber loss over residual values r.

residuals are not squared [32]. However, a disadvantage with the absolute loss is that it is not differentiable as $r = 0$. One approach that combines favorable aspects of the squared and absolute loss is the Huber loss

$$L(r,\delta) = \begin{cases} \frac{r^2}{2}, & |r| \leq \delta \\ \delta|r| - \frac{\delta^2}{2}, & |r| > \delta \end{cases} \tag{2.4}$$

which assigns a quadratic loss to the residual r if it is less than or equal to the parameter δ, otherwise an absolute loss [30]. As can be seen in Figure 2.2, the combination ensures a convex function when the loss is near zero and robustness for outliers when the loss is large. Additionally, the Huber loss is sometimes approximated to an expression that is twice differentiable and thus enables the loss to be used for second-order methods [33]. The approximation is called the Pseudo-Huber loss and can be calculated as in (2.5) [34].

$$L(r,\delta) = \delta^2\left(\sqrt{1 + \left(\frac{r}{\delta}\right)^2} - 1\right) \tag{2.5}$$

2.4 Machine Learning Models

Choosing the best model suited to the task at hand is a challenge that often requires testing several different models. In this section, the tested ML models are theoretically explained.

2.4.1 Polynomial Regression

A linear regression model calculates a linear function based on the input variables $x_{1,\dots,D}$ and model parameters $w_{0,\dots,D}$ [31]. In [31], the linear regression model is extended to model nonlinear functions by using basis functions $\phi(\mathbf{x})$. The model output can thus become nonlinear if a nonlinear basis function is chosen. For instance, polynomial regression can be obtained by modeling the j-order term in the polynomial with the basis function

$$\phi_j(x) = x^j \tag{2.6}$$

which enables the modeling of higher-order polynomials [31]. Using the basis function in (2.6), a j degree polynomial with variable x can be modeled as

$$\boldsymbol{\phi}(x) = \Big[\phi_0(x),\ \ \phi_1(x),\ \ \phi_2(x),\ \ \dots,\ \ \phi_j(x)\Big] = \Big[1,\ \ x^1,\ \ x^2,\ \ \dots,\ \ x^j\Big] \tag{2.7}$$

but polynomials are not limited to only using one variable. The extended linear regression model with variables $\mathbf{x}$, output $\hat{y}$ and M parameters can be written as

$$\hat{y}(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) \tag{2.8}$$

which can be simplified to

$$\hat{y}(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) \tag{2.9}$$

by setting the basis function $\phi_0 = 1$ [31].

In problems using (2.9), one aims to find suitable values of the model parameters $\mathbf{w}$, given the chosen basis functions $\boldsymbol{\phi}(\mathbf{x})$. The values of the model parameters are commonly determined by minimizing the squared error loss with respect to $\mathbf{w}$, which requires that basis functions of appropriate complexity are chosen to prevent the model from overfitting the data [31]. In [31], several methods to determine the model complexity are presented, and the choice of method often depends on the amount of training data available. For instance, the risk of overfitting can be reduced by increasing the number of training data. Furthermore, some of the data can be used in a validation set on which the different models can be evaluated. However, in the case of limited data available, a Bayesian approach to the regression problem can be used instead. The Bayesian approach enables the evaluation of a model's fit on the training set by calculating the marginal likelihood [31].

2.4.2 Multi-Layer Perceptron

The objective of a Multi-Layer Perceptron (MLP), also called feedforward neural networks, is to approximate some function f^* which maps an input $\mathbf{x}$ to the output $\hat{\mathbf{y}}$ with parameters $\boldsymbol{\theta}$ [29]. The information flows through the function with intermediate computations without any feedback, hence the name feedforward. The output can be viewed as consisting of multiple functions structured as

$$\hat{\mathbf{y}} = f(\mathbf{x}) = f^{(2)}(f^{(1)}(\mathbf{x})). \tag{2.10}$$

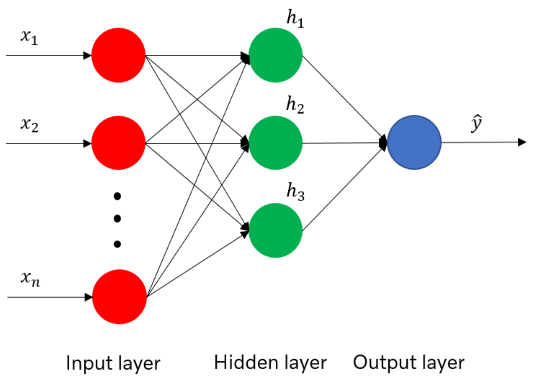


Figure 2.3: A two-layered MLP with one output having arrows indicating the flow of information. Input denoted as $\mathbf{x}_i$, hidden units as $\mathbf{h}_i$ and the output as $\hat{\mathbf{y}}$.

Most commonly the function $f^{(i)}$, using indexing i, is an affine transformation $\mathbf{Wx}+\mathbf{c}$ with weights $\mathbf{W}$ and bias $\mathbf{c}$ [29]. An MLP has one input and output layer respectively and at least one hidden layer. MLPs are fully connected, connecting every unit from one layer to all other units in the next layer. Figure 2.3 illustrates an example of a simple MLP. The width of the layers is determined by the number of units in each hidden layer and the depth of the model is the number of hidden layers added with the output layer in the network. The depth of the MLP in Figure 2.3 is 2, counting the hidden and output layer, and the hidden layer's width is 3 since there are three hidden units (h_1, h_2, h_3). Furthermore, an MLP introduces nonlinearities by having an activation function g applied elementwise, $h_i = g(f^i)$ [29]. Common choices for activation functions include

- logistic sigmoid: $\frac{1}{1+\exp(-x)}$
- hyperbolic tangent (tanh): $\tanh(x)$
- Rectified Linear Unit (ReLU): $\max\{0, x\}$.

There exist many different optimizers to choose from when implementing an MLP. Adam [35] is a popular first-order gradient-based method with momentum and a stochastic objective function. The simpler Stochastic Gradient Descent (SGD) has however shown to be able to generalize better [36]. Both can be implemented using mini-batches. Lowering the batch size affects the choice of learning rate as the model learns faster due to updating its parameters more times during an epoch. A larger batch size tends to result in slower learning, but with a lower variance of the accuracy for the validation set.

Dropout is a regularization method that improves the performance by randomly dropping units, including their connections, in a neural network with a fixed probability $1 - p$. The method reduces the co-adaptions on the training data and can thereby reduce overfitting [37]. Referring to Figure 2.3, dropping a unit corresponds to dropping, for instance, h_1, and its incoming and outgoing connections. As a result, the model has to find parameters that accurately predict the output independently of the dropped unit h_1 in this scenario. Typically dropout is only enabled during training and it increases the training time since it increases the noise of the parameters' updates [37].

Due to the added noise when including dropout, a larger learning rate and/or adding momentum is recommended to speed up the training, and max-norm regularization to constrain the weights of the network [37]. Common values of p regarding dropout are in the range [0.5, 0.8], but the value also depends on the choice of hidden units n. The network's size is furthermore affected by dropout since it decreases the capacity of a network, resulting in the recommendation of having n/p units with n being the optimal size of the layer of a standard neural network [37].

To introduce a measure of the MLP's uncertainty of its prediction, the method of adding dropout as a Bayesian approximation can be introduced [38]. An MLP model with dropout applied before every weight layer can be run, with dropout enabled also during prediction, to calculate the mean and predictive uncertainty from T stochastic forward passes. The final prediction is the mean, and the uncertainty is represented by the standard deviation. For regression tasks, $p = [0.8, 0.9]$ has been used in [38].

2.4.3 Classification And Regression Tree

A Classification And Regression Tree (CART), maps a model to the input by passing the input through a series of binary decisions. The series of binary decisions can be viewed as a tree with nodes that split into two nodes at each new level of the tree, where each split divides a decision region into two regions based on a variable and threshold [31]. See Figure 2.4 for a CART with binary decisions. The first node is called the root node and the end nodes in the tree are called leaf nodes. Each leaf node corresponds to a specific decision region to which a model is assigned. A model can be a single value in regression trees or a class in classification trees [31]. The model for each leaf node depends on the training samples assigned to it. For example, the value of the leaf node when using the squared error loss can be calculated as the average target value of the training samples assigned to the node [31].

Tree models generally have many possible solutions for a given problem and this can make traditional optimization methods insufficient to use for training [31]. It is therefore common to use a greedy optimization method, which sequentially chooses one node to split among multiple candidates [31]. The candidates also have multiple possible variables and thresholds that can be used to form the split condition, which must be determined. Finally, the greedy optimization method evaluates all candidates of nodes, variables, and thresholds, and chooses the one configuration that yields the lowest loss among all candidates [31].

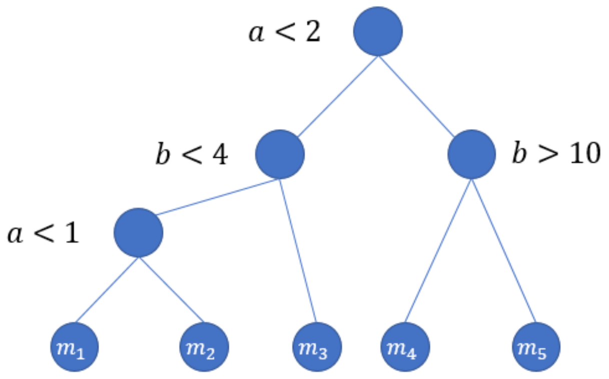


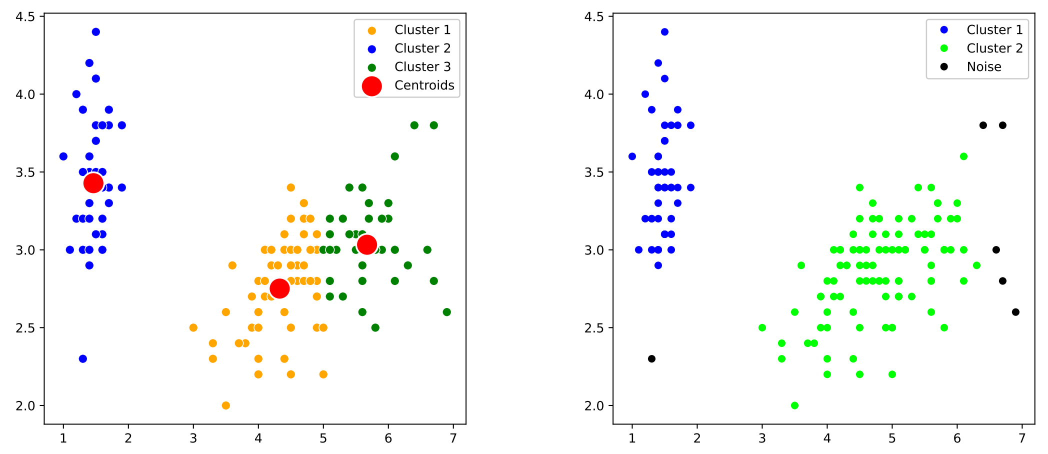
Figure 2.4: Example of a decision tree that assigns a model m based on binary decisions with variables a and b.

2.4.4 Boosting

In boosting, one can get improved accuracy by using several models to perform predictions [31]. Furthermore, models are added and optimized one at a time and the models added will emphasize on learning to predict the training samples that the last model predicted inaccurately [31]. A common type of boosting is gradient boosting, and an example of gradient boosting is presented in [39]. It proposes a boosting scheme of regression trees that are optimized with gradient descent. The method requires little preprocessing of data as it is robust to outliers and does not require any scaling of the input data [39]. There have been several other tree-based boosting algorithms developed that make use of the gradient boosting technique, of which eXtreme Gradient Boosting (XGBoost) [40] is widely used. XGBoost uses a second-order Taylor approximation of the boosting objective and minimizes it similarly to the Newton method. The second-order approximation, together with efficient split finding algorithms and parallel computing, makes XGBoost very fast compared to many other gradient boosting methods [40]. Furthermore, XGBoost uses a regularized learning objective that prevents overly complex models and overfitting [40].

2.4.5 Clustering

Clustering is an unsupervised ML method performed by partitioning the data into subsets based on a distance measure, such that the data is grouped into clusters [41]. The algorithms are either partitional (deciding all clusters at one time instance) or hierarchical (discovering clusters successively based on previous clusters), and common distance measures include the Manhattan distance and Euclidean distance [41]. A frequently used partitional algorithm is K-means which assigns points to the center of the nearest cluster called a centroid. K-means requires the number of clusters K to be set beforehand unlike many other methods. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a hierarchical, density-based clustering model based on two thresholds that must be specified [42]. The thresholds are the minimum number of neighbors within a set distance for a point



(a) K-means clustering using 3 classes.

(b) DBSCAN clustering with eps=0.4 and 5 as the minimum points of a cluster.

Figure 2.5: Examples of clustering algorithms applied on the same data.

to be considered a core point, and the radius of each point to search for neighbors within. An advantage of DBSCAN is that the number of clusters and points that are considered to be noise can be found by the algorithm itself if the chosen parameters are set appropriately. The DBSCAN parameters can however be difficult to tune and greatly affect the results. An example of K-means and DBSCAN is viewed in Figure 2.5.

2.4.6 Principal Component Analysis

Principal Component Analysis (PCA) is a widely used tool that calculates linear combinations of a set of orthogonal vectors to describe complex data in a lower dimensionality [43]. Using PCA and choosing to only include some components allows for dimensional reduction by keeping the most important dimensions, and hopefully excluding the redundant dimensions. As explained in [43], the procedure in PCA is to initially set the first principal component, a vector, to maximize the variance of the data $\mathbf{X}$. This direction is set as the vector $\boldsymbol{p_1}$. The size of $\mathbf{X}$ is $m \times n$ and the following $m-1$ vectors are sequentially set to the orthogonal direction of all previous set vectors that maximize the variance of the data individually. The new representation of the data $\mathbf{Y}$ can be described as $PX = Y$ having $\mathbf{X}$ as the original data and $\mathbf{P}$ as a linear transforming matrix consisting of the vectors $\boldsymbol{p_i}$. The vectors $\boldsymbol{p_i}$ can be judged by their importance according to their representation of the data variance, enabling a metric of how well the components describe the data, $\boldsymbol{p_1}$ describing it the best. Caution needs to be taken since PCA is only applicable to numerical data, is designed for continuous variables, and is sensitive to the scaling of the input data [43]. An example of representing the data with a new set of components using PCA is shown in Figure 2.6.

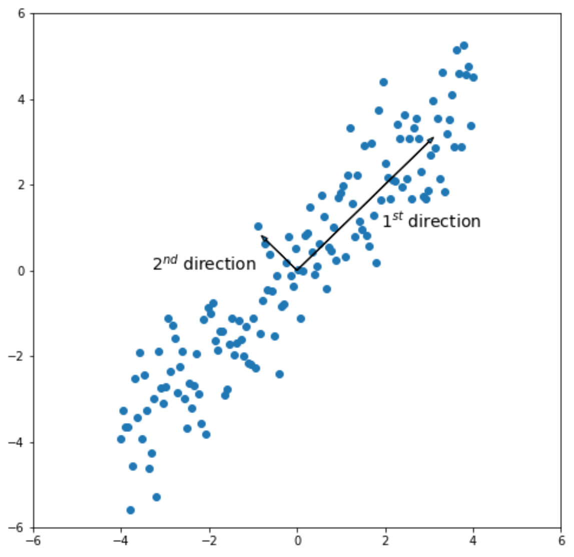


Figure 2.6: Example of two calculated components from a PCA based on the data points colored in blue.

3
Implementation

When using field test data the quality of the data will be heavily influenced by what is measured and the quality of the sensors. Preprocessing is therefore applied before analyzing the data to improve the usability of the data collected. The implementation steps of preprocessing the data and how the analysis of the data has been done are first presented in this section. Thereafter, the procedures for predicting and evaluating the different models are described. The implementation procedure is an iterative process since new insights can be gained from one procedure that influences another one. An example is that knowledge gained from the analysis can provide insight into possible improvements in the preprocessing.

3.1 Preprocessing

Raw data can consist of incorrect entries, missing values, and outliers which can negatively affect learning algorithms. Outliers, also referred to as anomalies, are in this thesis considered observations that appears to be contrary to the rest of the data. They can appear in the field test data due to for example faulty measurements or during test-drives that do not represent realistic driving scenarios. To mitigate the effect, only the preprocessed data was used. The first step is to analyze and process the raw data to deduce what data to keep and how it should be processed. The second step is to divide the data into routes, and lastly, route matching is done with a map provider to add data of the road and environment before finally formatting the data into links.

3.1.1 Raw Data Processing

Initial inspection of raw data gave a good insight into which tools should be used later and descriptive statistics such as the mean, variance, quantiles, and the minimum/maximum values served as a good starting point. Filtering and reasonable saturation limits were set individually for all signals using domain knowledge. The main cleaning operations considered for the raw data were to adjust or fill in missing values and remove unreliable measurements. This could in turn result in the loss of entire routes.

An example of an unreasonable measurement is measuring the weight to be below the minimum possible weight. Saturation limits were assigned to the vehicle weight signal such that the weight never exceeded the maximum specified weight by more than 5 tonnes or fell below the minimum possible weight. Moreover, information on the vehicle type was added instead of the vehicle's ID to make the algorithms more general. For instance, if the vehicle is a distribution truck or a fridge truck. Thereafter, the consumed auxiliary energy was calculated and set as the target variable.

Due to the large spread of values for the auxiliary energy consumption per link, it was decided to predict the average auxiliary power per link instead, as mentioned in Section 1.2. The new target variable was created by dividing the auxiliary energy consumption per link by the time of each link. The resulting target variable has a similar magnitude for all values as a result. To lower the amount of data, the data when the truck was at a standstill were aggregated into single instances of data. When at a standstill the auxiliary energy was aggregated and separated from the data when driving. It was also converted to the average auxiliary power per link. The result was having two different target variables that depend on if the truck is moving or not.

3.1.2 Dividing into Routes

The collection of data could be separated for each vehicle, but there was no predefined signal to distinguish when it was stopping for a longer period or moved without logging data, which could degrade the quality of the data. Therefore it was decided to divide the data into routes, determining that a new route had started if the vehicle had not logged data for longer than 5 minutes before starting to drive as defined in Section 1.3. The choice of 5 minutes as a threshold was based on the assumption that new routes are not created for temporary stops, for instance at junctions, and that the temperature of the cabin and batteries do not change much for such short stops. After longer stops, however, an increased auxiliary power may be required to get them up to operating temperature. This way of dividing the data into routes ensured that the potential increase in power demand generally comes at the beginning of each route when the cabin and batteries are not up to temperature.

After dividing the data into routes, additional cleaning was performed to improve the quality of the vehicle weight signal. The weight signal was noisy and could vary during routes that had no stops, which is unreasonable. It is highly unlikely that a significant change in vehicle weight occurs while driving but the same assumption does not hold when standing still since the vehicle can be loaded/unloaded. The oscillations of the weight were therefore reduced by calculating the average vehicle weight in between stops.

3.1.3 Route Matching and Filtering

The final step of the preprocessing was to match the GPS coordinates of the data with a map provider. The map provider took the data as input, linked the measured position to a road based on GPS coordinates, and returned the data as a sequence of shorter road links. The road links could vary in size but were typically between

5-100 meters. It also returned the road inclination of each link, which was considerably more accurate than the vehicles' estimated inclination. The resulting dataset consisted of the data aggregated into road links with added information from the map provider, assigned to a unique route-ID.

After having all the data divided into routes the final step in the preprocessing was to filter out those routes not deemed valid. The filtering was performed based on the criteria

- Driven distance of at least 2.5 kilometers
- Time while driving at least 10 minutes
- Difference in GPS coordinates corresponding to a Manhattan distance of at least 2.5 kilometers

to exclude short routes and the most unreliable GPS measurements due to the matching of GPS measurements with the map provider.

3.2 Data Analysis

Following the preprocessing of data was the procedure of gaining a better understanding of the data by analyzing it. The two specific procedures used were to see how the data was distributed and the relationships between variables. After gaining new knowledge about the data it could be applicable to go back to preprocessing to modify the dataset used for better performance.

3.2.1 Exploratory Data Analysis

Since there can be hundreds or even thousands of signals available, an initial selection of signals was done by using domain knowledge to eliminate the signals that with high certainty were not useful. Descriptive statistics and histograms were thereafter used to check the distribution of the data. PCA was performed for inspecting the potential of dimensional reduction. As PCA is designed to find the directions that maximize the variances of the data, it does not function well for categorical data. Similarly, many clustering methods tend to function poorly for categorical data that do not have a natural ordering. Therefore, PCA and clustering were only performed on numerical features.

By using clustering the presence of outliers and redundant features could be detected, which gave indications of how to remove or handle them in the preprocessing stage, if possible. The inputs were scaled with min-max scaling before clustering since a distance measure was used. In addition, clustering the data and checking each cluster's distribution of average auxiliary power per link served as an indicator of how well classification is suited to the prediction. In particular, DBSCAN was used to enable outlier detection by a combination of the criteria distance and the minimum number of samples in a cluster. Due to the sensitivity of the DBSCAN parameters, the tuning was very important to get satisfying results. The parameters were set according to the recommendations presented in [42].

3.2.2 Componentwise Energy Consumption

The signals used to calculate the total auxiliary power were the measured time, current, and voltage of the entire low voltage system directly. To validate the use of these signals and gain more domain knowledge, a componentwise analysis of the low voltage system energy consumption was performed. The difference between the summed energy of all measured components and the total was compared, anticipating a relatively small difference due to only measuring the major energy-consuming components.

The componentwise analysis provided domain knowledge of which components have a substantial impact on the total auxiliary energy consumption. This information was helpful to indicate the importance of the measured components. For example, if components related to heating and cooling had a high contribution the ambient temperature is likely an important feature. This knowledge was also used to determine how the individual components' consumption varied between vehicles and seasons, and if some components' consumption can be considered to be constant.

3.2.3 Creation and Transformation of Features

Following the analysis of already existing features was the process of investigating the transformation of features and creating new ones. Only features available for offline predictions were considered and examples of signals added are:

- T_{dev20}: Ambient temperature deviating from 20°C. The signal is calculated as $|T - 20|$ and is used in previous work for predicting the auxiliary power, see Section 1.2 for a more detailed description.
- t_{off_time}: Total time the vehicle has been turned off before starting the route. This signal was created to make assumptions about the temperature in the cabin when starting the vehicle. If the vehicle has been turned off for a long time in very cold/warm climates, the cabin is considered to not be in operating temperature when starting.
- t_{route}: Measures the time the vehicle has been turned on during a route. It was assumed that additional auxiliary power generally is required at the beginning of the route to get the cabin close to the set temperature.
- B_{pre_cond}: A combination of the signals t_{off_time} and t_{route}. The variable is equal to 1 when t_{route} is greater than 10 minutes or t_{off_time} is less than 20 minutes, indicating that the vehicle is up to temperature. The variable is equal to 0 otherwise.
- Month, hour, weekday: Categorical variables obtained by decoding the timestamp signal.

Attention was also given to which signals were chosen and how they were used. For instance, using the ID of the vehicle, GPS coordinates, and sensitive customerspecific information, the data can be coupled to an individual driver, resulting in possible privacy issues. Lastly, transformations were done on the features depending on the data according to Section 2.2.4, since much of the transformation depends on the distribution of data.

3.2.4 Feature Selection

Numerous methods were used to identify and select important features for predicting the average auxiliary power. The thresholds were initially set to relatively low values to ensure that only the features with very low association with the power were filtered out. The values were then increased slightly to filter out more features to reduce the computational complexity. The variables having Pearson's r or Spearman's ρ correlations with the auxiliary power above the threshold of 0.15 were kept. Firstly, filtering was performed with Pearson's r to search for linear relationships. The results could then be compared to Spearman's ρ correlation such that nonlinear relationships also were captured. Additionally, as both Pearson's r and Spearman's ρ are not suitable for capturing correlations of categorical variables, filtering was also performed based on MI and ϕ_K correlation. The threshold decided for MI was set to 0.15, the same as for Pearson's and Spearman's. The ϕ_K metric generally assigned greater values compared to the other methods and was therefore given a threshold of 0.2 to avoid including too many features.

The correlation between all features selected from the filter methods could then be calculated and visualized in a correlation matrix to search for redundant features. That is, if two features were highly correlated, one of them could possibly be removed. However, the final decision regarding redundancy was made based on the wrapper method backward elimination for each model individually.

3.3 Prediction of Auxiliary Energy

How the development of the models and their respective procedures for prediction of the energy was done is described in further detail in this section. The baseline from the current research was set up as (1.1), which is a linear predictor. According to Chapter 2, the training and tuning were done differently depending on the model at hand and some of the models needed additional processing of the input such as numerical encoding and scaling. Common for all models is that they are initially only trained on the data when the vehicle is driving, and then evaluated also on the standstill data independently to check if a separate model for standstill is needed.

The data was divided into sets for training, validation, and testing. The split ratio was initially decided to be 80% for training which is commonly used in ML [30], and 10% respectively for the validation and testing set. The ratio was later modified to ensure that the number of routes in the test set was large enough to get a good estimate of the generalization properties. An alternative is k-fold cross-validation, but since the dataset is large it was not considered. Due to a large correlation of the power between links within the same route the data was split such that all links within a route are only within one of the sets of data. When splitting the data a random seed was set such that all models are always trained and tested on the same data to enable a fair comparison. The test set is unseen data for the models and was only used when the training was complete while the validation set was used during training to determine when to stop. To prevent data leakage from the training set to the validation and test set, the scaling parameters were calculated on the training set only and later applied to the other sets.

The tuning differed greatly between the models as they have different kinds of hyperparameters. Two searches for parameters were done to find satisfactory hyperparameters for the different models. Firstly a grid-search was performed with values spread out to find a good starting point. Thereafter a finer search was done to find values that performed better. For both searches, the Mean Absolute Error (MAE) was set as the evaluation metric.

3.3.1 Polynomial Regression

The Scikit-learn package [44] was used to create and fit polynomial regressors by minimizing the squared error function. The first polynomial considered was a linear regression model with only T_{dev20} as a variable. This polynomial was also considered as the baseline model in this thesis, as it has been used in related work. The basis functions of the baseline, $\boldsymbol{\phi}$, can be written as

$$\boldsymbol{\phi}(T_{dev20}) = \begin{bmatrix}\phi_0, & \phi_1\end{bmatrix} = \begin{bmatrix}1, & T_{dev20}^1\end{bmatrix} \tag{3.1}$$

where $\phi_0 = 1$ is added as a bias term. As the baseline is of very low model complexity, higher-order polynomials were created by adding basis functions of T_{dev20} with an increased exponent to the baseline model. The polynomial order was increased until the loss on the test set stopped decreasing. To investigate the effect of potential outliers in the data, polynomials were also fitted with the Huber loss instead of the squared error loss. The best polynomial for each loss function were compared, and the polynomial with the lowest Mean Absolute Error (MAE) was considered to use the more appropriate loss function. With a suitable degree of polynomial based on T_{dev20} and an appropriate loss function, other features were added to further improve the model's performance.

3.3.2 Multi-Layer Perceptron

MLP was chosen as a model as it is widely used and can fit many different types of functions. Since it can only handle numerical input it was decided to one-hot encode the categorical features and also scale the inputs per Section 2.2.4. One-hot encoding encodes categorical features by adding a binary variable for each category that is possible and setting those variables to either one or zero. Zero corresponds to not having a certain category present, and a one if the category is present. The hyperparameters and how they were set initially for a grid-search to find the best model are shown in Table 3.1, excluding the optimizers' parameters. The optimizers, loss functions, and activation functions used are popular choices applicable for regression

Table 3.1: Hyperparameters and their initial values considered to be tested for an MLP model, excluding the optimizers' hyperparameters.

Hyperparameter	**Value/Method**		
Optimizer	SGD	Adam	
Loss function	Squared	Absolute	Huber
Batch size	32	128	512
Activation function	ReLU	sigmoid	tanh
Hidden layers	1	2	4
Units in each hidden layer	10	40	160
Dropout rate (1-p)	0.1	0.2	0.3

and served as a good starting point. Dropout served as regularization of the network and the choice of optimizer was done empirically as there are several pros and cons to consider. The values were set with some distance between each other such that the search was done coarsely and with incremental gains of the numerical values. The architectures of the models were initially set to be of a small size and were sequentially increased to find when the models' added complexity only gave minor improvements. Only architectures with the same number of units in each layer were tested to limit the number of combinations to test. When the improvement of a larger size dropped to a small percentage compared to the previous size, new sizes were not tested. Considering the accuracy and computational complexity, a model was thereafter selected. The parameters of the optimizer were set to fixed values, but for the final model, a decaying learning rate was used to reach slightly better results.

3.3.3 Tree-based Models

A single regression tree was firstly implemented with Scikit-learn as it is the least complex tree-based model. Features could be used directly as input, without any scaling, since tree-based models generally do not require any transformation of variables. Tuning of the regression tree was performed by changing the depth of the tree while evaluating the loss on the validation set. Moreover, the minimum required number of samples of each leaf node was specified to reduce the risk of overfitting. The tuned regression tree could then be evaluated on the test set and be used as a reference for comparing models with multiple regression trees.

As mentioned in Section 2.4.4, boosting can be used to train an ensemble of regression trees. In this work, two different algorithms were used to develop gradient boosted trees. Initially, gradient boosting similar to the scheme presented in [39] was implemented with Scikit-learn. For convenience, this model will be referred to as regular gradient boosting. Similar to a single regression tree, the boosting models required tuning of the depth of the tree and the minimum required number of samples per leaf node. Additionally, hyperparameters specific to the boosting such as the learning rate and the number of trees used had to be specified. The number of trees is equivalent to the number of training iterations, and there is a trade-off between this hyperparameter and the learning rate. Moreover, the models can be tuned to use a randomly selected subset of the training data to optimize each tree, which reduces the risk of overfitting.

The second algorithm used for applying gradient boosting was XGBoost. The chosen hyperparameters for the previous gradient boosting algorithm served as a starting point for tuning the XGBoost model. The implementation used for XGBoost also provides additional methods for reducing the effect of overfitting. For example, the percentages of randomly sampled features to be used for each tree, level of depth, and node can be specified individually. Each tree can therefore be restricted to only use a subset of input features during training. Thereafter one can further limit the number of available features by sampling for each depth of the tree and finally sampling features for each node. As there were a limited amount of features available, features were only sampled for each tree and no more restrictions were made regarding the levels and nodes. After tuning the XGBoost model, the results were evaluated and compared with the previous gradient boosting algorithm and the single regression tree in terms of accuracy and computational time.

3.3.4 Evaluation

The evaluation was done on the links individually and for entire routes to see how well the models performed at the intended level. For evaluating the models, various metrics were used as they offer different pros and cons. The metrics deemed relevant to use were

- Mean Bias Error (MBE) = $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)$
- Mean Absolute Error (MAE) = $\frac{1}{n}\sum_{i=1}^{n}|(y_i - \hat{y}_i)|$
- Mean Absolute Percentage Error (MAPE) = $\frac{1}{n}\sum_{i=1}^{n}|\frac{(y_i - \hat{y}_i)}{y_i}|$

having y as the target variable and $\hat{y}$ as the predicted value. They provided metrics to get the total difference in power, with and without considering the absolute value to check if it consistently predicts too low or much, as well as the percentage error. For a single link, the only difference between MBE and MAE is the sign, but when aggregating links they can differ more. MAE is always a positive number that will accumulate the error of the links while MBE will evaluate the total bias of the aggregated links. See Figure 3.1 for a comparison of the metrics when the MBE is similar but MAE is significantly different for two different models evaluated on the same data. An accurate model will have a low MBE, but not necessarily for MAE. An accurate and robust model on the other hand has low values for both metrics.

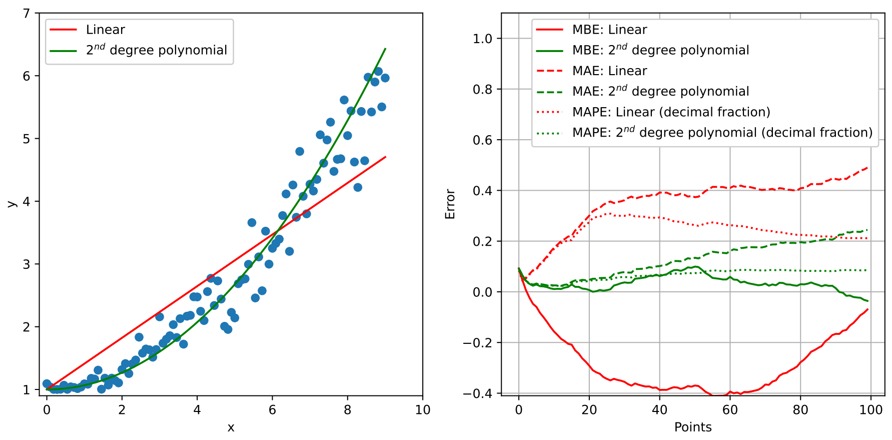


Figure 3.1: Comparison of how MBE, MAE and MAPE differ for two different models when evaluated on the same data, colored in blue. Input is denoted as x and the output as y.

4
Results

The results of the data analysis and predictions using the preprocessed data are presented in this chapter. The predictions of the models are compared with each other using the presented evaluation metrics and it was done both per link and per route. The models were analyzed and trained only on data when driving unless stated otherwise.

4.1 Data Analysis

The average auxiliary power of the links will in this section be presented with different metrics and grouping of the data. Firstly, how the data is distributed, thereafter the analysis of the auxiliary components, and lastly the results from the feature selection.

4.1.1 Data Distribution

The preprocessing of data resulted in about 1 million road links (5873 routes) ready for analysis. Since the predictions are done on links individually the analysis was done on the links as well. The split ratio for the training/validation/test data was set to 70%-10%-20% since 1000 routes in the test set were deemed to be enough to validate the results. A lot of data was available for fridge and refuse trucks but not as much data for distribution trucks. The dataset was therefore imbalanced regarding vehicle classes. Moreover, the data was also somewhat imbalanced in terms of ambient temperature. Figure 4.1 shows that a lot of data is available for temperatures between 0 and 25 °C but little data is available with temperatures outside this span.

To visualize how the average auxiliary power per link, while driving, is distributed a histogram was produced, see Figure 4.2. The power was then also compared with the temperature in a 2D-histogram, see Figure 4.3. It shows that the data has several peaks. Interestingly, there are many values with either relatively high or low auxiliary power for temperatures near 20°C, indicating that temperature alone is insufficient for accurate predictions.

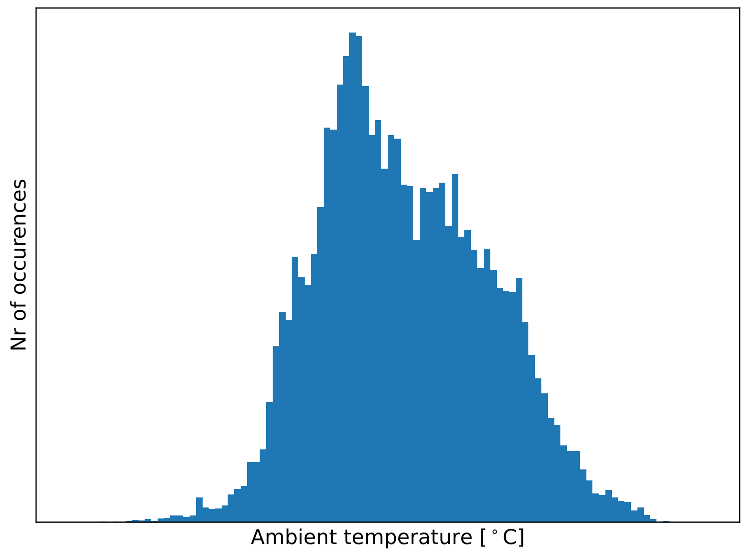


Figure 4.1: Temperature distribution of the links in the data.

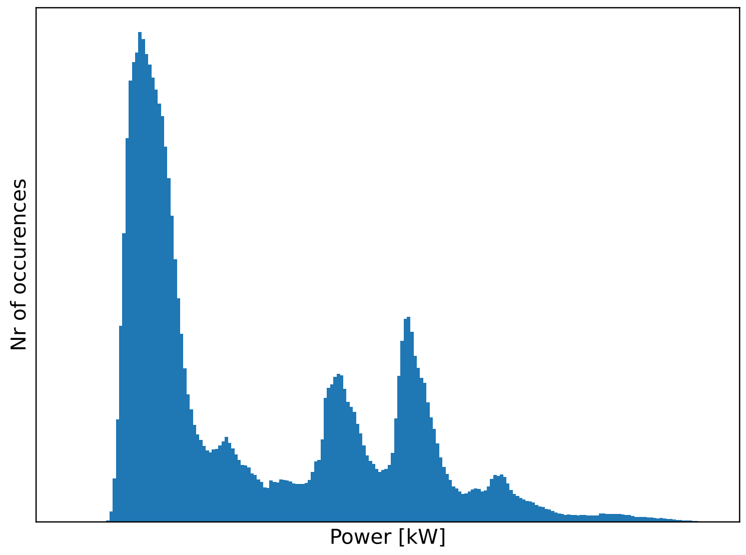


Figure 4.2: Distribution of the average auxiliary power per link, when driving.

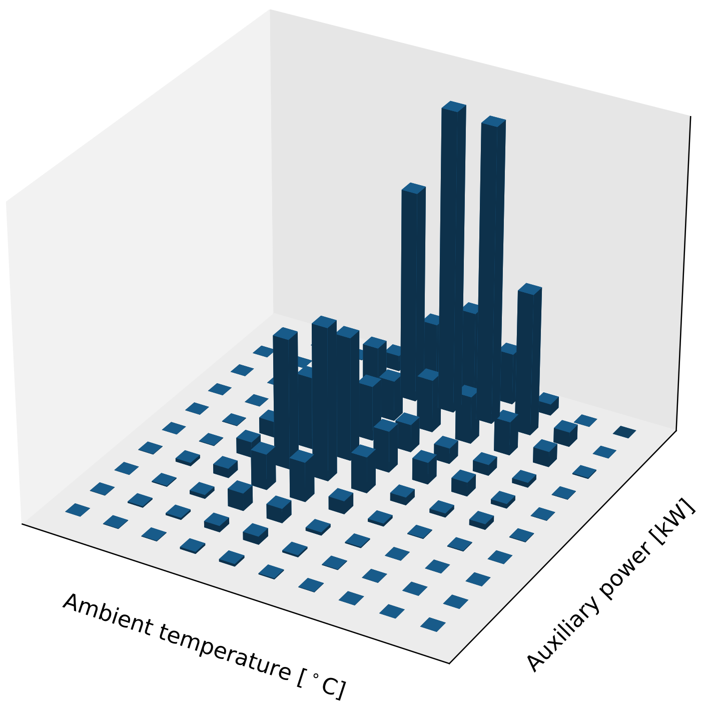


Figure 4.3: Distribution of the temperature and average auxiliary power per link.

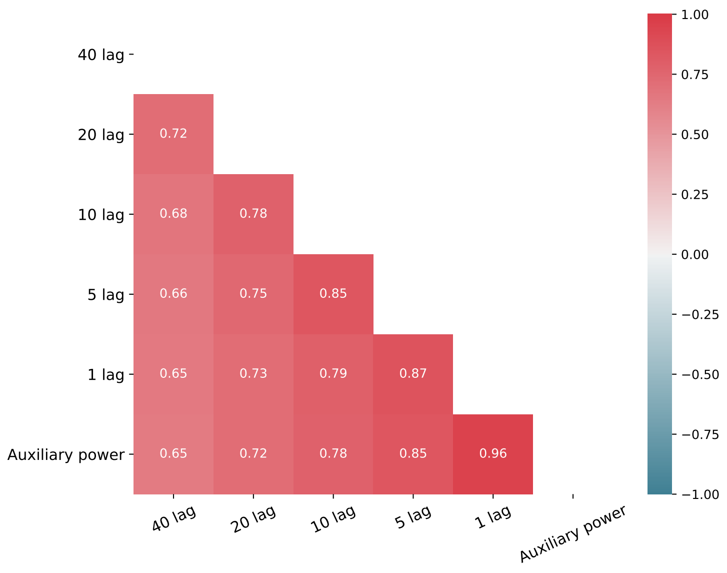


Figure 4.4: Correlation using Pearson's r of different lags of the auxiliary power per link when driving.

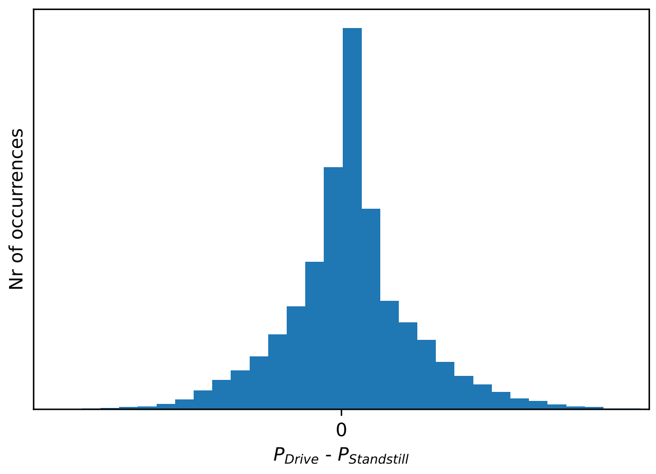


Figure 4.5: Distribution of how the average auxiliary power per link differs when driving or standing still.

To investigate how the auxiliary power changes during driving an analysis was done on how different lags of average auxiliary power per link are correlated with each other, see Figure 4.4. The results indicate a strong correlation of auxiliary power between links meaning that the power is strongly correlated in time since the average time of a link is about 8 seconds.

Since the auxiliary power was separated into variables when it was either at a standstill or driving, the distribution of how they differed was investigated. In Figure 4.5 the difference proved to be relatively small, and most importantly centered about zero. Therefore, it was deemed that the models developed could be applied to both types of target variables without losing too much accuracy.

Performing PCA on the numerical features proved that most of the variance in the data is explained by a smaller subset of features, as seen in Figure 4.6. Using PCA and then clustering with a tuned DBSCAN, provided the number of outliers and location of the clusters, see Figure 4.7. The worst outliers could then be handled in the preprocessing by either removing or correcting data if possible, but not all outliers could be removed.

The clustering and PCA analysis showed that using classification and/or tree-based models could yield decent results, but far from perfect. This was done by analyzing the auxiliary power of the clusters. When using all components, most of the clusters' standard deviation of auxiliary power per link was similar to the standard deviation of the entire dataset. This implies the difficulty of using classification for accurate predictions, as each cluster has a wide variety of possible values of auxiliary power. These results were however only applicable to the numerical features, meaning that some features were excluded.

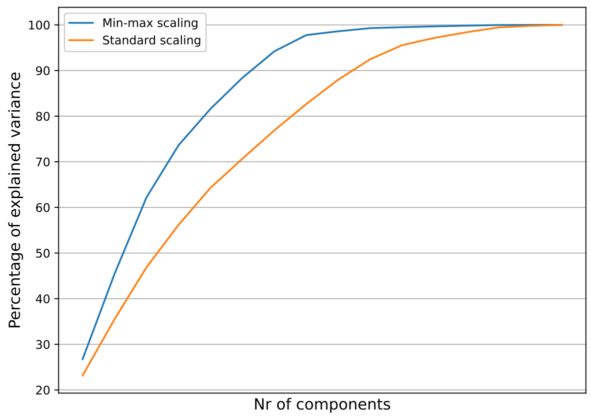


Figure 4.6: Explained variance of components using PCA (on the numerical features), scaling used prior.

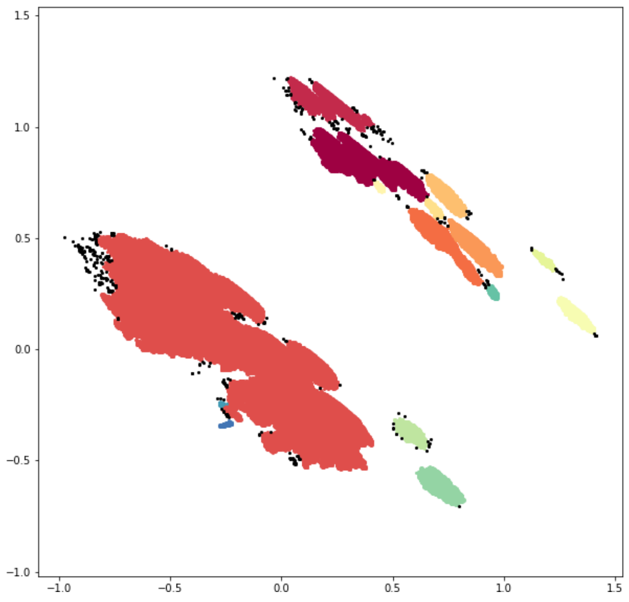


Figure 4.7: DBSCAN clustering using the first two principal components of the numerical features, scaled by min-max scaling.

4.1.2 Componentwise Energy Consumption

In Figure 4.8 the result of aggregating the energy consumption of the auxiliaries for one vehicle is visualized. As anticipated, the total auxiliary consumption of the measured components, the brown bar, is less than the total auxiliary consumption shown in pink. About 85% of the total auxiliary consumption was measured in Figure 4.8 for a refuse truck driving in Gothenburg, Sweden. The data is only from a few chosen days of driving but the results are consistent across vehicles and periods. Typically, signals of all components were not available and the known auxiliary power was close to the actual when the most significant components' power were measured.

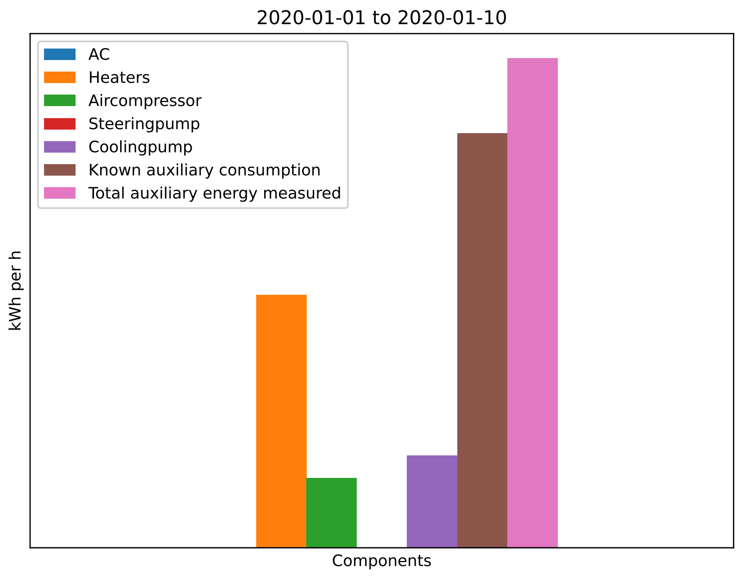
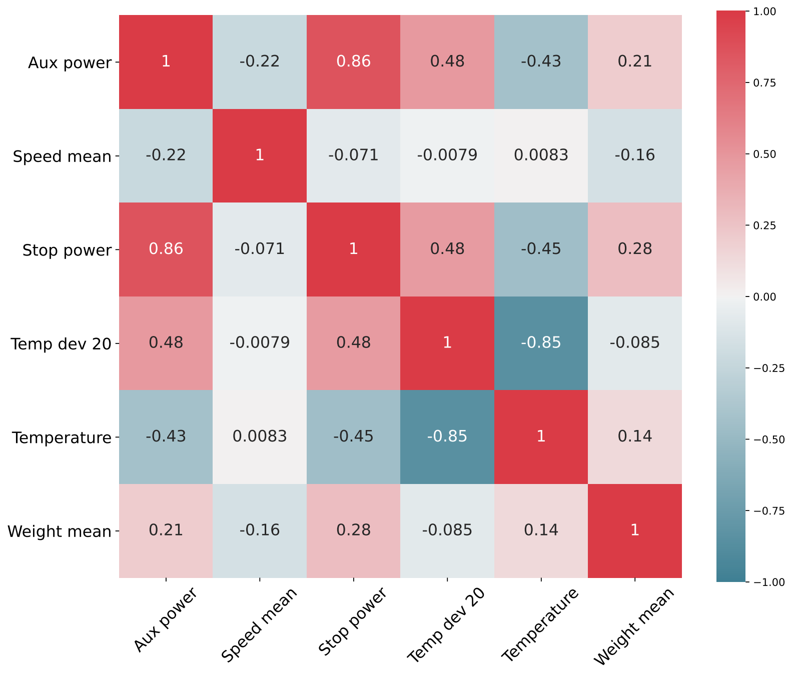


Figure 4.8: Consumption of the major auxiliary components that are most often measured. The vehicle analyzed here is of the type refuse driving in January.

4.1.3 Feature Selection

As indicated by the data analysis, there exist several features adding little to no valuable information to predict the auxiliary power. Therefore, not all features were selected. The initial selection of features was done by filtering based on Pearson's r and Spearman's ρ correlation, the latter viewed as a correlation matrix in Figure 4.9. The correlation matrix shows that there is a high correlation of 0.86 between the auxiliary power per link during stops and driving. Moreover, the auxiliary power per link when driving and at a standstill have a similar association with the other variables in the correlation matrix, except for the average vehicle speed. The correlation matrix also shows that the ambient temperature and T_{dev20} are the most important features. Using the temperature deviating from 20°C instead of the actual temperature thus yields an increased correlation with the auxiliary power. Similar results were given when Pearson's r correlation was used instead, indicating that the relationships between features are fairly linear.



	Aux power	Speed mean	Stop power	Temp dev 20	Temperature	Weight mean
Aux power	1	-0.22	0.86	0.48	-0.43	0.21
Speed mean	-0.22	1	-0.071	-0.0079	0.0083	-0.16
Stop power	0.86	-0.071	1	0.48	-0.45	0.28
Temp dev 20	0.48	-0.0079	0.48	1	-0.85	-0.085
Temperature	-0.43	0.0083	-0.45	-0.85	1	0.14
Weight mean	0.21	-0.16	0.28	-0.085	0.14	1

Figure 4.9: Correlation matrix viewing important correlations with the auxiliary power per link based on Spearman's ρ.

A different set of features was selected when MI was used instead of correlations, see Figure 4.10. According to MI, the vehicle weight is the most important feature for predicting the auxiliary power. Other relevant features based on MI are the GPS position and t_{off_time}, which all had a low correlation in Pearson's r and Spearman's ρ correlation. The two signals of temperature are also considered to be important with MI, but not as important as in the correlation methods. Additionally, the results in Figure 4.10 show that categorical features such as the month, hour of the day, and vehicle type are among the top features regarding MI with the auxiliary power. The vehicle type signal provides information on whether the vehicle is a refuse, fridge, or distribution truck.

The final method used for selecting features was the ϕ_K correlation. As explained in section 2.2.1, ϕ_K can be used for all variable types and the resulting features when performing filtering can be seen in Figure 4.11. Filtering with the ϕ_K metric gives a similar set of features as in Spearman's correlation and MI combined. Two features with much higher importance compared to previous methods are the vehicle-ID and vehicle type, with correlations of 0.51 and 0.44 with the auxiliary power respectively. The vehicle-ID is the name of each vehicle, which cannot be used for prediction as there will be no training data available for newly produced vehicles. However, the

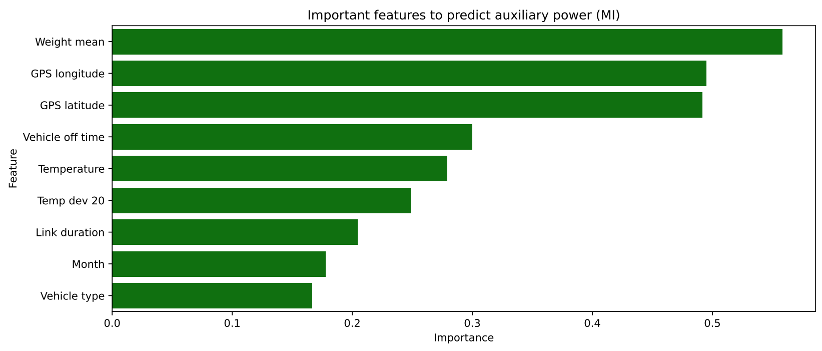


Figure 4.10: Bar plot of features with MI above 0.15 with the auxiliary power.

information on vehicle type is available when making predictions and can thus be used as a feature. Another interesting result in Figure 4.11 is that some features with a high correlation with the auxiliary power also have a high correlation with the temperature. Examples of such features are the month and vehicle type. The month may be a useful feature as it provides information on the ambient temperature, but knowing the ambient temperature might make the month a redundant feature.

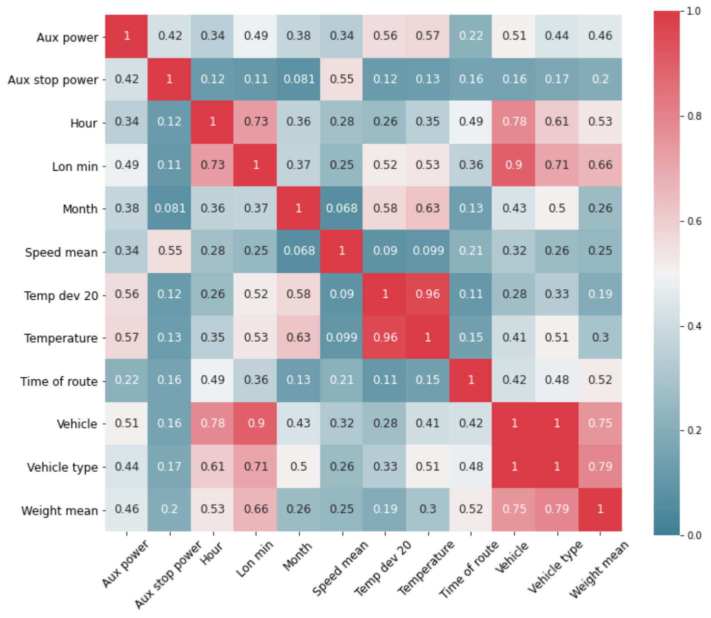


Figure 4.11: Correlations with auxiliary power above the threshold 0.2 using the metric ϕ_k.

Filtering based on Pearson's r, Spearman's ρ, MI, and ϕ_K resulted in 12 features being selected. All features were initially kept for developing models, although some features were suspected to be redundant. A wrapper method was therefore used to determine the redundancy of features and which features to discard. The full list of selected features can be seen in Table 4.1. Worth noting is that B_{pre_cond} is the only created feature that was not selected in the end.

Table 4.1: Resulting features from filtering and the corresponding criteria each feature was selected by.

Feature	Method(s) selected by
T	all
T_{dev20}	all
Weight	all
Speed	ρ, ϕ_K
GPS longitude	MI, ϕ_K
GPS latitude	MI, ϕ_K
Month	MI, ϕ_K
Vehicle type	MI, ϕ_K
t_{off_time}	MI
t_{link}	MI
t_{route}	ϕ_K
Hour	ϕ_K

The association of different powers of T_{dev20} with the auxiliary power was also analyzed. Since the feature is monotonically increasing, Spearman's ρ gives a correlation of 1 for all powers. An important takeaway when checking Pearson's r, ϕ_K and MI was that T_{dev20} and T_{dev20}^2 were very similar to each other for the mentioned metrics, and that increasing the power of T_{dev20} decreased the association with the auxiliary power. As a result, one should consider adding powers of T_{dev20} as an input to the models but making sure to perform the wrapper method to not include redundant features. For the tree-based models however, including powers of T_{dev20} does not add any new information due to the nature of the model.

4.2 Power Prediction per Link

The results of the average auxiliary power predictions per link are presented in this section for the various models used. As explained in Section 3.3, the models were evaluated on a test set which is unseen data for the models. Moreover, a random seed was used when dividing the data into sets such that all models were trained and evaluated on the same data. The evaluation is based on several metrics presented in Section 3.3.4, but MAE is considered to be the most important metric when predicting the links. This is due to MAE evaluating the total absolute deviation from the target variable, whereas MAPE quantifies the relative error. That is, using MAE ensures that the accuracy is not evaluated based on the size of the target variable. The training and prediction of all models were done with a laptop having commonly used specifications.

4.2.1 Polynomial Regression

The first polynomial regression models considered were polynomials using T_{dev20} as a variable and fitted by minimizing the squared error loss. Table 4.2 views the result of evaluating polynomials of different degrees on all links in the test set based on the metrics MAE and MAPE. As can be seen in the table, the performance is improved when the polynomial degree of the baseline is increased. A polynomial degree of 4 is considered the best, as it achieves the smallest error and higher degrees gave insignificant improvements. The effect of outliers was investigated by fitting the same polynomials with the Huber loss instead of the squared error loss. Table 4.3 views the result of using the Huber loss, which shows that a polynomial degree of 4 or 5 are appropriate model complexities. Moreover, using the Huber loss gives a decrease in MAE but an increase in MAPE. This means that the polynomials fitted with the Huber loss generally deviates less from the target value, but still has a higher relative error. The relative error favors models that are more accurate on links with low power, as the deviation is scaled with the target value. Hence, the polynomials using the squared error loss tend to perform better on links with low power, while it deviates more on the links with high power. As MAE is considered to be the most important metric for predicting links, the Huber loss is deemed the more appropriate loss function.

Table 4.2: Performance of polynomials of various degrees with T_{dev20} as the variable, fitted with the squared error loss. The MAE given by the baseline model is used as reference for the reduction.

Polynomial degree	MAE reduction [%]	MAPE [%]
1^{st} (baseline)	Ref	45.2
2^{nd}	0	45.3
3^{rd}	0.5	45.2
4^{th}	1.5	44.9
5^{th}	1.5	44.9

As found in Section 4.1.3, several other features have a meaningful association with the average auxiliary power per link. Backward elimination was therefore used to add more features to the best performing polynomial with T_{dev20}. The resulting polynomial was a third-degree polynomial with T_{dev20}, and with the vehicle speed and weight as additional features. The performance of the final polynomial is visualized in Table 4.4.

Table 4.3: Performance of polynomials of various degrees with T_{dev20} as the variable, fitted with the Huber loss. The MAE given by the baseline model is used as reference for the reduction.

Polynomial degree	MAE reduction [%]	MAPE [%]
1^{st}	2.0	47.7
2^{nd}	3.0	47.1
3^{rd}	5.5	46.7
4^{th}	6.5	48.1
5^{th}	7.5	49.0
6^{th}	7.0	48.3

Table 4.4: Performance of the best polynomial using T_{dev20}, the vehicle speed and weight, fitted with the Huber loss. The MAE given by the baseline model is used as reference for the reduction.

MAE reduction [%]	MAPE [%]
10.5	47.2

4.2.2 Multi-Layer Perceptron

The MLP model was decided to be of a simple architecture that can easily be extended to include the Bayesian approximation method as explained in Section 2.4.2. Therefore, dropout was introduced as the regularizer and p was initially set to 0.2 according to the recommendations. Hidden layer depths of $[1, 2, 3]$ in combination with hidden units ranging from $5 \times 2^{j}_{j=0,\dots,5}$ were initially tested. The size of the network is in the thesis denoted as *hidden layers* $\times$ *hidden units per layer* since all variants will have an output layer consisting of one unit for the regression problem. The features used when determining the size were the ones in Table 4.1.

The evaluation of some different architectures for the MLP model is summarized in Table 4.5 using the same hyperparameters. The results indicate that increasing the depth and number of hidden units improves the accuracy, but the improvement rate is low after reaching the size of 3×100. A size of 3×160 only gave a small improvement compared to 3×100. Therefore, 3×100 was chosen as the size of the model. Using a size of 3×100 compared to 3×160 results in having 25301 trainable parameters instead of 59681.

Table 4.5: Performance of varying MLP architectures when predicting links in the test set. The MAE given by the baseline model is used as reference for the reduction.

Size of MLP	MAE reduction [%]	MAPE [%]
3×80	20.1	35.5
3×100	22.1	34.6
3×160	22.5	34.5

Table 4.6: Values of the hyperparameters used for the final MLP model, excluding the optimizers' hyperparameters. ReLU and dropout were applied between each layer, always using the same dropout rate.

Hyperparameter	**Method/Value**
Optimizer	Adam
Loss function	Huber
Batch size	32
Activation function	ReLU
Dropout rate (1-p)	0.2

Having determined the size of the MLP, the final hyperparameters that performed the best according to empirical testing are shown in Table 4.6. Adam was selected as the optimizer as it converged faster than the SGD and made the tuning of hyperparameters easier. The hyperparameters of Adam were initially set to the standard values of it and then modified by lowering the learning rate. Using the absolute loss or the Huber loss achieved similar performance, both better than the squared error loss. However, the Huber loss was used for the final MLP model to ensure that a similar loss function is used as for the other models. ReLU was chosen as the activation function as it gave the best results. Choosing a smaller batch size enhanced the training and the accuracy on the training set when lowering it from 512 until 32, after which the test accuracy decreased.

Finally, the backward elimination method was performed to determine the final features of the MLP. Powers of T_{dev20} were also tested along with the features in Table 4.1. The features used in the final model were all features in the table with an added term of T_{dev20}^2 which gave a final performance as shown in 4.7. The presented MLP model took about 30 minutes to train.

Table 4.7: Performance of the final MLP architectures when predicting links in the test set. The MAE given by the baseline model is used as reference for the reduction.

Size of MLP	MAE reduction [%]	MAPE [%]
3×100	24.6	33.5

4.2.3 Tree-based Models

As the tree-based models are optimized with greedy optimization methods, all features in Table 4.1 were used for developing the models. The performance of a single regression tree for different tree depths is shown in Table 4.8. A suitable choice of tree depth for a single regression tree proved to be 11, as this setting achieved the lowest loss on all links in the test set. Furthermore, the model was less prone to overfit the data when the minimum required number of samples per leaf node was increased. Appropriate values of this parameter were approximately 20-40 samples as other values gave a higher loss.

Table 4.8: Performance of a single regression tree for various depths, trained with the squared error loss. The MAE given by the baseline model is used as reference for the reduction.

Depth	MAE reduction [%]	MAPE [%]
8	11.5	55.0
9	12.5	54.4
10	13.5	53.4
11	14.0	53.7
12	13.0	53.7

Increasing the model complexity by using boosting algorithms gave a significant improvement compared to a single regression tree, see Table 4.9. XGBoost has better performance than regular gradient boosting in terms of MAE and MAPE and there is a big difference regarding the computational time as XGBoost is about 10 times faster during training. XGBoost is considered to be the better choice of boosting algorithm, and will therefore be the only tree-based model considered for the remainder of this thesis. The results in Table 4.9 also show that using the Huber or Pseudo-Huber loss gives better accuracy compared to using the squared error loss.

Table 4.9: Performance of gradient boosting and XGBoost on all links in the test set, using different loss functions. The MAE given by the baseline model is used as reference for the reduction.

Algorithm	Loss	MAE reduction [%]	MAPE [%]
Regular	Squared	18.5	49.8
XGBoost	Squared	19.0	50.2
Regular	Huber	20.5	48.0
XGBoost	Pseudo-Huber	15.0	42.9

Table 4.10 views the hyperparameters used for the XGBoost model that achieved the lowest loss on the test set. The model took about 1-2 minutes to train with this configuration of hyperparameters. Early stopping was used such that the training is halted if 10 trees have been added without a reduction in validation loss. This usually occurs at around 550-600 trees, which means that the number of trees specified as a hyperparameter generally is not equal to the actual number of trees. The best choice of depth of each tree was found to be 8 and therefore not as deep as the single regression tree. Furthermore, using a subset of training samples and features for each tree proved to reduce overfitting and increase the model performance. Finally, backward elimination was used to exclude redundant features, which resulted in T_{dev20} and t_{link} being removed without affecting the model performance.

Table 4.10: Fine-tuned hyperparameters used in XGBoost.

Hyperparameter	XGBoost
Learning rate	0.0075
Number of trees	1000
Max depth	8
Minimum samples per leaf node	35
Percentage of training samples per tree	75
Percentage of features per tree	75

4.2.4 Comparison of Predictions per Link

In Table 4.11, the best polynomial regression, MLP, and tree-based models are compared when evaluated on all links in the test set. As can be seen in the table, the MLP and XGBoost model achieve very similar performance in terms of MAE, but the MLP has a quite much lower MAPE. Referring back to when comparing polynomials with the squared and Huber loss in section 4.2.1, the MLP thus generally performs better on links with low power and worse on links with high power, compared to XGBoost. The polynomial regression model is unable to match the performance of the other two models, but it is also the least complex model among the three.

Table 4.11: Results of predicting the power of each link in the test set using the best performing models. The MAE given by the baseline model is used as reference for the reduction.

Model	MAE reduction [%]	MAPE [%]
Polynomial	10.5	47.2
MLP	24.6	33.5
XGBoost	25.0	42.9

4.2.5 Comparison of Predictions per Stop

As the auxiliary power proved to be fairly similar when driving or standing still, the best model of each model type was also evaluated on the auxiliary power when at a standstill. The result is visualized in Table 4.12, which shows that all models have an increased MAE and MAPE compared to their performance on the auxiliaries while driving in Table 4.11. This is reasonable as the models were trained to predict the auxiliary consumption while driving. Separate models for the auxiliaries when at a standstill were therefore created by training the same models but with the auxiliary power when standing still as the target variable. The result can be seen in Table 4.13, which shows a slight improvement for all models. Although the models improve when trained on the auxiliaries at a standstill, there is still a quite large performance gap when comparing results between driving and standing still. This is probably due to the limited amount of training data for the auxiliary power during stops, as approximately 8% of the links in the training data contain a stop.

Table 4.12: Results of predicting the auxiliary power of each stop in the test set using the best performing models trained on data when driving. The MAE increase is relative to the models' own performance on the auxiliaries while driving.

Model	MAE increase [%]	MAPE [%]
Polynomial	15.6	52.8
MLP	35.1	38.7
XGBoost	13.4	52.1

Table 4.13: Results of predicting the auxiliary power of each stop in the test set using the best performing models trained on data when at a standstill. The comparison with the baseline is when it has been trained on the standstill data, and the drive reduction is relative to the models' own performance on the standstill data when trained on data when driving.

Model	MAE reduction (Baseline) [%]	MAE reduction (Drive) [%]	MAPE [%]
Polynomial	6.9	1.9	52.6
MLP	11.6	5.3	36.7
XGBoost	16.5	3.2	45.8

4.3 Energy Prediction per Route

In this section, the predictions are done on entire routes made up of sequences of links. The predictions are still the power of each link, but the final prediction is the energy consumption of routes. The energy is calculated by multiplying the power and time of each link, and then aggregating them into routes. To get a better understanding of how the auxiliary power changes during a route, the power of the routes' links were plotted over time. An example of some difficult routes to predict the power of are shown in Figure 4.12. As can be seen, the power can vary a lot, while the temperature, which is the most important feature, can be close to constant. This is also the case with many of the other important features used in the models, and it is therefore difficult for the models to predict sudden power spikes similar to those viewed in the routes.

When predicting routes it was decided to use MAPE as the deciding evaluation metric to retrieve a relative error of the energy predictions of entire routes. Using MAE would favor the algorithms predicting the total absolute amount of energy the best instead of each route individually.

4.3.1 Comparison of Predictions per Route

In Table 4.14 the best results from the different models are compared to each other when predicting the total energy of the routes in the test set. The MLP and XGBoost outperform the baseline and polynomial with a large margin and have a similar accuracy when compared to each other. XGBoost is the best performing model for routes despite being worse than the MLP regarding MAPE and similar in terms of MAE when evaluated on links. The underlying reasons for this could be the

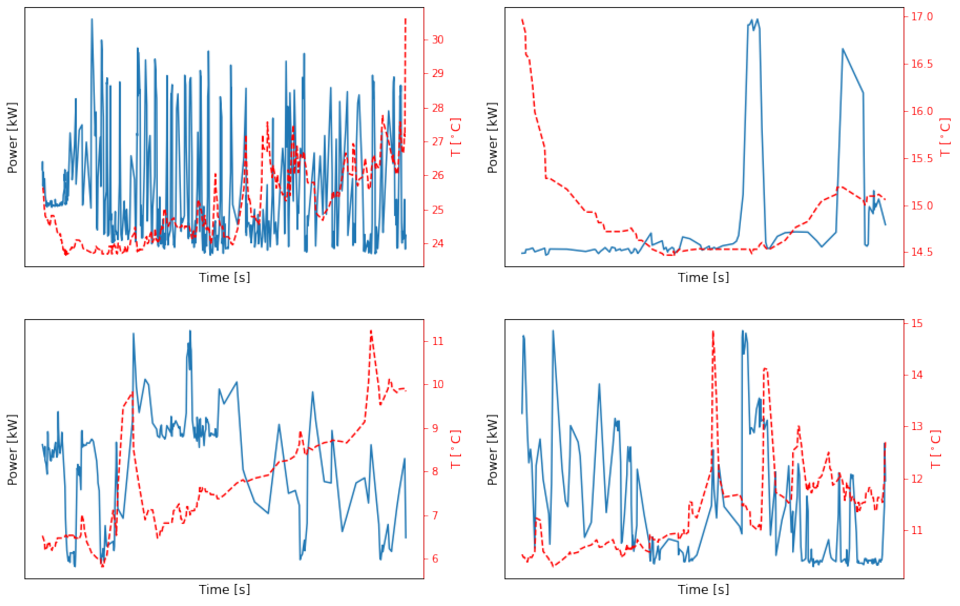


Figure 4.12: Links' auxiliary power and ambient temperature aggregated over time for different routes. The blue and red line corresponds to the auxiliary power and ambient temperature, respectively.

slightly lower MAE and MBE per link of the XGBoost compared with the MLP. The computational complexity of them differs, but all models predicted the entire test dataset within 10 seconds when using a laptop with commonly used specifications. The polynomial is the quickest to evaluate and the MLP is the slowest.

Table 4.14: Results of predicting the total energy of the routes while driving in the test set.

Model	MAPE [%]
Baseline	40.4
Polynomial	31.5
MLP	22.9
XGBoost	22.0

In Figure 4.13, the baseline, best polynomial, MLP, and XGBoost models are evaluated on four randomly selected routes. The results in the figure accurately reflect the models' performance when evaluated on all links and routes (Table 4.11 and 4.14). The MLP and XGBoost models predict with similar accuracy and are generally quite close to the target energy. The two models also consistently outperform the baseline and the polynomial model, which shows that the given problem of predicting the auxiliaries requires a model of higher complexity.

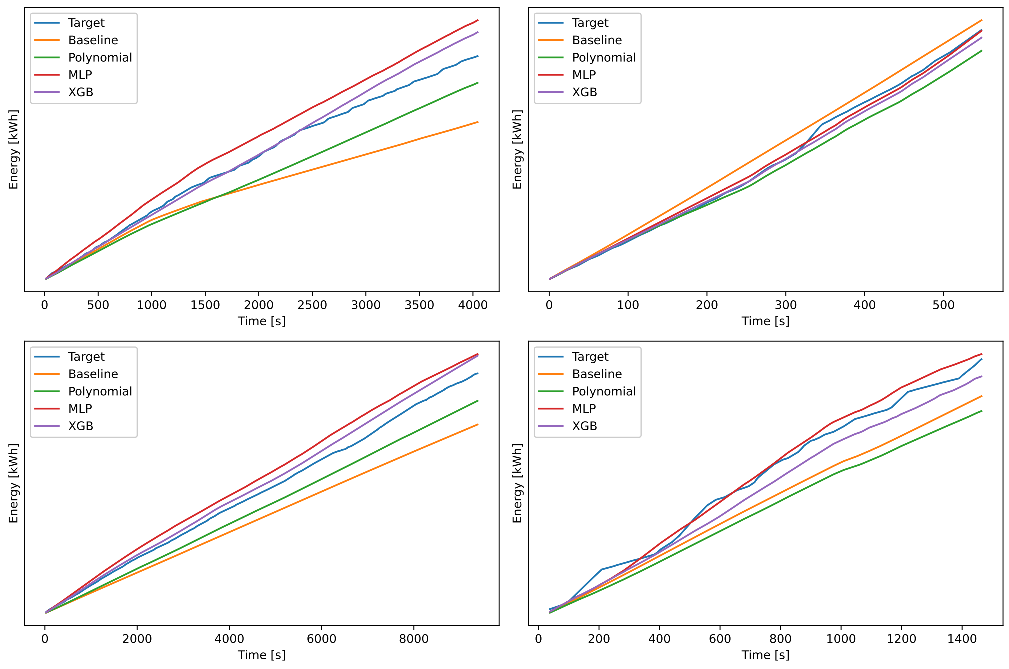


Figure 4.13: Results when predicting four randomly selected routes with the best models.

4.3.2 Uncertainty Estimates using Dropout

Selecting the same four random routes as in Figure 4.13, the resulting predictions of the MLP with uncertainty included are visualized in Figure 4.14. The uncertainty is provided by using dropout as explained in Section 2.4.2. The uncertainty grows cumulatively and typically includes the target within the interval even when the predictions are not very accurate. An interesting property is that for the routes plotted, the uncertainty seems to increase more when a sudden change of the target energy appears.

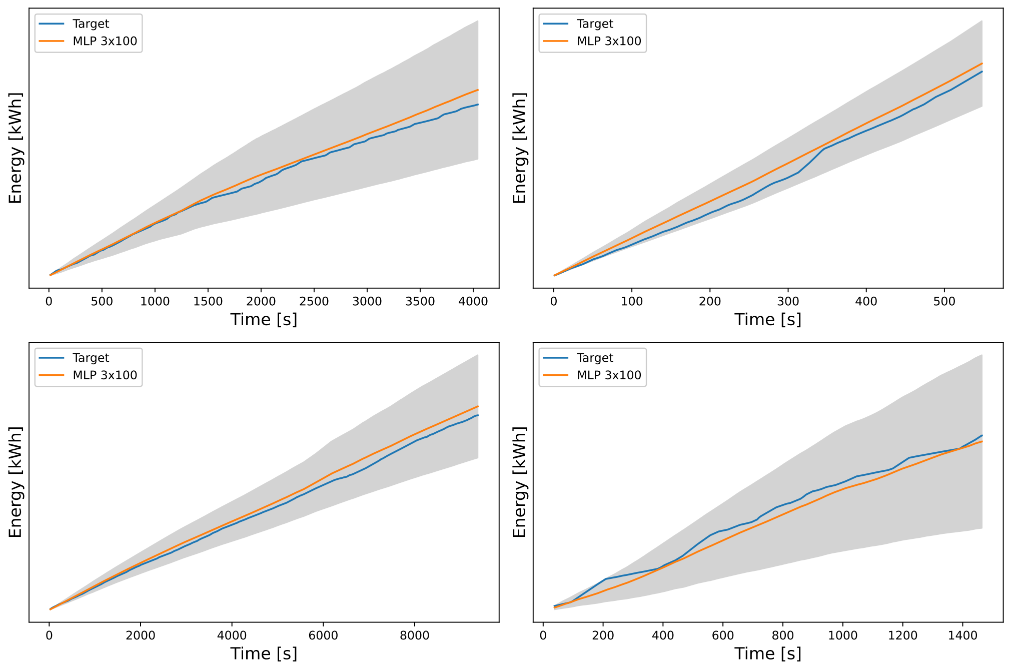


Figure 4.14: Results when predicting the same four routes as in Figure 4.13 with an MLP of size 3×100 trained with uncertainty estimates included. The shaded area in gray represents ± 2 standard deviations.

5 Conclusion

The thesis has successfully implemented a pipeline to process field test data and perform energy predictions of the auxiliaries with great improvements compared to other algorithms suggested in the literature, when applied to the data provided. All models outperformed the baseline, revealing the lack of research that has been invested in predicting electric trucks' auxiliaries. As presented by the association of variables and the large difference in power during a route, it is very difficult to accurately predict the energy consumption of auxiliaries for entire routes. The temperature proved to be the most important feature but is by itself far from enough to enable accurate predictions. Therefore, the algorithms found in the related work are significantly worse than the models developed in this thesis. Models with low complexity such as polynomial regression proved to be far too simple to accurately predict the auxiliary energy. Using more features and ML models with higher complexity fared well considering the problem at hand and the large amount of data. The results are however very dependent on preprocessing of data to ensure that they learn correctly.

This work has found data preprocessing to be a very significant step to enable accurate predictions. During the implementation, the preprocessing steps were updated in an iterative process coupled with the analysis after gaining new knowledge about the data. There still seem to be improvements that can be done on the preprocessing when analyzing the results, but it is difficult to determine the significance since the errors of the links accumulate when predicting routes. For instance, adding more features could be an important step to improving the predictions. The auxiliary power during driving is expected to be large initially when it is very cold outside, but no robust feature has been found to handle if the truck has been preconditioned or not, despite creating new features that try to tackle the problem. The desired temperature or range of temperature is a variable for both the batteries and the cabin, which might be helpful. It has a large effect on the HVAC system and therefore the total auxiliary energy consumption.

The MLP and XGBoost predicted the auxiliary power while driving with relatively high accuracy. However, using the same models to predict when at a standstill resulted in a quite large decrease in performance. Separate models trained specifically on auxiliaries when standing still were thus created to reduce the performance gap between driving and standstill, but with limited success. In conclusion, the XGBoost model is deemed the most appropriate model when evaluated on entire routes, although similar results were given with the MLP. What makes XGBoost favorable other than the lowest MAPE is that it is easier to implement, as it is robust to scaling of features and has fewer hyperparameters than the MLP. Additionally, the Huber loss is considered to be a suitable choice of loss function as it provides robustness to outliers.

The approach of adding the uncertainty of the predictions using dropout as a Bayesian approach seems to work decently in many cases, but the uncertainty region is large and the approach should be investigated further. One reason why the regions can be very large is that a few of the features are very important, in particular the temperature-related features. Not including them makes it very difficult to predict accurately.

An important note is that the inputs to the model are actual values measured during driving. When predicting the energy in a real-life application ahead of time, the input variables such as the temperature and time are also predicted beforehand. The energy predictions' accuracy will therefore be dependent on how accurate other predictions are, for instance, the vehicle routing and weather forecasts. It would therefore be interesting to test the performance in such a scenario.

5.1 Further Work

The thesis has accomplished improved energy predictions of auxiliaries but has also resulted in recommendations on how to further improve the accuracy. It includes which data is collected and used, processing of the data, and predictions using different models. Many aspects can be further analyzed and the most significant areas of future work we consider are

- Including more features, such as the weather conditions, features representing a built-in memory of states, and the preferred temperature of the cabin and batteries.
- Testing new models applicable for the regression problem and further optimizing the currently tested ones.
- Stating the problem differently. Testing models when assessing the problem as a time-series prediction is interesting since the power of the links have a very large correlation with each other within a route. For instance, a time-series recurrent neural network. One could also try to predict the energy of each link instead of the power.

Allocating the time to further investigate the mentioned areas is believed to increase the accuracy, but how much is difficult to tell. The problem at hand is difficult and the models developed need to be further tested on new data to validate their performance. Another interesting topic to investigate is the approach of updating the predictions during the route in real-time to produce more accurate results. Since the auxiliary power is very correlated in time it is believed that significant improvements can be made.

Bibliography

[1] ACEA, "Fact sheet: trucks," Available at https://www.acea.auto/files/trucks_fact_sheet_ACEA.pdf (2022/04/15).

[2] ACEA, "Acea report - vehicles in use, europe 2022," Available at https://www.acea.auto/files/ACEA-report-vehicles-in-use-europe-2022.pdf (2022/04/15).

[3] McKinsey Company, "Why most eTrucks will choose overnight charging," Available at https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/why-most-etrucks-will-choose-overnight-charging (2022/04/21).

[4] T. R. Hawkins, B. Singh, G. Majeau-Bettez, and A. H. Strømman, "Comparative environmental life cycle assessment of conventional and electric vehicles," *Journal of Industrial Ecology*, vol. 17, no. 1, pp. 53–64, 2013. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1530-9290.2012.00532.x

[5] European Environment Agency, "Trends and projections in Europe 2021," 2021. [Online]. Available: https://www.eea.europa.eu/publications/trends-and-projections-in-europe-2021

[6] H. Liimatainen, O. van Vliet, and D. Aplyn, "The potential of electric trucks – an international commodity-level analysis," *Applied Energy*, vol. 236, pp. 804–814, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261918318361

[7] EUR-Lex, "REGULATION (EC) No 561/2006 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL," Available at https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:02006R0561-20200820 (2022/04/21).

[8] C. Fiori, K. Ahn, and H. A. Rakha, "Power-based electric vehicle energy consumption model: Model development and validation," *Applied Energy*, vol. 168, pp. 257–268, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030626191630085X

[9] R. Basso, B. Kulcsár, and I. Sanchez-Diaz, "Electric vehicle routing problem with machine learning for energy prediction," *Transportation Research Part B: Methodological*, vol. 145, pp. 24–55, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191261520304549

[10] R. Basso, B. Kulcsár, B. Egardt, P. Lindroth, and I. Sanchez-Diaz, "Energy consumption estimation integrated into the electric vehicle routing problem," *Transportation Research Part D: Transport and Environment*, vol. 69, pp. 141–167, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361920918304760

[11] M. Dollinger and G. Fischerauer, "Model-based range prediction for electric cars and trucks under real-world conditions," *Energies*, vol. 14, no. 18, 2021. [Online]. Available: https://www.mdpi.com/1996-1073/14/18/5804

[12] J. bo Wang, K. Liu, T. Yamamoto, and T. Morikawa, "Improving estimation accuracy for electric vehicle energy consumption considering the effects of ambient temperature," *Energy Procedia*, vol. 105, pp. 2904–2909, 2017, 8th International Conference on Applied Energy, ICAE2016, 8-11 October 2016, Beijing, China. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1876610217307099

[13] C. De Cauwer, J. Van Mierlo, and T. Coosemans, "Energy consumption prediction for electric vehicles based on real-world data," *Energies*, vol. 8, no. 8, pp. 8573–8593, 2015. [Online]. Available: https://www.mdpi.com/1996-1073/8/8/8573

[14] I. Sagaama, A. Kchiche, W. Trojet, and F. Kamoun, "Proposal of More Accurate Energy Model of Electric Vehicle For SUMO," in *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, 2018, pp. 464–469.

[15] F. Morlock, B. Rolle, M. Bauer, and O. Sawodny, "Forecasts of electric vehicle energy consumption based on characteristic speed profiles and real-time traffic data," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 1404–1418, 2020.

[16] J. Lindgren and P. D. Lund, "Effect of extreme temperatures on battery charging and performance of electric vehicles," *Journal of Power Sources*, vol. 328, pp. 37–45, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378775316308941

[17] K. N. Genikomsakis and G. Mitrentsis, "A computationally efficient simulation model for estimating energy consumption of electric vehicles in the context of route planning applications," *Transportation Research Part D: Transport and Environment*, vol. 50, pp. 98–118, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361920915302881

[18] P. Iora and L. Tribioli, "Effect of ambient temperature on electric vehicles' energy consumption and range: Model definition and sensitivity analysis based on nissan leaf data," *World Electric Vehicle Journal*, vol. 10, no. 1, 2019. [Online]. Available: https://www.mdpi.com/2032-6653/10/1/2

[19] K. R. Kambly and T. H. Bradley, "Estimating the hvac energy consumption of plug-in electric vehicles," *Journal of Power Sources*, vol. 259, pp. 117–124, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S037877531400216X

[20] R. A. Barnitt, A. D. Brooker, L. Ramroth, J. Rugh, and K. A. Smith, "Analysis of off-board powered thermal preconditioning in electric drive vehicles: Preprint." [Online]. Available: https://www.osti.gov/biblio/1001443

[21] K. Kambly and T. H. Bradley, "Geographical and temporal differences in electric vehicle range due to cabin conditioning energy consumption," *Journal of Power Sources*, vol. 275, pp. 468–475, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378775314017613

[22] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Computers Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014, 40th-year commemorative issue. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790613003066

[23] J. Cai, J. Luo, S. Wang, and S. Yang, “Feature selection in machine learning: A new perspective,” *Neurocomputing*, vol. 300, pp. 70–79, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231218302911

[24] M. A. Hall, “Correlation-based feature selection for discrete and numeric class machine learning,” in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. ICML ’00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, p. 359–366.

[25] C. M. Bishop *et al.*, *Neural networks for pattern recognition.* Oxford university press, 1995.

[26] M. Baak, R. Koopman, H. Snoek, and S. Klous, “A new correlation coefficient between categorical, ordinal and interval variables with pearson characteristics,” 2018. [Online]. Available: https://arxiv.org/abs/1811.11440

[27] R. Kohavi and D. Sommerfield, “Feature subset selection using the wrapper method: Overfitting and dynamic search space topology,” *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, 03 2001.

[28] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *CoRR*, vol. abs/1502.03167, 2015. [Online]. Available: http://arxiv.org/abs/1502.03167

[29] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT Press, 2016, http://www.deeplearningbook.org.

[30] K. P. Murphy, *Machine learning: a probabilistic perspective.* MIT press, 2012.

[31] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics).* Berlin, Heidelberg: Springer-Verlag, 2006.

[32] P. J. Rousseeuw and A. M. Leroy, *Robust regression and outlier detection.* John wiley & sons, 1987.

[33] K. Gokcesu and H. Gokcesu, “Generalized huber loss for robust learning and its efficient minimization for a robust statistics,” 2021. [Online]. Available: https://arxiv.org/abs/2108.12627

[34] D. Barrow, N. Kourentzes, R. Sandberg, and J. Niklewski, “Automatic robust estimation for exponential smoothing: Perspectives from statistics and machine learning,” *Expert Systems with Applications*, vol. 160, p. 113637, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417420304619

[35] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[36] N. S. Keskar and R. Socher, “Improving Generalization Performance by Switching from Adam to SGD,” 2017. [Online]. Available: https://arxiv.org/abs/1712.07628

[37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *Journal*

of Machine Learning Research, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[38] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," 2015. [Online]. Available: https://arxiv.org/abs/1506.02142

[39] J. H. Friedman, "Greedy function approximation: A gradient boosting machine." *The Annals of Statistics*, vol. 29, no. 5, pp. 1189 – 1232, 2001. [Online]. Available: https://doi.org/10.1214/aos/1013203451

[40] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 785–794. [Online]. Available: https://doi.org/10.1145/2939672.2939785

[41] T. S. Madhulatha, "An overview on clustering methods," 2012. [Online]. Available: https://arxiv.org/abs/1205.1117

[42] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN," *ACM Trans. Database Syst.*, vol. 42, no. 3, jul 2017. [Online]. Available: https://doi.org/10.1145/3068335

[43] J. Shlens, "A tutorial on principal component analysis," 2014. [Online]. Available: https://arxiv.org/abs/1404.1100

[44] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.