MA 420

E.ON Energy Research Center | RWTH AACHEN UNIVERSITY

Master's Thesis

Time Series Classification of Building Automation System Data Points Using Supervised Learning

Zeitreihenklassifikation von Datenpunkten in Gebäudeautomationssystemen mit Supervised Learning

Aachen, December 2021

Balázs Tóth
Matriculation number: 342515

Supervisors:
Florian Stinner, M.Sc.
Univ.-Prof. Dr.-Ing. Dirk Müller

The present work was submitted to the:
E.ON Energy Research Center | ERC
Institute for Energy Efficient Buildings and Indoor Climate | EBC
Mathieustrasse 10, 52074 Aachen, Germany

Abstract

Germany was one of the first countries, following the ratification of the Paris Agreement, to create its climate action plan which details the goals and targets to reduce greenhouse gas emissions across all sectors. This includes the building sector, where target is a 66% reduction in GHG emissions compared to 1990 levels until 2030. Upgrading the energy efficiency of existing buildings plays a huge role in reaching that target. As a result of the ongoing energy transition, energy systems are becoming increasingly decentralized and complex. Building automation and control systems of non-residential buildings collect enormous amounts of data, that offers opportunity for optimizing these systems. The availability of identifiable, structured data is integral in developing a control strategy for efficient energy use. Analysis of such data on a large scale requires manpower and time, that limit the realization of potential energy savings. Artificial intelligence could offer a solution to reduce the human involvement required.

Data points of building automation systems are generally labelled using non-standardized conventions. An algorithm based on natural language processing called AIKIDO has previously been developed to successfully automate the translating process from non-standardized names to a machine-readable form based on the BUDO schema. This translation method, however, does not offer solutions for recognition and translating completely arbitrary named labels, recognition of originally falsely labelled data points.

In this thesis, a data-based approach for classifying building automation system data points is employed, using state-of-the-art supervised time series classification algorithms such as the ROCKET and the MiniRocket, data points of the E.ON ERC building and classes derived from the label categories of the BUDO schema. The algorithms are tested on multiple different data sets including up to 10 classes and 2010 individual data points. The results are presented using metrics, such as F1 score, precision, accuracy, recall, the Matthews correlation coefficient and the confusion matrix. Multiple attempts for improving the input data, such as data filling, alleviating over-representation and increasing sampling rate are discussed in detail and tested. Finally, a comparison with the language-based approach is presented.

The results have shown, that supervised time series classification algorithms used with the time series data of building automation system data points and BUDO signal type category tags as classes were successful on the tested data sets. The MiniRocket RFC classifier reached the overall best performance in each trial and dominated all other tested classifiers in almost every single metric each time. The overall best performance was achieved in trial 5b with the average metrics of 83.58%. Data quality issues and the relatively low number of data points

belonging to certain classes had a negative effect on the overall performance. Tested methods for improving the input data, such as backward filling, alleviating over-representation, and increasing sampling rate were shown to positively effect the overall result.

Zusammenfassung

Deutschland hat als eines der ersten Länder nach der Ratifizierung des Pariser Abkommens seinen Klimaschutzplan erstellt, der die Ziele und Vorgaben zur Reduzierung der Treibhausgasemissionen in allen Sektoren konkretisiert. Dazu gehört auch der Gebäudesektor, dessen Ziel eine Reduzierung der Treibhausgasemissionen um 66% gegenüber dem Niveau von 1990 bis 2030 ist. Die Verbesserung der Energieeffizienz bestehender Gebäude spielt eine große Rolle, um dieses Ziel zu erreichen. Durch die fortschreitende Energiewende werden Energiesysteme immer dezentraler und komplexer. Gebäudeautomations- und Steuerungssysteme von Nichtwohngebäuden sammeln enorme Datenmengen, die Möglichkeiten zur Optimierung dieser Systeme bieten. Die Verfügbarkeit identifizierbarer, strukturierter Daten ist ein wesentlicher Bestandteil bei der Entwicklung einer Kontrollstrategie für eine effiziente Energienutzung. Die Analyse solcher Daten in großem Maßstab erfordert Personal und Zeit, die die Realisierung potenzieller Energieeinsparungen einschränken. Künstliche Intelligenz könnte eine Lösung bieten, um die erforderliche menschliche Beteiligung zu reduzieren.

Datenpunkte von Gebäudeautomationssystemen werden in der Regel nach nicht standardisierten Konventionen gekennzeichnet. Ein auf natürlicher Sprachverarbeitung basierender Algorithmus namens AIKIDO wurde zuvor entwickelt, um den Übersetzungsprozess von nicht standardisierten Namen in eine maschinenlesbare Form basierend auf dem BUDO-Schema erfolgreich zu automatisieren. Dieses Übersetzungsverfahren bietet jedoch keine Lösungen zur Erkennung und Übersetzung von völlig willkürlich benannten Labels, oder zur Erkennung von ursprünglich falsch gekennzeichneten Datenpunkten.

In dieser Arbeit wird ein datenbasierter Ansatz zur Klassifizierung von Datenpunkten von Gebäudeautomationssystemen verwendet, mit modernsten überwachten Zeitreihen- Klassifizierungsalgorithmen wie ROCKET und MiniRocket, Datenpunkte des E.ON ERC-Gebäudes und Klassen abgeleitet von den Label-Kategorien des BUDO-Schemas. Die Algorithmen werden an mehreren verschiedenen Datensätzen getestet, darunter bis zu 10 Klassen und 2010 einzelne Datenpunkte. Die Ergebnisse werden anhand von Metriken wie F1-Score, Präzision, Genauigkeit, Recall, dem Matthews-Korrelationskoeffizienten und der Konfusionsmatrix präsentiert. Mehrere Versuche zur Verbesserung der Eingabedaten, wie Datenfüllung, Minderung von Überrepräsentation und Erhöhung der Abtastrate, werden ausführlich diskutiert und getestet. Abschließend wird ein Vergleich mit dem sprachbasierten Ansatz vorgestellt.

Die Ergebnisse haben gezeigt, dass überwachte Zeitreihen- Klassifizierungsalgorithmen, die mit den Zeitreihendaten von Gebäudeautomationsdatenpunkten und BUDO Signaltyp Kategorie Tags als Klassen verwendet wurden, auf den getesteten Datensätzen erfolgreich waren.

Der MiniRocket RFC-Klassifikator erreichte in jedem Versuch die beste Gesamtleistung und dominierte jedes Mal alle anderen getesteten Klassifikatoren in fast jeder einzelnen Metrik. Die insgesamt beste Leistung wurde in Versuch 5b mit den durchschnittlichen Metriken von 83,58% erzielt. Probleme mit der Datenqualität und die relativ geringe Anzahl von Datenpunkten, die zu bestimmten Klassen gehören, wirkten sich negativ auf die Gesamtleistung aus. Getestete Methoden zur Verbesserung der Eingabedaten, wie Rückwärtsfüllung, Minderung von Überrepräsentationen und Erhöhung der Abtastrate, haben sich positiv auf das Gesamtergebnis ausgewirkt.

Table of Contents

Nomenclature

Abbreviations

Symbol	Meaning
AI	Artificial Intelligence
AIKIDO	Artificial Intelligence based Key Interface for Data point metadata nOrmalisation
BACS	Building Automation and Control System
BAS	Building Automation System
BMS	Building Management System
BUDO	Buildings Unified Data point naming schema for Operation management
CNN	Convolutional Neural Network
DL	Deep Learning
DTW	Dynamic Time Warping
EBC	E.ON Institute for Energy Efficient Buildings and Indoor Climate
ERC	E.ON Energy Research Center
FN	False Negative
FNN	Feedforward Neural Network
FP	False Positive
GHG	GreenHouse Gas
ISE	Institute for Solar Energy Systems
LT-LEDS	Long-Term Low GHG Emission Development Strategy
MCC	Matthews Correlation Coefficient
ML	Machine Learning
MLP	MultiLayer Perceptron
NaN	Not a Number
NLP	Natural Language Processing
NN	Neural Network
RFC	Random Forest Classifier
RNN	Recurrent Neural Network

Continued on the next page

Abbreviations

Symbol	Meaning
ROCKET	Random Convolutional Kernel Transform
SGD	Stochastic Gradient Descent
TN	True Negative
TP	True Positive
TSC	Time Series Classification

List of Figures

List of Tables

1 Introduction

1.1 Environmental aspects - motivation

Following the ratification of the Paris Agreement [UN, 2015], a legally binding international treaty on climate change, all participating countries were required to define a long-term low greenhouse gas emission development strategy (LT-LEDS). Germany was one of the first countries that submitted their plan to the United Nations, called the Climate Action Plan 2050 [BMU, 2016], adopted in 2016, which specifies the long-term goals and targets on reducing greenhouse gas (GHG) emissions per sector.

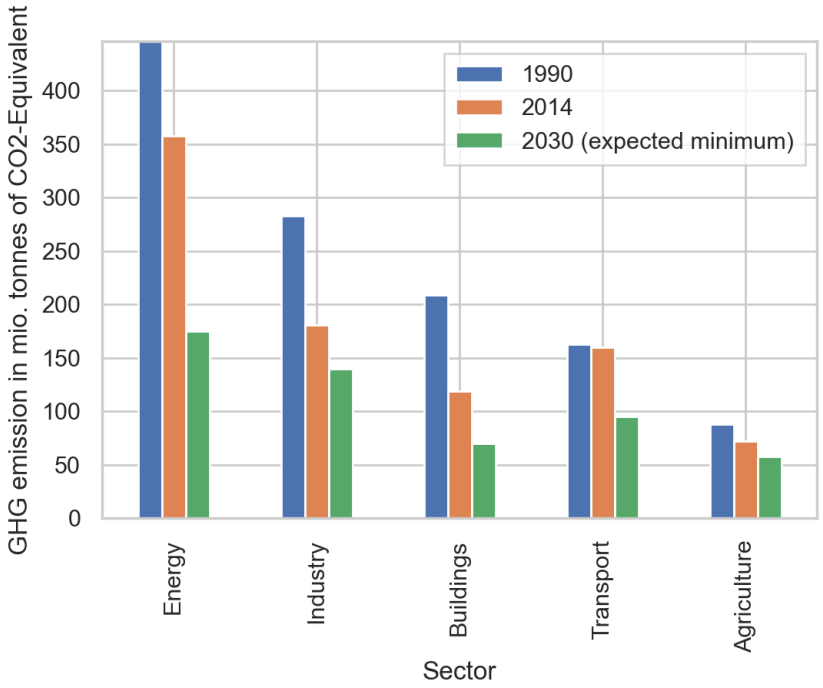

Figure 1.1: Sectoral targets in the Climate Action Plan 2050 [BMU, 2016]

The road-map for the buildings sector, the sector with the 3rd largest impact on GHG emissions in the Climate Action Plan, has been drawn up to achieve a basically climate-neutral building stock by 2050. Target for the building sector is to reduce GHG Emissions by about 66% compared to 1990 levels until 2030.

To achieve these goals a three-prong approach was set out:

- Introducing progressively developing zero-energy building standards for new buildings
- Gradually phasing out fossil-fuel heating systems with help of incentives where necessary
- **Long-term strategies for upgrading the energy efficiency of existing buildings**

1.2 Technical aspects - motivation

As a result of the ongoing energy transition throughout the whole world from conventional to renewable sources, energy supply systems are becoming increasingly decentralised and complex. Sensors in building automation and control systems (BACS) of non-residential buildings collect enormous amounts of data that could be used to optimize these systems. The availability of structured data of these sensors is integral in developing a control strategy for efficient energy use [Bode et al., 2019]. Analysis of such data on a large scale requires manpower and time, two scarcely available resources, that limit the realization of potential energy savings. Once sensory data is collected in a structured and standardized manner, deployment of artificial intelligence algorithms would be possible to optimize control strategy.

In order to successfully apply artificial intelligence in BACS, the collected data has to be distinctly identifiable. Generally the data points of sensors in BACS are identified by labels (data point names), that are, in most cases, given manually, and decided individually often using non-standardized meta data schemes, conventions of the manufacturer or the engineer tasked with installing them. A standardized naming convention for the system components and data points would provide basis for a reliable working algorithm for developing a control strategy. [Stinner et al., 2018]

To solve this problem standardized methods for naming data points have been developed. One of these methods is the BUDO schema, developed by a joint team from E.ON ERC and Frauenhofer ISE. [Stinner et al., 2018], [Stinner, 2020]. Using this schema the original labels can be translated into a machine-readable standard labels.

An algorithm based on natural language processing called AIKIDO has previously been developed to successfully automate the translating process and significantly reduce the manual effort while maintaining an accuracy between 0.82 - 0.97. [Krieger, 2021], [Stinner et al., 2019]

This language-based method, however, solely considers the original labeling of the data point, when determining the standardized translation. It does not offer solutions for:

- Recognizing and translating completely arbitrarily named labels.
- Finding out any further information about a data point not indicated by its label that might be necessary for the development of a control strategy.

- Recognition of originally falsely labelled data points.

A data-based approach, while having its own limitations, could further improve the results of the language-based method, as well as point out inaccuracies, that the language-based approach inherently cannot.

1.3 Forewords and structure

This thesis aims to investigate the feasibility of supervised time series classification (TSC) of data sets using data of the sensors and control units from the E.ON ERC building as data points and BUDO schema category tags as pre-determined classes. Data sets will include 10 different classes that are selected to represent a wide variety of possible signal behavior. All programming work was done in Python using the EBC Supervised Learning package, which allows for testing multiple classifier algorithms. Recently, new state-of-the-art classifiers were added to the supervised learning package, these will be further discussed and excessively tested.

Throughout the thesis the following subjects will be investigated:

- Theoretical backgrounds, related work and metrics for evaluation
- Data set and class selection
- Conventional and state-of-the-art classifiers
- Influencing factors, such as:
    - Data quality
    - Pre-processing
    - Time span
    - Sampling rate
    - Merging of data points
- Comparison with the language-based approach

Results will be discussed with help of tools and metrics commonly used in supervised classification, such as the confusion matrix, F1-score and Matthews correlation.

At the end of each chapter, where applicable, a short summary will conclude the findings and reasoning for the decision making going forward.

To ensure replicability, a short „user manual" is provided together with all python files, data sets and results in the digital documentation.

2 Theoretical background

2.1 Time series

Time series are series of data points most commonly taken at successive equally spaced intervals in time. They are a sequence of discrete-time data, which means that the data point values remain unchanged within each time period. In this thesis, individual time series data is stored in tables where one column contains the beginning of a each time interval in a sequence the data was taken, the other column starts with the name of the data point and contains the data point values corresponding to each specified time period. Time series are used in virtually every domain of applied science and engineering where temporal measurements are taken.

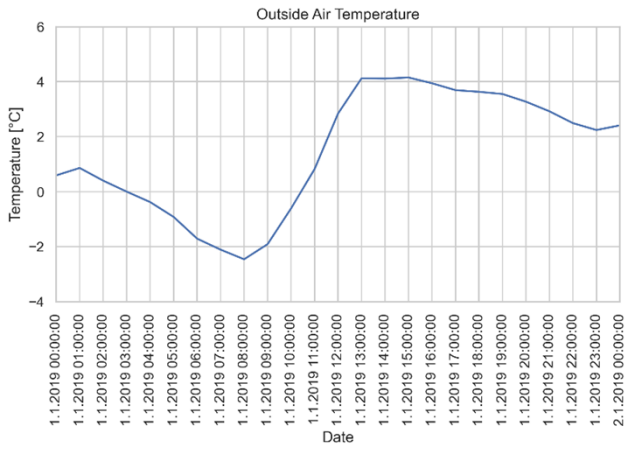

Figure 2.1: Example time series of outside air temperature (non-discrete representation)

Time series make up the basis for the data sets used in the investigations throughout this thesis.

2.2 Building automation and control systems

Building automation and control systems (BAS or BACS), also generally referred to as building management systems (BMS) are computer-based control systems installed in buildings. They are a collection computers, controllers, sensors and actuators (or other output devices), that are electrically connected together to monitor and control the electrical and mechanical

equipment of a building, such as lighting, power systems, air-handling, closed-circuit television, motion detectors and the fire- and security system. Depending on the automation level, BACS can encompass all significant sources of energy consumption in a building.

The architecture of BASs can conceptually be divided into four layers. The input/output layer, the field controller layer, the supervisory layer and the server/application layer. [CA, 2021]

The **input/output layer** consist of sensors and actuators or other output devices. Sensors are measuring devices that convert a physical reality into a signal. Actuators modulate valves and dampers. The sensors and actuators are attached to input and output ports on hardware modules.

The **field controller layer** is responsible for orchestrating the interaction between the input and output devices. The controllers are embedded with control logic, that control the actuators according to the feedback from the inputs or by commands from the system.

The components of the previous two layers work together in a process called the control loop: The input signal from a sensor is received the controller, the controller processes the input according to its embedded logic and controls the response to the output device. Setpoints are used by the controller programming to compare the inputs with a desired value and generate an output.

The **supervisory layer** is where the data from the field controllers is consolidated by the supervisory devices. These manage communication between multiple field controllers and the server/application layer. Some supervisory devices can also serve as user interfaces.

The **server/application layer** consolidates the data received from multiple supervisory devices and delivers it to the end user through a user interface. Servers typically store all relevant data, that is required for running a BAS (settings, setpoints, schedules and alarms) as well as time series data delivered by the sensors and output devices.

2.3 Machine learning

Machine learning (ML) is a branch of the computer science artificial intelligence (AI) that aims to imitate the way humans learn with the help of data, algorithms and mathematical models to gradually improve accuracy.

In today's world data is generated and stored on massive scale. So-called big data applications are used in various fields ranging from building automation systems, flight control systems, geomorphic mapping of planetary surfaces [Zhang, 2010] to the new industry of self driving vehicles [Foote, 2019]. Recent developments in the fields of image analysis and speech

recognition have generated interest from data scientists studying different domains. [Deng and Li, 2013]

Arthur Samuel, an employee of IBM, came up with the with the expression „machine learning" in 1952. In the 1950s, he developed an algorithm for playing checkers. Due to lack of available computational memory at the time, he came up with a scoring function, which, depending on the position of the pieces, attempted to measure the chances of each side winning, and used an algorithm, called the "minimax algorithm", to decide its next optimal move. He also designed multiple mechanisms to allow his algorithm to learn and improve over time. [Foote, 2019]

Keith D. Foote [Knight, 2017] describes machine learning as follows:

"Machine learning, at its most basic, is the practice of using algorithms to parse data, learn from it, and then make a determination or prediction about something in the world. So rather than hand-coding software routines with a specific set of instructions to accomplish a particular task, the machine is 'trained' using large amounts of data and algorithms that give it the ability to learn how to perform the task."

Since the invention of the minimax algorithm machine learning has been used for solving various tasks. However, its simplistic model, only using one statistical or mathematical operation, does not allow for solving more complex tasks. To deal with more complex tasks, deep machine learning or deep learning (DL) was developed. As opposed to ML models, deep learning models use multiple operation layers. [Hrabia, 2020]

2.3.1 Neural networks

Neural networks (NN) are a subset of machine learning models, that are used in deep learning algorithms. They are designed to mimic the way the human brain works. A neural network model is comprised of node layers, containing an input and an output layer, and one or more hidden layers in between. Each artificial neuron (or node) connects to another and has a weight and a threshold value. A node is activated, if its output is above the specified threshold value, otherwise no data will be passed along to the next layer. Neural networks require training data to improve their accuracy and learn over time. [IBM, 2020]

There are different types of neural network architectures, each using different principles in determining their own internal rules, such as how the nodes of each layer are connected, how many hidden layers they use, what activation functions, thresholds and eventual feedback loops they employ.[Culurciello, 2017] Some of the more common ones are the feedforward neural network (FNN), the multilayer perceptron (MLP), the convolutional neural network (CNN) and the recurrent neural network (RNN) [GL, 2020].

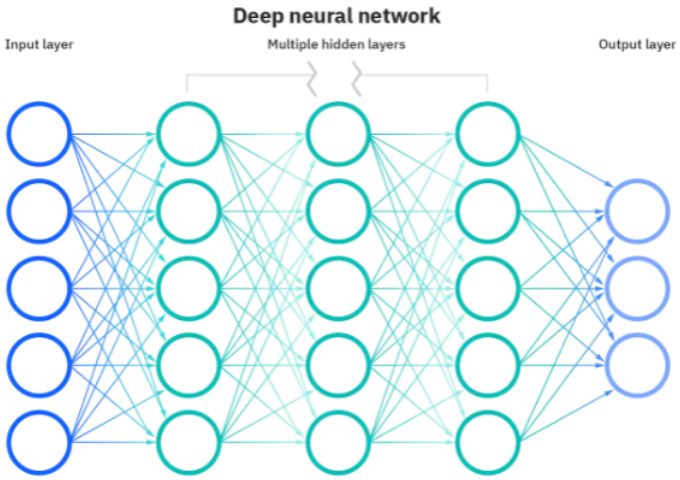

Figure 2.2: Deep neural network [IBM, 2020]

Algorithms based on the CNN architecture have been shown to be the fastest when used for time series classification tasks while maintaining state-of-the-art accuracy. [Dempster et al., 2020],[Dempster et al., 2021] These will be discussed in further detail in Chapter 3.

2.3.2 Types of machine learning

Depending on the task, the data available and the desired outcome, multiple methods can be employed to train a machine learning algorithm. Three of the commonly used methods are the following: supervised learning, unsupervised learning and reinforcement learning.

Supervised learning is a conceptually straight forward way of training a Machine Learning algorithm. The data used for training is labeled or classified, meaning that for each input the algorithm is presented with a previously defined output. Human labor is usually required to define the labels. Using enough training data, the algorithm can establish a cause-and-effect relation between the parameters given. Thus, when presented with previously unknown input data, it can predict the most likely output. This makes it well-suited for tackling classification tasks.

Unsupervised learning method allows the algorithm to identify patterns within the data set and classify, label or group data points without any external guidance required, as it uses data that was previously not labeled. It can be deployed on large amounts of data without much human labor and can tackle tasks with large complexity.

Reinforcement learning is a way of training machine learning models to make a sequence of decisions. In each sequence the previous decision is either rewarded or penalized. The goal is to find the sequence of decisions with the maximum reward. This method is probably the

closest to the original intent of Arthur Samuel's minimax algorithm and it is used in areas such as robotics, trading, finance and gaming among many others. [Brownlee, 2015]

2.3.3 Classification and regression

The general task machine learning algorithms are employed for is to use available data to predict an outcome.

If the prediction is a certain numeric measurement, it is called a **regression problem**. An example for this task type could be when the desired outcome is to predict the movement of the stock market.

If the prediction is represented as classes, it is called a **classification problem**. An example for this task type could be an image recognition problem, where the machine learning algorithm is used to make predictions differentiating between two objects. This thesis focuses on classification problems exclusively.

2.3.4 Supervised time series classification

Supervised time series classification is a machine learning approach that uses the supervised learning method to solve a classification problem using time series as input data and predefined classes as labels. Once a model is trained, it should be able to predict (or assign) a class to each previously unknown time series it is presented with. [Amidon, 2020]

As the main subject of this thesis, the practical application and evaluation of this approach will be excessively discussed throughout all following chapters.

2.4 Metrics for evaluation

Metrics, or performance indicators, are useful when comparing and evaluating multiple machine learning algorithms. They give an indication on how well a model is performing tackling a certain task and can provide the user with clues about opportunities for improvement. An overview of the most commonly used metrics for multi-class classification is summarized in the study by Margherita Grandini et. al [Grandini et al., 2020]. Advantages and disadvantages of certain performance indicators over others are detailed in the papers of Davide Chicco et.al. [Chicco and Jurman, 2020] and Boaz Shumeli [Shmueli, 2019].

This chapter will shortly introduce the metrics and performance indicators used in this thesis.

		PREDICTED classification				
	Classes	a	b	c	d	Total
ACTUAL classification	a	6	0	1	2	9
	b	3	9	1	1	14
	c	1	0	10	2	13
	d	1	2	1	12	16
	Total	11	11	13	17	**52**

Figure 2.3: Example of confusion matrix [Grandini et al., 2020]

2.4.1 The confusion matrix

The confusion matrix is a cross table that allows for comparison between the actual classification and the predicted classification provided as the result of a machine learning model. The classes are listed in both the rows and columns in the same order. The numbers represent the number of elements of a class that were predicted. The number correctly classified elements are on the main diagonal of the matrix from top left to bottom right. The elements of the last column (Total) show the sum of all elements actually belonging to each class. The elements of the last row (Total) show the sum of all elements that are predicted to be of the corresponding class. An example confusion matrix is shown in figure 2.3.

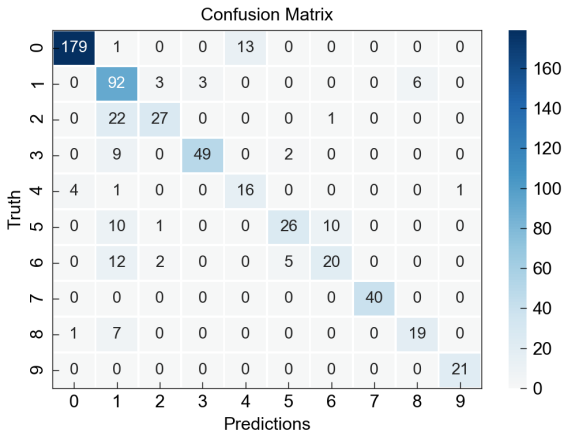

Figure 2.4: Example of regular confusion matrix

Heat-map depictions are often used to simplify the evaluation of a confusion matrix, especially

when dealing with large number of elements. In this thesis, two types of depictions will be used to evaluate the findings:

The **regular confusion matrix** shows the exact number of elements of each class. It makes use of the heat-map, but it is biased towards over-represented classes, almost hiding certain correlations between classes with lower number of elements.

In the **normalized confusion matrix**, the elements of each row are normalized using the sum of all true elements of the given class, making it easier to observe the dynamic between the classes.

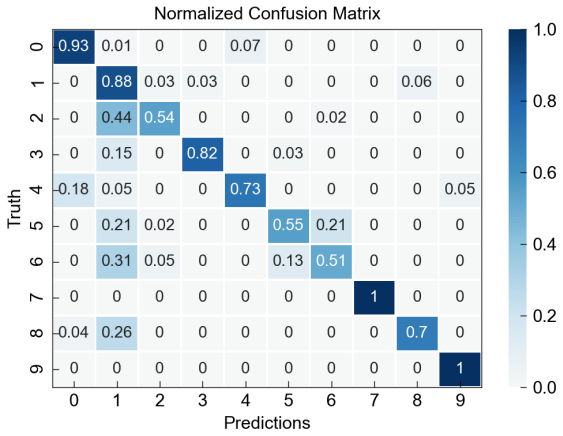

Figure 2.5: Example of normalized confusion matrix

2.4.2 Precision and recall

The two metrics precision and recall will serve as building blocks for calculating the accuracy and the F1-score. These are explained using a basic two-class confusion matrix, where one class is referred to as positive (1), the other as negative (0) class.

		PREDICTED		
	Classes	Positive (1)	Negative (0)	Total
ACTUAL	Positive (1)	TP = 20	FN = 5	25
	Negative (0)	FP = 10	TN = 15	25
	Total	30	20	**50**

Figure 2.6: Example of two-class confusion matrix [Grandini et al., 2020]

True positive (TP) elements are the elements that were predicted positive by the model and are also actually positive, whereas False Positive (FP) elements are the ones predicted positive by the model but are actually negative. Similarly, the false negative (FN) elements are the elements that were predicted by the model as negative, but are actually positive, whereas the true negative (TN) elements are the ones predicted negative by the model and also actually negative.

Precision

Precision is calculated by dividing the number of true positive elements by the sum of all positively predicted elements. As such, precision gives an indication of how likely a model is at correctly predicting an individual as positive.

$$Precision = \frac{TP}{TP + FP} \tag{2.1}$$

Recall

Recall is calculated by dividing the number of True Positive elements by the sum of True Positive and False Negative elements. As such, it gives an indication of a model's ability to predict all positive elements.

$$Recall = \frac{TP}{TP + FN} \tag{2.2}$$

2.4.3 F1-score

The F1-Score uses the harmonic mean of the Precision and Recall performance indicators, thereby helping to find the best trade-off between the two metrics. It can assume values in the range of 0 to 1 and can be applied in binary classification cases. However, just like precision and recall, it does not take into consideration the number of true negative (TN) elements.

$$F1 - Score = 2 \cdot \left(\frac{precision \cdot recall}{precision + recall} \right) \tag{2.3}$$

2.4.4 Accuracy

Accuracy is calculated by dividing the sum off all true positive and true negative elements by the sum of all elements of the confusion matrix. As such, accuracy gives an indication of how much a model is correctly predicting on the entire data set.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.4}$$

2.4.5 Matthews correlation coefficient

Unlike the F1-score, the Matthews correlation coefficient (MCC) takes into account all the cells in the confusion matrix, resulting in a more balanced performance indicator in binary classifications. The MCC can take up values in the range of [-1; 1]. An MCC value of -1 indicates, that none of the elements were correctly predicted, while an MCC value of 1 indicates that all elements were correctly predicted. An MCC value of 0 means that the predictions made by a classifier were no more accurate, than a randomized classification would be.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FN) \cdot (TP + FP) \cdot (TN + FN) \cdot (TN + FP)}} \tag{2.5}$$

All performance indicators discussed in Chapter 2.4) can be directly computed from the elements of the confusion matrix and can be extended from binary to multi-class classification cases. In practice, python libraries used for supervised learning algorithms already include built-in evaluation tools that provide the user with the desired metrics.

2.5 The BUDO schema

The buildings unified data point naming schema for operation management (BUDO schema) is a machine-readable, uniform and expandable meta data schema developed by Stinner et al., that can combine all necessary information about a data point into one hierarchically structured label. [Stinner, 2020]

The standardized part of the label (starting with the § character) is divided into multiple categories. These categories are: system, specification of the system, optional designation,

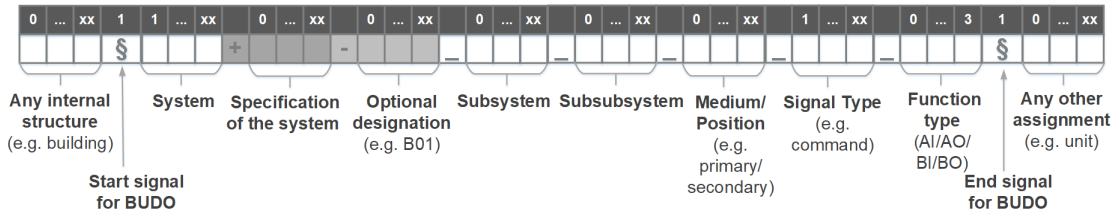

Figure 2.7: Budo schema structure [Stinner, 2020]

subsystem, subsubsystem, medium/position, signal type and function type. The categories can then be further specified with category tags (or attribute types).

Using this structure, non-standardized data point labels can be translated into machine-readable labels either manually, or automatically using the AIKIDO tool, which was shown to have achieved an accuracy between 0.82 – 0.97 when tested on 5 different buildings [Stinner et al., 2019]. This solution, though not perfect, can drastically reduce the human labor required for translating data point labels of existing building automation and control system data points.

In this thesis, various category tags from the BUDO schema are selected to be pre-determined classes in the input data for supervised time series classification algorithms. (Further discussed in Chapter 4.)

3 Related Work

Numerous studies have been conducted on the subject of automated data point mapping, including time series classification of BACS data points using both supervised and unsupervised methods. The closest ones related to the subject of this thesis are discussed below.

Studies have argued the need for automated point mapping of building automation systems data points, as well as the benefits of reducing the human involvement in the process. [Wang et al., 2018] points to the fact, that in the future, due to cost-effective solutions, even smaller commercial buildings are likely to install BACS for improving energy efficiency. On the modernized electric grid, commercial buildings will be able take part in load control optimization for potential economic benefit. [Callaway, 2011]

Various approaches exist for selecting meta data sources for automated data point mapping. [Stinner et al., 2019] uses the AIKIDO tool and the existing non-standardized data point labels to translate the labels into machine-readable form.[Balaji et al., 2015] and [Park, 2012] extended the metadata sources to include units, data types and text descriptions.

Studies by [Hong et al., 2015] and [Gao et al., 2015] use time series data for BAS data point classification by extracting statistical features (such as variance, median, maximum, minimum, etc.) and using these features as classifier inputs. Results of these two studies range from 50% - 100% accuracy depending on the data point type. [Gao et al., 2015] was able to reach a mean F1-score between 0.6 and 0.75 on two different buildings, however, using the classifier trained on one building to predict on the data points of the other resulted in an F1-score close to zero for many data point types.

Bode, Schreiber et al. using data sets from the E.ON ERC building found, that multivariate unsupervised algorithms are able to find clusters of similar data points in general, using statistical features of the time series such as mean value, variance, skewness, etc . They showed, however, that unsupervised learning is mostly inadequate at finding clusters of data points that represent the pre-defined class labels and that unsupervised learning cannot be used to avoid the need for training data [Bode et al., 2019]. In this study a short term (one day with a sampling rate of 60s) and a long term (20 days with an sampling rate of 15 min) data set was used. In the conclusions and future work section the authors specify the importance of using larger time frames to be able to identify time series behavior related to influencing factors such as seasonality.

Fütterer, Kochanski et al. [Fütterer et al., 2017] investigated multivariate supervised learning

methods with similar data set and time series features extracted as Bode, Schreiber et al. using 22 pre-defined classes. 13 different classifiers were tested. It was confirmed that the bagged tree and the random forest classifiers are best suited for handling such time series data. In this study only one data set was used (one day with a sampling time of 60s). As next steps, the authors specify the need for further improvement in classification accuracy and minimizing computational time.

Most time series classification methods that are able to achieve state-of-the-art accuracy require high computational complexity and considerable training times even for small data sets. [Dempster et al., 2020]

Since conducting these studies, several advances have been made regarding computational time. As recently as 2020 December, Alejandro Pasos Rulz et al. [Bagnall et al., 2017] released a study called "The great multivariate time series classification bake off", where multivariate time series classification algorithms are compared.

Among all algorithms tested, compared to dynamic time warping (DTW, the most used benchmark for time series classification), the ROCKET (random convolutional kernel transform) algorithm was shown to be by far the fastest of all currently used algorithms, while only marginally falling behind in classification accuracy.

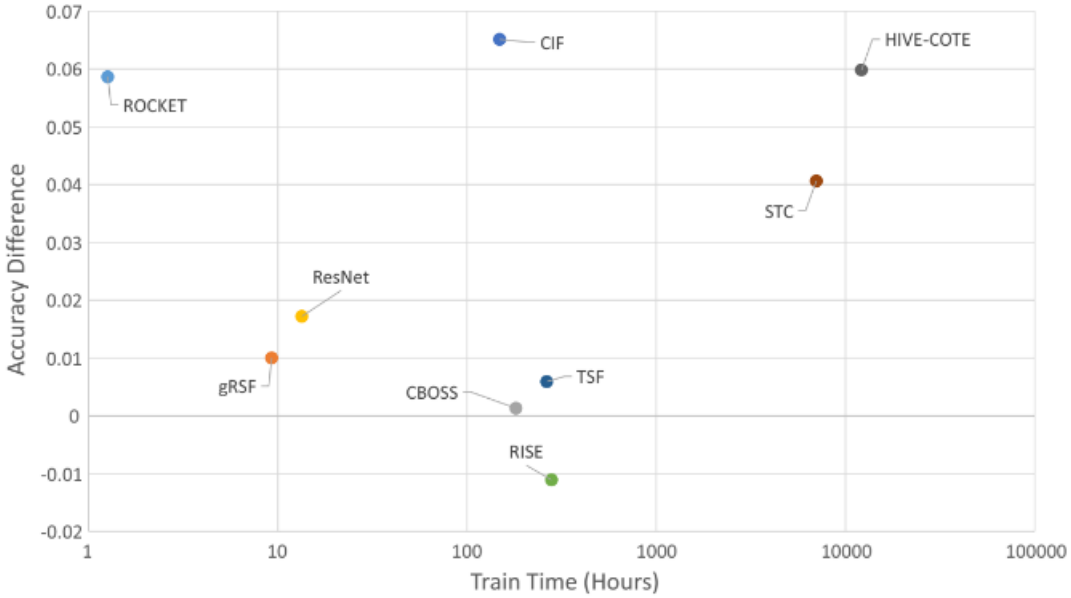

Figure 3.1: Average difference in accuracy to DTW vs. train time for 9 multivariate time series classifier algorithms [Bagnall et al., 2017]

Alexandra Amidon [Amidon, 2020], in a Brief survey of time series classification algorithms also discusses the advantages using ROCKET over other classifiers. Angus Dempster et

al. [Dempster et al., 2020] showed that on the tested 85 different data sets, the ROCKET classifier was able to finish all data sets within 1 hr 50 minutes compared to 6 days for the second fastest classifier (inceptiontime) with similar accuracy. In a further study, Angus Dempster et al. [Dempster et al., 2021] reformulated the ROCKET classifier into an even faster algorithm called MiniRocket, that is up to 75 times faster than the ROCKET, while retaining comparable accuracy and is significantly more accurate than any other method requiring similar computational power.

3.1 Conclusions - related work

- Studies on time series classification have confirmed, that supervised learning is an appropriate method to use when labeling BACS data points, as clusters of data points identified by unsupervised methods did not correspond to labels used in BACS.
- Due to computational restraints, most previous studies were conducted using short time frames which does not allow for capturing a completely representative data point behavior.
- Previous computational power and time restraints can be alleviated using new state-of-the-art algorithms, such as ROCKET and MiniRocket.

4 Data

As also discussed in chapter 2, supervised time series classification algorithms require data sets (training data) to be trained. Data sets in this study consist of time series data from data points, as well as a class manually assigned to each data point. Using enough training data, a supervised learning algorithm can establish a cause-and-effect relation between the time series data and the classes and provide predictions on previously unknown data points as to which class they most likely belong.

In this chapter, the methods used for data collection for the data sets, selection of training data, the definition of classes and the evaluation of data quality will be detailed.

4.1 Database and data collection

Time series data used in this study was collected from the building automation and control system of the E.ON ERC building (Mathieustraße 10, 52074 Aachen, Germany).

Data collected by the BACS is available for visualization as well as for downloading at the Aedifion online database (https://aedifion.io/). The database can be accessed through an online portal, the data can be downloaded directly from the website, or with an automated request using a simple python script, specifying:

- Name of the data point
- Desired time interval
- Desired sample rate
- Type of interpolation

In case of this study, an automated script was used to download the data and store it in excel format for each data point separately. When dealing with large time intervals and low sample rates it can be beneficial to use other formats, such as pandas data frame saved in python pickle format, due to the size limitations of excel.

To collect the data, the names of the data points were provided by the EBC institute. Multiple specifications regarding time interval and sample rate were used to create data sets for the trials in chapter 5, these will be discussed before each trial. To be able to asses data quality and test various pre-processing methods, interpolation from the database side was deactivated.

4.2 Data composition

At the time of conducting this study, 13.131 data point labels from the E.ON ERC building were translated manually in accordance with the BUDO schema. From these, time series data of 12.958 data points were available for download from the Aedifion database. These data points form the base data set.

In the BUDO structure, each data point label contains specifications for standard categories (discussed in chapter 2.5). To each category specification in a data point label, further category tags are assigned.

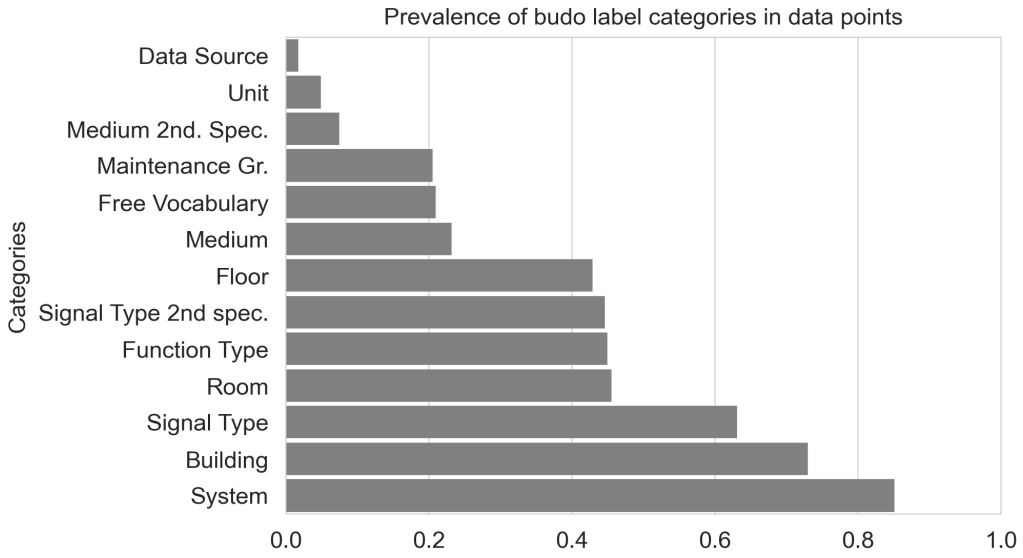

Figure 4.1: BUDO categories present in the base data set

The figure 4.1 demonstrates how prevalent the different categories are in case of the data point labels of the E.ON ERC Building in descending order.

The three most represented categories are system, building and signal type, all between 62-83%. All other categories are present in less than 50% of all data point labels.

As all BUDO categories are exclusively translated from the original labels of the data points, there is no indication on how successfully a machine learning algorithm could pick up on the patterns in the underlying time series data of any category.

This thesis, however, focuses only on data points containing labels from the signal type category and its category tags, as time series data is the direct representation of signal behavior.

4.2.1 Conclusions - data composition

- None of the BUDO category labels were represented in all data points in the base data set.
- Using the signal type category and its category tags appear to be most adequate when creating training data for time series classification.
- Further studies could reveal patterns in the underlying time series of data points when using other BUDO categories and category tags to create training data.

4.3 Selection of pre-defined classes

To specify the pre-defined classes for the time series classification training data, all possible tags assigned to the signal type category of the data points labels were examined. There were overall 325 different tags, many of which only with a few occurrences in the base data set.

Training classifier algorithms on a data set, where some individual classes are underrepresented, leaves the classifier algorithm little opportunity to train on that specific class [Brownlee, 2015]. Therefore, as a first step, all individual tags with less than 50 occurrences were excluded. This left 9.793 individual data points with 35 possible tags that could qualify as a pre-defined class, shown in figure 4.2.

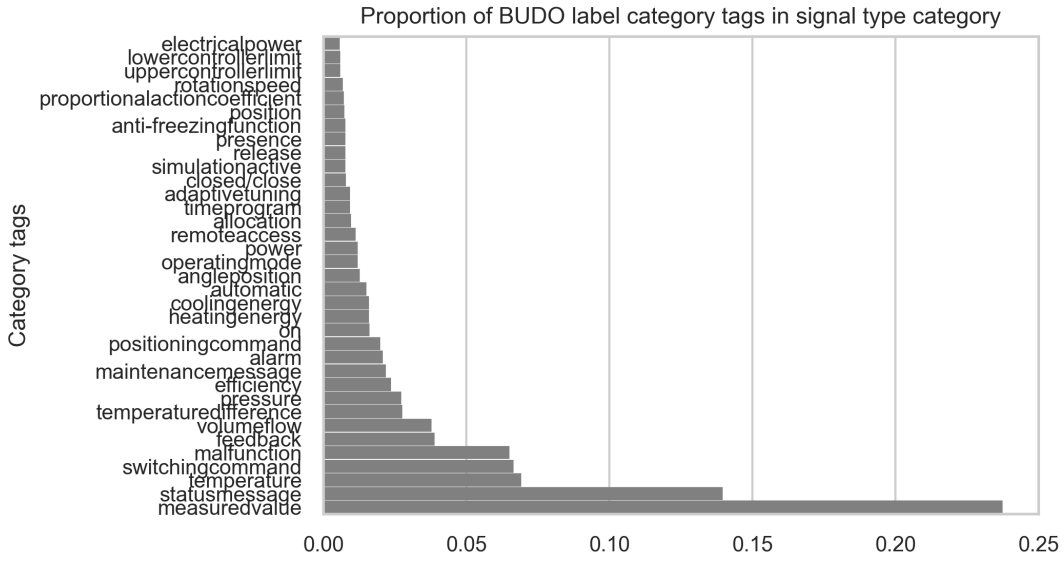

Figure 4.2: Possible pre-defined classes with over 50 unique occurrences

Just as under-representation, over-representation can negatively effect the results of supervised classification [Brownlee, 2015]. Figure 4.2 shows, that after eliminating all under-

represented label category tags, the two largest groups, "status message" and "measured value" are over-represented compared to all others.

Furthermore, there are several category tags that cover binary data point behavior such as: “on”, “presence”, “closed/close”, "status message" and “feedback”. It is assumed, that leaving too many such classes in the data set will likely dilute the results of classifier algorithms. Creating one class where all tags covering similar data point behavior are merged into one single class is an option that is discussed in chapter 5.5.

Finally, 10 category tags were selected, covering data points with a variety of behaviors but also leaving room for investigating possible influences of over-representation and merging. This resulted in the selection of 2010 individual data points, that create the basis for the main data set used throughout this study.

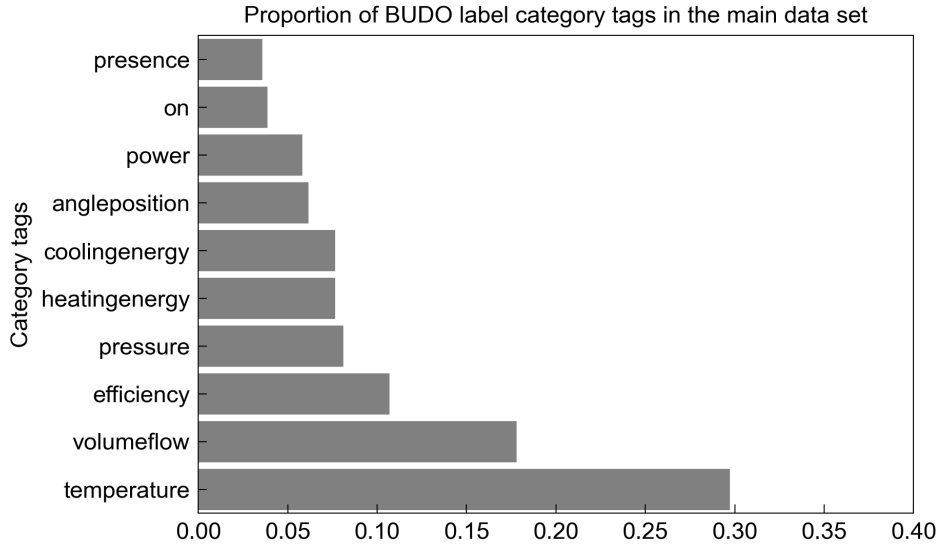

Figure 4.3: Final pre-defined classes for training data

Table 4.1 shows the selected classes and their numerical representation, which is used in the training data to refer to each class, as well as in the confusion matrices during the evaluation of results in chapter 5.

Table 4.1: Final selection of classes for supervised time series classification

Class	Numerical representation
temperature	0
volumeflow	1
pressure	2
efficiency	3
on	4
coolingenergy	5
heatingenergy	6
angleposition	7
power	8
presence	9

4.4 Data quality

When dealing with time series classification, the quality of input data is of outermost importance [Sessions and Valtorta, 2006]. Building automation and control systems can collect massive amounts of data delivered by sensors and control units in a building. This data is then transferred to a data collection system through communication protocols where it is stored for later use. Information must pass through multiple devices and layers of communication before it can be stored. This poses a risk to the quality of the collected data.

In a study, Ralf Gitzel [Gitzel, 2016] introduces 26 different reasons that can lead to data quality problems in time series data. Some of these are: event data loss, values out of range, value spikes, wrong timestamps, signal noise, signal alteration, units of measurements, data formats, timestamp formats, diverging sampling, data not updated, etc.

The combination of these possible issues in real life applications result in imperfect data quality, that can partially be mitigated during data pre-processing. In this study 2 different pre-processing methods were tested, these are discussed in chapter 4.5.2.

4.4.1 Missing data

As a simple indicator of data quality, time series data of the selected data points were examined to see how many empty time stamps they contain. Empty time stamps (or empty values) are represented in the data base as "NaN" values. Details of the data set and the settings used for acquiring them from the Aedifion data base are shown in table 4.2.

Table 4.2: Main data set used for quality report

Data set	Main data set
Number of classes	10
Number of data points	2010
Time span	01.01.2020 - 01.01.2021
Sample rate	1 h
Time intervals	8784

Figure 4.4 shows the proportion of "NaN", or empty values in the time series data of each data point in descending order, the results can be summarized as follows:

- About 58% of all data points have 90% or more time stamps empty in the requested period.
- Only 17% of all data points have less than 20% the time stamps empty.
- Overall, 72% of all time stamps of the data points in the requested period contained no recorded values.

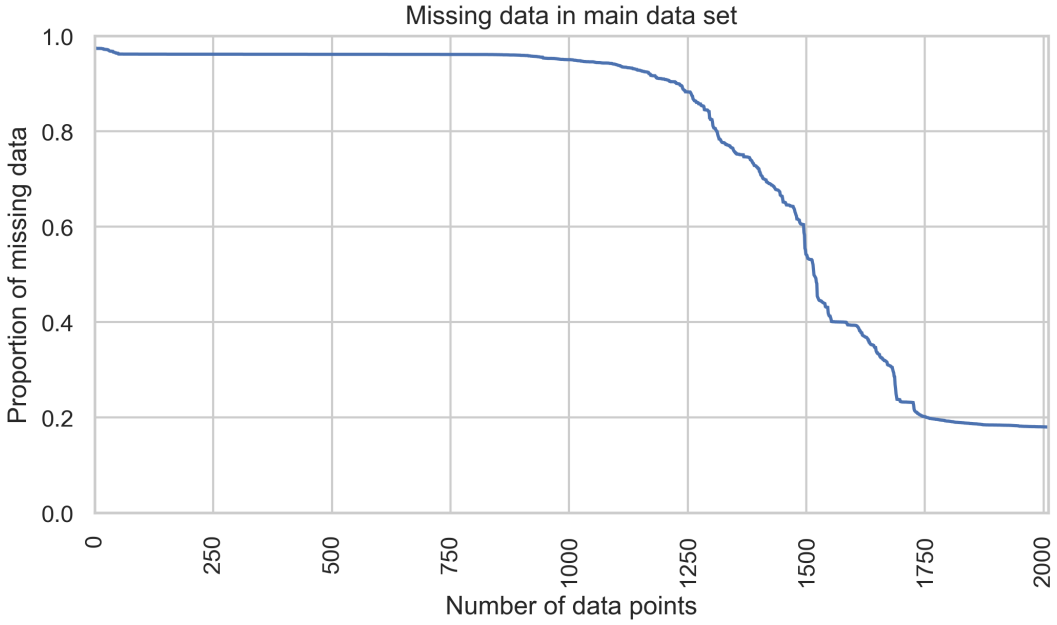

Figure 4.4: Quality report of main data set

Table 4.3 shows the proportion of time stamps containing recorded values among all classes in the main data set. Class 0 (temperature) is the only class where in the examined time period more than 50% of the time stamps contained recorded values in the data base. All other classes fall between 3.8% - 30.3%, most of them closer to the lower end of the spectrum.

Table 4.3: Proportion of time stamps with recorded data per class in the main data set

Class	Recorded data available
0 - temperature	52.1%
1 - volumeflow	16.0%
2 - pressure	3.8%
3 - efficiency	4.9%
4 - on	4.6%
5 - coolingenergy	17.8%
6 - heatingenergy	10.6%
7 - angleposition	23.0%
8 - power	30.3%
9 - presence	4.4%

This finding, however, might not be as grim as figure 4.4 and table 4.3 suggest, as sensor data is only updated in the data base when a change in the output occurs. This can result in seemingly missing data when in reality the output simply stayed constant. Classes that include BACS data points with mostly constant or binary outputs, such as set point settings, status messages or sensors such as presence detectors or switches are especially prone to this error in data quality.

Examining the average number of unique values of the data points of each class gives an indication of which classes in the main data set can be effected by the above described phenomena, shown in table 4.4.

Table 4.4: Average number of unique values of the data points in each class

Class	Avg. unique values
0 - temperature	3617.00
1 - volumeflow	1054.25
2 - pressure	1.26
3 - efficiency	20.60
4 - on	1.75
5 - coolingenergy	1342.50
6 - heatingenergy	661.10
7 - angleposition	1431.00
8 - power	2486.80
9 - presence	4.38

Table 4.4 shows, that Classes 2, 3, 4 and 9 have on average much lower number of unique values than the other classes. This indicates possible binary or constant output of most of the data points belonging to these four classes.

Upon further examination of the data points of each class regarding their minimum and

maximum values, the number of unique values, possible sensor failures and the findings previously presented in this chapter the classes can be described as follows:

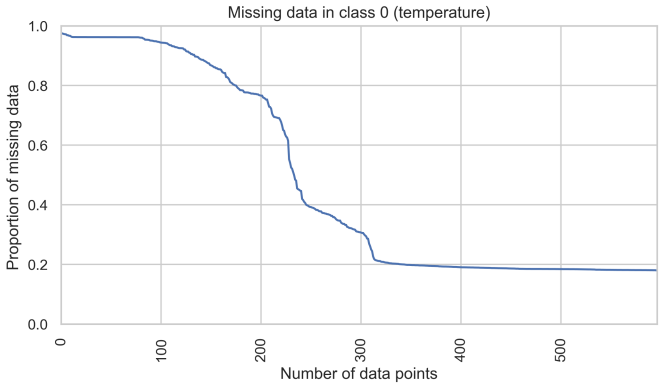

Figure 4.5: Quality report class 0

Class 0 (temperature) is the most represented class in the main data set with 596 data points. It is the class with the most recorded data available (52.1%) and the with largest average amount of unique values (3617). Data points in this class take up values between 0 - 49. However, in case of 24 of the data points physically impossible negative values were recorded indicating possible sensor error. About 12% of data points in this class take up only one constant value (with recorded data of only 3% among all time stamps) throughout the entire time period indicating set point behavior. The remaining data points have recorded data available up to 82% of the time in the requested time period and up to 7200 unique values.

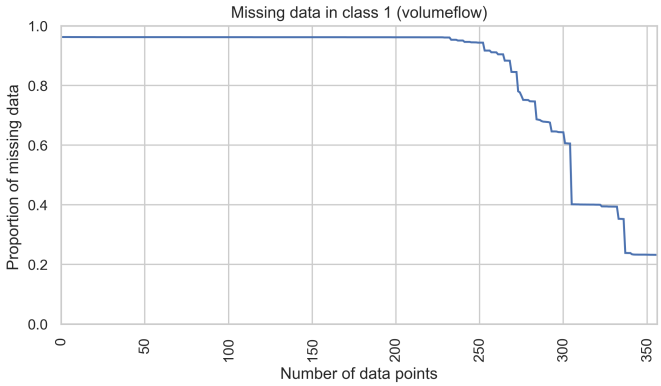

Figure 4.6: Quality report class 1

Class 1 (volumeflow) is the second most represented class in the main data set with 356 data points. 16% of all time stamps contained recorded data in the time period of the main data set with an average number of unique values of 1054.25. Data points in this class take up values between 0 - 105. However, in case of 12 of the data points physically impossible negative values were recorded indicating possible sensor error. About 11% of data points in

this class take up only two unique values showing binary data point behavior. The remaining data points have recorded data available up to 76% of the time in the requested time period and up to 6700 unique values.

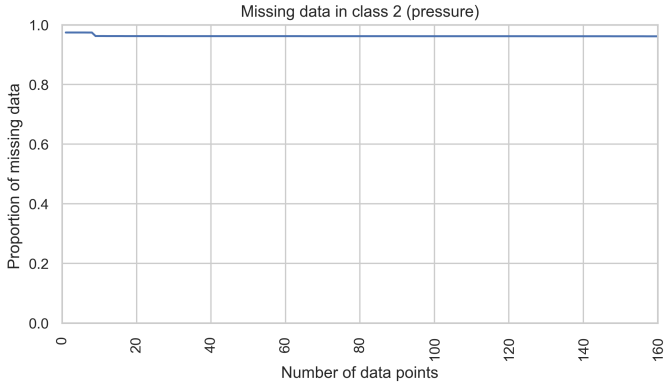

Figure 4.7: Quality report class 2

Class 2 (pressure) is represented with 160 data points in the main data set. Only 3.8% of all time stamps contained recorded data in the time period requested with an average number of unique values of 1.26. Data points in this class take up values between 0 - 718. No indication of sensor error was found. 99.3% of data points in this class take up either one or two unique values indicating set point and binary data point behavior.

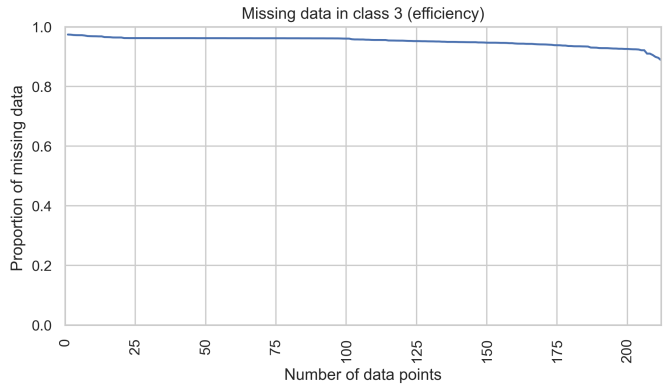

Figure 4.8: Quality report class 3

Class 3 (efficiency) is the third most represented class in the data set with 212 data points. Only 4.9% of all time stamps contained recorded data in the time period requested with an average number of unique values of 20.6. Data points in this class take up values between 0 - 120. No indication of sensor error was found. 18.8% of data points in this class take up one unique value. The remaining data points had recorded data available only up to 11% of the time in the requested time period and up to 300 unique values.

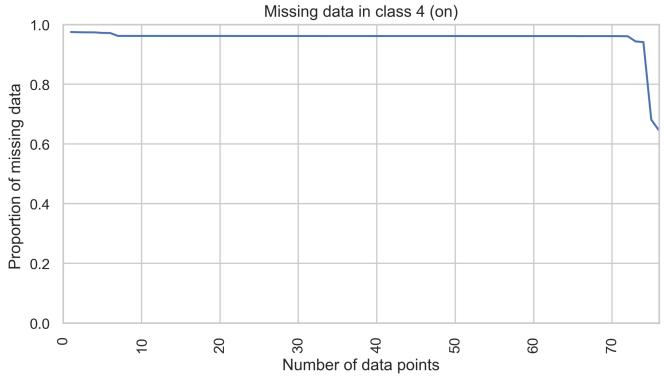

Figure 4.9: Quality report class 4

Class 4 (on) is represented in the data set with 76 data points. Only 4.6% of time stamps contained recorded data in the time period requested with an average number of unique values of 1.75. Data points in this class take up values between 0 - 1. No indication of sensor error was found. 95% of data points in this class take up one unique value staying constant for the entire time period. The remaining four data points had recorded data available up to 35% in the requested time period and up to 30 unique values.

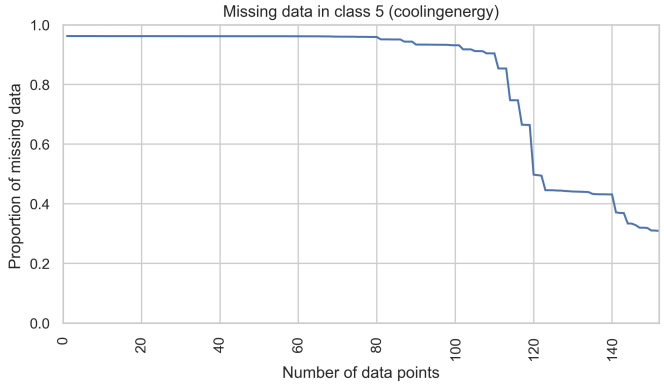

Figure 4.10: Quality report class 5

Class 5 (coolingenergy) is represented in the data set with 152 data points. Only 17.8% of all time stamps contained recorded data in the time period requested with an average number of unique values of 1342.5. Data points in this class take up values between 0 - 35,286. No indication of sensor error was found. 31.6% of data points in this class take up only one unique value staying constant for the entire time period indicating set point behavior. The remaining data points had recorded data available up to 69% of the time in the requested time period and up to 6060 unique values.

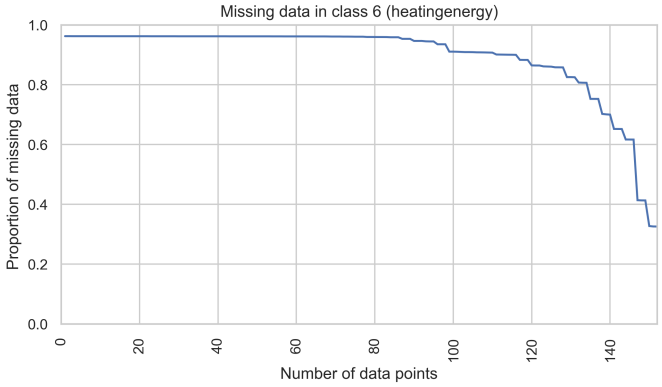

Figure 4.11: Quality report class 6

Class 6 (heatingenergy) is represented in the data set with 152 data points. Only 10.6% of time stamps contained recorded data in the time period requested with an average number of unique values of 661.1. Data points in this class take up values between 0 - 182,351. No indication of sensor error was found. 29.6% of data points in this class take up only one unique value staying constant for the entire time period indicating set point behavior. The remaining data points had recorded data available up to 68% of the time in the requested time period and up to 5900 unique values.

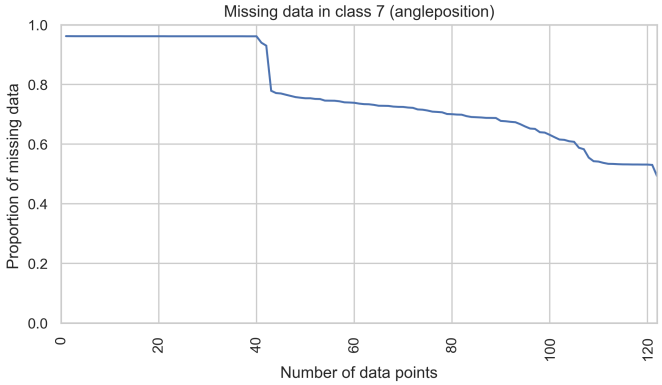

Figure 4.12: Quality report class 7

Class 7 (angleposition) is represented in the data set with 122 data points. Only 23% of all time stamps contained recorded data in the time period requested with an average number of unique values of 1431. Data points in this class take up values between -1 - 100. 32.7% of data points in this class take up unique values between two and four indicating set point behavior. Only in case of these data points, recorded data in all time stamps was under 4% and the unique value -1 occurred. The remaining data points have a data recorded data available up to 40% of the time in the requested time period and up to 4000 unique values.

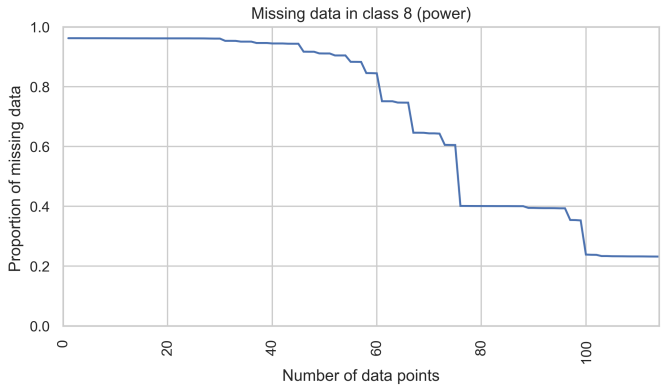

Figure 4.13: Quality report class 8

Class 8 (power) is represented in the data set with 114 data points. 30.3% of all time stamps included recorded data in the time period requested with an average number of unique values of 2486.8. Data points in this class take up values between 0 - 31. No indication of sensor error was found. 23.7% of data points in this class posses up to four unique values indicating set point behavior. The remaining data points had recorded data available up to 76% of the time in the requested time period and up to 6700 unique values.

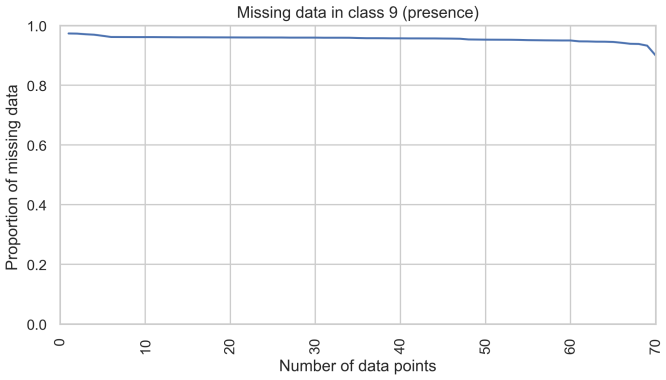

Figure 4.14: Quality report class 9

Class 9 (presence) is the least represented class in the data set with 70 data points. Only 4.4% of all time stamps contained recorded data in the time period requested with an average number of unique values of 4.38. Data points in this class take up values between 0 - 1. No indication of sensor error was found. Data points in this class posses up to 11 unique values, this is most likely due to the data base calculating the fraction of an hour when the sensor was active.

4.4.2 Conclusions - data quality

- Without using any built-in interpolation feature of the Aedifion data base when acquiring the data, 72% of all time stamps included no recorded values.
- Data points of classes with low number of unique (mostly constant or seldom changing) values are especially effected, as among these classes only 3.8 - 4.9% of all time stamps contain recorded data.
- Classes 4 (on) and 9 (presence) appear very similar, both take up values in the same range, have really low percentage of recorded values and have similarly low number of unique values.
- Classes 5 (coolingenergy) and 6 (heatingenergy) appear also similar, both taking up large number of unique values in a wide range.
- Class 0 (temperature) represents close to 30% of all data points with the most amount of recorded available data. It is expected that without using any pre-processing methods for filling the empty time stamps, any classifier algorithm will be heavily biased towards this class.

4.5 Data set format and pre-processing

After establishing the collection of data points and classes to be used as data sets for supervised time series classification, the data was transformed and packed into a format usable by classifier algorithms of the EBC supervised learning python package.

4.5.1 Formatting

Data pre-processing involves transforming the raw data acquired from the database and turning it into a format that can be used by the desired algorithm. In case of the EBC Supervised Learning package the classifier algorithms expect data sets to be separated into two groups of two files. A group of training files (“X_train.pkl” and “y_train.npy”) and a group of testing files (“X_test.pkl” and “y_test.npy”), as it is common in any supervised learning application. These files include respectively the part of the data set that is selected to train a classifier and a part of the data set to test a classifier.

The proportions of data points in the two groups are generally kept at 70% training and 30% testing data throughout this study. Furthermore, before the training and testing files were created, the order of the data points were randomized to avoid any influences resulting from the order in which the data was acquired from the data base.

The X_train and X_test files contain data in a shape of an n x m matrix (example shown in equation 4.1). Each row of the matrix corresponds to a data point and each column in that row contains the data from the first recorded time period to the last.

$$X_{\text{train}} = \begin{bmatrix} X_{11} & \dots & X_{1\text{m}} \\ \vdots & & \vdots \\ X_{\text{n}1} & \dots & X_{\text{nm}} \end{bmatrix} \tag{4.1}$$

The y_train and y_test files contain data in a shape of an n-dimensional column vector (example shown in equation 4.2). Each element, or row, of the vector corresponds to a data point in the same order as the X_train and X_test files, but only includes one column, that contains the numerical representation, an integer between 0 and 9, of the class the corresponding data point belongs to.

$$y_{\text{train}} = \begin{bmatrix} y_1 \\ \vdots \\ y_{\text{n}} \end{bmatrix} \tag{4.2}$$

Naturally, the number of rows of the train and test files must be equal. To generate these data sets, a simple algorithm was created that combined the numerical representation of the pre-defined classes discussed in Chapter 4.3 with the acquired data from the Aedifion database discussed in Chapter 4.1. and generated the 4 files needed to run the classifiers.

4.5.2 Filling of missing data

As later discussed in Chapter 5, missing data in the data sets resulted in overly inaccurate classification, or simply a termination with an error in case of most tested classifiers. Therefore, as part of pre-processing, filling of the missing data was introduced.

At the time of this thesis, the developer version of the EBC supervised learning python package offered 2 types of simple filling methods: forward filling and backward filling.

Forward Filling: Starting from the first index of the time series data, every missing value is filled with the last existing value before it.

Backward Filling: The opposite of forward filling, starting from the last index of the time series data going towards the first index, every missing value is filled with the last existing value ahead of it.

The effects of the use of forward and backward filling on the results of the tested supervised time series classification algorithms are discussed in section 5.2.

5 Investigations and results

This chapter discusses the details and results of supervised time series classification algorithms tested on the data sets of the E.ON ERC building. All data sets are derived from the 2010 individual data points of the main data set introduced in chapter 4.

In the first trial, two state-of-the-art time series classification approaches, the ROCKET and the MiniRocket transformation, are compared with conventional time series classification algorithms in terms of their computational expense. The effects of influencing factors such as data filling, over-representation, sampling rate and merging of classes are then examined using the ROCKET and MiniRocket algorithms with three different classifiers (RFC, ridge and SGD) through four different trials. Finally, a comparison of the best performing classifier with the latest version of the language based classification approach [Krieger, 2021] is presented.

All tested supervised time series classification algorithms are available in the EBC supervised learning python package, which is based on the sktime unified framework for machine learning.

5.1 Data sets and hardware

In each of the following trials a different data set is used. The details of each data set are introduced before each trial, specifying the name, the type of pre-processing used, the number of data points, the classes used, the time span, the sample rate, the ratio of data points in the test and train data sets and a short description.

All trials were done on a basic desktop computer, running Windows 10 with hardware specifications shown in table 5.1.

Table 5.1: Hardware

CPU	Intel Core i5-4590 CPU @ 3.30 Ghz
GPU	NVIDIA GeForce GTX 1660 Ti
Hard drive	WD Green 1TB SSD
Ram	16 Gb DDR3 1600 Mhz

All data sets and results as well as the detailed system information are available in the digital documentation.

5.2 Trial 1 - computational time

Long computational time and required processing power are both bottlenecks when it comes to solving time series classification problems in real-life applications. Previous studies conducted on similar data sets ([Bode et al., 2019],[Fütterer et al., 2017]) were mostly restricted to shorter time frames such as one day with one-minute time samples or a week with fifteen minute time samples for this exact reason. The ROCKET and MiniRocket algorithms promise a great reduction in computational time [Bagnall et al., 2017].

In the first trial, three conventionally used time series classification algorithms, the "multilayer perceptron" (MLP), the "inception" and the "tsfresh" are compared with a ROCKET and MiniRocket algorithms using the "ridge" classifier.

The data set used in this trial was kept simple, only including seven days of data of ten data points from classes 6 and 8. Forward filling was used during pre-processing, as all classifier algorithms except for the MiniRocket terminated with an error when presented with empty time stamp values. Table 5.2 shows the details of the data set used in trial 1.

Table 5.2: Data set for trial 1

Name	Trial 1
Pre-processing	Forward fill
Classes used	6, 8
Time start	2020-01-01 00:00:00
Time end	2020-01-07 00:00:00
Sample rate	1 hour
Number of data points	10
Test / train ratio	3 / 7

As only ten data points were used in this trial, metrics such as F1-score or MCC were highly distorted due to the lack of training data and were not considered in the evaluation. The sole purpose was to examine the differences in computational time. Each classifier algorithm ran five times on the data set, the results are shown in table 5.1.

Both MiniRocket and ROCKET finished in well under one second with the classification task, in 0.167 and 0.324 seconds on average respectively. Tsfresh finished on third place with 6.611 seconds on average, approximately 40 times slower than MiniRocket. MLP finished with 20.120 seconds on average. The inception classifier was the slowest with 92.02 seconds on average, 551 times slower than the MiniRocket.

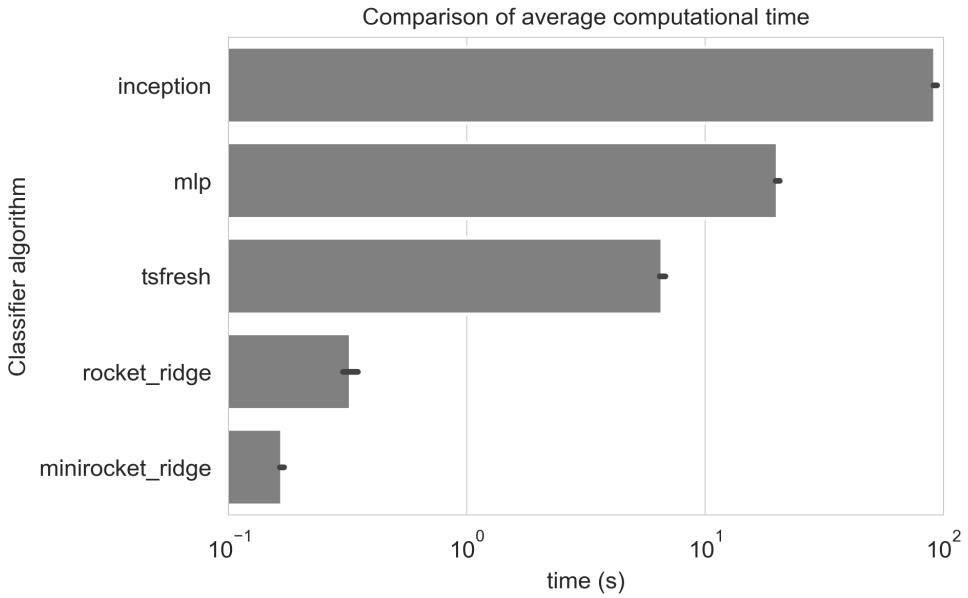

Figure 5.1: Trial 1 results

5.2.1 Conclusions - trial 1

- The ROCKET and MiniRocket algorithms far outperformed the conventional classifier algorithms in terms of computational time on the tested data set.

5.3 Trial 2 - influence of missing data and data filling

The need for pre-processing the raw data acquired from the Aedifion data base became apparent when first testing out the classifiers. All tested algorithms except for iterations of the MiniRocket terminated with an error when presented with "NaN", or empty, time stamp values, which make up 72% of the main data set (discussed in chapter 4).

In this trial, the influence of two, in the EBC supervised learning package available data filling methods (forward filling and backward filling), are tested and compared with the results of the data set containing unchanged raw data. Three data sets were used, covering the data of an entire year of the 2010 data points and ten classes of the main data set with one hour sample rate. The only difference is the pre-processing method. Trial 2a uses no data filling, trial 2b uses backward filling and trial 2c uses forward filling.

In comparison to the data set used in trial 1, this data set includes about 200 times more data points and a 52 times longer time period, equating to more than 10,000 times more individual time stamps that the algorithms have to process.

5.3.1 Trial 2a

The data set used in trial 2a contains only the unchanged raw data, as it was acquired from the Aedifion database. In this trial, only the MiniRocket algorithm was tested with the RFC, ridge and SGD classifiers, as all other algorithms (including the conventional ones tested in trial 1) were unable to run due to missing data errors. Table 5.3 shows the details of the data set used.

Table 5.3: Data set for trial 2a

Name	Trial 2a
Pre-processing	-
Classes used	0,1,2,3,4,5,6,7,8,9
Time start	2020-01-01 00:00:00
Time end	2021-01-01 00:00:00
Sample rate	1 hour
Number of data points	2010
Test / train ratio	603 / 1407

Each classifier algorithm ran five times on the data set, the results are shown in table 5.4. It is apparent, that all three iterations of the MiniRocket algorithm delivered results with the exact same metrics. Overall very poor macro F1-score of 0.0467 and the MCC of 0 show that the classifiers performed worse than a simple random distribution would have. Equal accuracy and the precision and recall micro values of 0.3051 indicate that all correctly predicted classes were true positive and no correct true negative values were predicted. Considerably better micro averages suggest that the results are biased by over-representation of one or more classes and that the classifiers were much more successful at predicting these. This could also mean that all data points were predicted to be of one single class.

Table 5.4: Average metrics of trial 2a, five tests per classifier

Classifier	MiniRocket RFC	MiniRocket Ridge	MiniRocket SGD
F1 macro	0.0467	0.0467	0.0467
F1 micro	0.3051	0.3051	0.3051
Precision macro	0.0305	0.0305	0.0305
Precision micro	0.3051	0.3051	0.3051
Recall macro	0.1	0.1	0.1
Recall micro	0.3051	0.3051	0.3051
Accuracy	0.3051	0.3051	0.3051
MCC	0	0	0
Duration (s)	10.34	9.35	10.34

All three classifiers produced identical confusion matrices on the test data, shown in figure

5.2. Examining the confusion matrix confirms, that indeed all data points were predicted by the classifiers to be of class 0 (temperature). Bias towards class 0 was previously anticipated due to the over-representation of its data points in the data set and the disproportionately large amount of missing data of the data points of all other classes. It can be concluded, that raw data, as acquired from the Aedifion data base, is not adequate for supervised time series classification with the tested classifier algorithms.

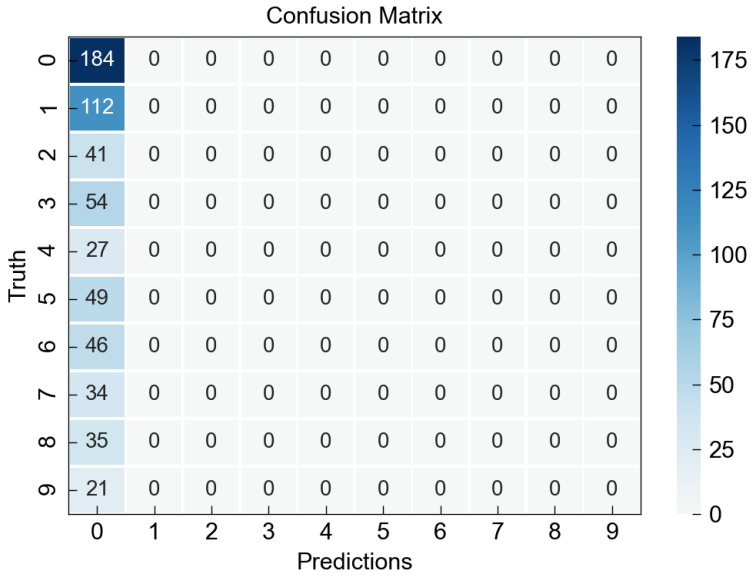

Figure 5.2: Confusion matrix of trial 2a

5.3.2 Trial 2b

Table 5.5: Data set for trial 2b

Name	Trial 2b
Pre-processing	Backward filling
Classes used	0,1,2,3,4,5,6,7,8,9
Time start	2020-01-01 00:00:00
Time end	2021-01-01 00:00:00
Sample rate	1 hour
Number of data points	2010
Test / train ratio	603 / 1407

The data set used in trial 2b was pre-processed using the backward filling data filling method offered by the EBC supervised learning package. Every other aspect of the data set was identical to the one used in trial 2a. Table 5.5 shows the details of the data set. In this trial,

the ROCKET and the MiniRocket algorithms were tested with the RFC, ridge and SGD classifiers. Each classifier algorithm ran five times on the data set, the results are shown in two tables.

Table 5.6: Average metrics of the MiniRocket algorithm in trial 2b, five tests per classifier

Classifier	MiniRocket RFC	MiniRocket Ridge	MiniRocket SGD
F1 macro	0.7577	0.7454	0.6167
F1 micro	0.7874	0.7844	0.6966
Precision macro	0.8025	0.8097	0.7262
Precision micro	0.7874	0.7844	0.6966
Recall macro	0.7392	0.7191	0.6197
Recall micro	0.7874	0.7844	0.6966
Accuracy	0.7874	0.7844	0.6966
MCC	0.7478	0.7445	0.6507
Duration (s)	10.42	9.88	13.42

Table 5.6 shows the results of the iterations of the MiniRocket algorithm. The results show a massive improvement of all metrics when compared to the results of trial 2a. The MiniRocket RFC classifier showed the best performance, closely followed by the MiniRocket ridge, falling behind in all metrics except for the macro precision with only about 0.5% on average while finishing the fastest among all three. The MiniRocket SGD classifier fell behind with about 10% on average in all metrics compared to the RFC while being the slowest. All three iterations finished incredibly fast, considering that in trial 1 the MLP and inception classifiers took 20.120 seconds and 92.02 seconds respectively to train and predict on a data set with only ten data points and one week time span.

Slight differences between the macro and micro averages suggest, that while over-representation bias is present in the data set, it is not as prevalent as in trial 2a.

Table 5.7: Average metrics of the ROCKET algorithm in trial 2b, five tests per classifier

Classifier	ROCKET RFC	ROCKET Ridge	ROCKET SGD
F1 macro	0.6352	0.6638	0.3670
F1 micro	0.7088	0.7055	0.4859
Precision macro	0.7183	0.7256	0.4997
Precision micro	0.7088	0.7055	0.4859
Recall macro	0.6024	0.6527	0.4086
Recall micro	0.7088	0.7055	0.4859
Accuracy	0.7088	0.7055	0.4859
MCC	0.6533	0.6630	0.4182
Duration (s)	550.45	566.85	547.26

Table 5.7 shows the results of the iterations of the ROCKET algorithm. The results show a massive improvement of all metrics when compared to the results of trial 2a, however slightly falling behind all the iterations of the MiniRocket algorithm, while being up to 57 times slower. In trial 2b the ROCKET RFC and ridge classifiers showed very similar results. The ROCKET SGD classifier fell behind with about 23% on average in all metrics compared to the RFC and the ridge but finishing a couple seconds faster.

The best result of trial 2b in terms of metrics was achieved in the third run of the MiniRocket RFC classifier. The normalized confusion matrix in figure 5.3 shows the outcome of the classification.

Classes 0, 1, 3, 7 and 9 were all correctly classified more than 88% of the time and up to 100% in case of class 7. There are, however, certain dynamics between classes that were unexpected during the data analysis in chapter 4. Although the data points of class 1 (volumeflow) were correctly classified 88% of the time, false predictions of data points from other classes occurred, in the majority of the time, due to being falsely classified as class 1. Another previously unexpected co-relation was shown between class 0 (temperature) and class 4 (on). All falsely classified data points of class 0 were predicted as class 4 and the majority of falsely predicted data points of class 4 were predicted as class 0, even though their characteristics vastly differ. Finally, data points of class 5 (coolingenergy) and class 6 (heatingenergy) were falsely classified as class 1, 27% and 24% of the time, as well as misunderstood for each other in 8% and 22% percent of the time respectively. This, however, could be expected as classes 5 and 6 bear many similarities with each other and also with class 1.

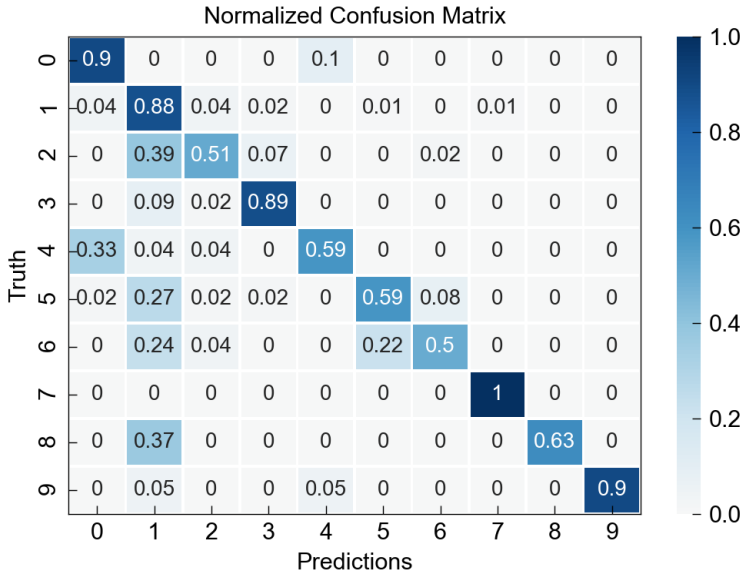

Figure 5.3: Normalized confusion matrix of trial 2b

5.3.3 Trial 2c

The data set used in trial 2c was pre-processed using the forward filling data filling method offered by the EBC supervised learning package. Every other aspect of the data set was identical to the one used in trial 2a and trial 2b. Table 5.8 shows the details of the data set. In this trial, the ROCKET and the MiniRocket algorithms were tested with the RFC, ridge and SGD classifiers.

Table 5.8: Data set for trial 2c

Name	Trial 2c
Pre-processing	Forward filling
Classes used	0,1,2,3,4,5,6,7,8,9
Time start	2020-01-01 00:00:00
Time end	2021-01-01 00:00:00
Sample rate	1 hour
Number of data points	2010
Test / train ratio	603 / 1407

Each classifier algorithm ran five times on the data set, the results are shown in two tables. Table 5.9 shows the results of the iterations of the MiniRocket algorithm. The results show a massive improvement of all metrics when compared to the results of trial 2a and a slight improvement when compared to trial 2b. The MiniRocket RFC classifier showed the best performance, closely followed by the MiniRocket ridge, falling behind in all metrics with about 2.14% on average while finishing the fastest among all three. The MiniRocket SGD classifier fell behind with about 12,12% on average in all metrics compared to the RFC while being the slowest. All three iterations finished incredibly fast, although slightly slower than in trial 2b.

Table 5.9: Average metrics of the MiniRocket algorithm in trial 2c, five tests per classifier

Classifier	MiniRocket RFC	MiniRocket Ridge	MiniRocket SGD
F1 macro	0.7247	0.7000	0.6173
F1 micro	0.8103	0.7910	0.6600
Precision macro	0.7854	0.7622	0.7391
Precision micro	0.8103	0.7910	0.6600
Recall macro	0.7030	0.6798	0.6365
Recall micro	0.8103	0.7910	0.6600
Accuracy	0.8103	0.7910	0.6600
MCC	0.7760	0.7525	0.6269
Duration (s)	11.36	10.54	14.51

Micro averages outperformed all macro counterparts, showing that predictions about over-

represented classes were slightly more accurate. When comparing with trial 2b, macro averages are slightly lower in trial 2c.

Table 5.10 shows the results of the iterations of the ROCKET algorithm. The results show a massive improvement of all metrics when compared to the results of trial 2a and a slight improvement in all metrics compared to trial 2b.

Table 5.10: Average metrics of the ROCKET algorithm in trial 2b, five tests per classifier

Classifier	ROCKET RFC	ROCKET Ridge	ROCKET SGD
F1 macro	0.6870	0.6907	0.5188
F1 micro	0.7602	0.7187	0.6405
Precision macro	0.7865	0.7740	0.5758
Precision micro	0.7602	0.7188	0.6405
Recall macro	0.6605	0.6886	0.5409
Recall micro	0.7602	0.7187	0.6405
Accuracy	0.7602	0.7187	0.6055
MCC	0.7270	0.6992	0.5940
Duration (s)	556.64	539.44	547.26

The ROCKET RFC classifier showed the best performance, closely followed by the ROCKET ridge, falling behind in all metrics except for the F1 macro with only about 2.18% on average while finishing the fastest among all three. The ROCKET SGD classifier fell behind with about 14% on average in all metrics compared to the RFC while being the slowest. All three iterations of the ROCKET were up to 55 time slower than the MiniRocket.

The best result of trial 2c in terms of metrics was achieved in the fifth run of the MiniRocket RFC classifier. The normalized confusion matrix in figure 5.4 shows the outcome of the classification.

Although on average, using forward filling, the MiniRocket RFC delivers slightly better metrics than when using backward filling, examining the normalized confusion matrix of the best result of trial 2c (shown in figure 5.4) reveals a similar but more pronounced trend regarding class 1 just like in trial 2b.

Overall the classes 0 (temperature), 1 (volumeflow), 3 (efficiency), 7 (angleposition) and 8 (power) were all correctly predicted more than 88% of the time, up to 96% in case of class 1. Using forward filling caused the classifier in case of 73% of the data points of class 2 (pressure) and 100% of the data points of class 4 (on) to be falsely predicted as members of class 1. This shows, that forward filling essentially made two classes completely blend into a third class, even though, as shown in chapter 4 during the analysis of classes, class 1 vastly differs from class 2 and 4 in terms of the value range and overall behavior.

The dynamics between class 5 (coolingpower) and class 6 (heatingpower) changed in trial 2c.

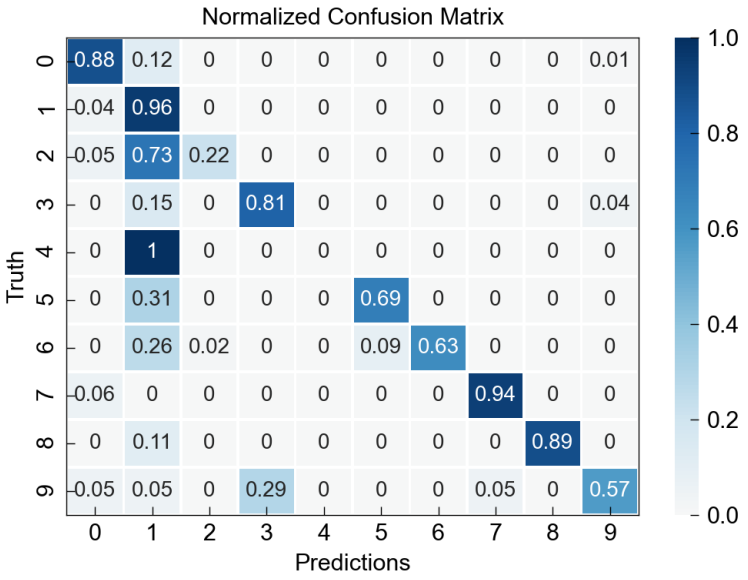

Figure 5.4: Normalized confusion matrix of trial 2c

Correct predictions on these two classes were up by 10% and 13% reaching 69% and 63% respectively. They were less likely to be falsely predicted as each other, but more likely to be falsely predicted as class 1. The only class whose false predictions as class 1 reduced during trial 2c was class 8 (power).

5.3.4 Highlights - trial 2

During the tests of trial 2, the MiniRocket algorithm outperformed the ROCKET in all cases in terms of both all metrics and computational time. Using the MiniRocket with the RFC classifier resulted in slightly better metrics, while using the ridge was faster every time. Depending on the application of the supervised time series classification, the duration of the classification might be of concern. In case of the tasks preformed in this thesis, the differences in the already very short duration is not a deciding factor.

Comparing the two tested data filling methods, forward filling and backward filling, the metrics seem to indicate a narrowly better performance when using forward filling. However, upon examination of the confusion matrices of the best results of trial 2b and 2c it became apparent, that forward filling caused the classifier algorithm to falsely predict the majority of two previously distinctly separate classes as a third class, as well as distorting the results of multiple other classes compared to backward filling. In the following trials, backward filling method is recommended.

5.3.5 Conclusions - trial 2

- Even though it was possible to run some supervised time series classification algorithms on data sets with large amounts of missing data, the results were completely biased towards the class with the most data points and data available.
- Data filling during pre-processing can greatly improve the results of supervised classification.
- The data filling method of choice can produce seemingly superior results when only taking metrics into consideration. Confusion matrices help determine if better metrics also translate into better real results regarding the classification task at hand. Backward filling is the recommended data filling method for the remaining trials.
- Seeing the different impact the two very similar data filling methods had on the results of time series classification, it is recommended for future studies to explore with different methods to optimize performance.
- The MiniRocket algorithm outperformed the ROCKET in all tests during this trial in terms of all metrics while being also up to 57 times faster. It is the recommended algorithm for the remaining trials.
- For the classification problems tackled in trial 2, the MiniRocket algorithm with the RFC classifier delivered the best results.

5.4 Trial 3 - influence of over-representation

Over-representation as well as under-representation of one or more classes in the training data used for supervised time series classification can cause bias in favor or against any given class [Brownlee, 2015]. In the previous trials, the distribution of the data points of the ten classes was in the same proportion as they were represented in the BACS of the E.ON ERC building, as the main data set included all available data points.

In trial 3, two different training data sets were tested. One, where all classes are kept at equal proportions in the training data, and one, where the number of data points of class 0 (temperature) were cut to half, while keeping all other classes unchanged.

Both data sets were tested five times but only with the MiniRocket RFC, ridge and SGD algorithms, as trial 2 showed the dominance of the MiniRocket over the ROCKET algorithm in every single test.

5.4.1 Trial 3a

By keeping the number of data points of all classes equal in the training data of trial 3a, the goal was to create a completely over- and under-representation bias-free data set for the classifiers. Details of the data set are shown in table 5.11.

Table 5.11: Data set for trial 3a

Name	Trial 3a
Pre-processing	Backward filling
Classes used	0,1,2,3,4,5,6,7,8,9
Time start	2020-01-01 00:00:00
Time end	2021-01-01 00:00:00
Sample rate	1 hour
Number of data points	700
Test / train ratio	210 / 490

Contrary to expectations, the results (table 5.12) show an overall decrease in performance across the board compared to trial 2b. The MiniRocket RFC performed on average 6.2% worse, the MiniRocket ridge 9.6% worse and the MiniRocket SGD 14.1% worse. Reduction in the data set size resulted in a faster running time for all classifiers.

Table 5.12: Average metrics of the MiniRocket algorithm in trial 3a, five tests per classifier

Classifier	MiniRocket RFC	MiniRocket Ridge	MiniRocket SGD
F1 macro	0.7194	0.6834	0.5099
F1 micro	0.7048	0.6590	0.5248
Precision macro	0.7304	0.7042	0.6041
Precision micro	0.7048	0.6590	0.5248
Recall macro	0.7587	0.7351	0.5729
Recall micro	0.7048	0.6590	0.5248
Accuracy	0.7048	0.6590	0.5248
MCC	0.6714	0.6312	0.4950
Duration (s)	5.47	4.57	5.23

The overall best metrics were achieved in the fifth trial of MiniRocket RFC, whose normalized confusion matrix is shown figure 5.5.

Overall the classes 0 (temperature), 3 (efficiency), 4 (on), 7 (angleposition), 8 (power) and 9 (presence) were all correctly predicted more than 78% of the time, and up to 99.8% in case of class 9 and 100% of class 7. Previously, in trial 2 already observed behavior repeated itself in case of classes 5 (coolingpower) and 6 (heatingpower), as they were often falsely classified as each other. Unlike in the previous trials, this time class 2 (pressure) was the one class most

data points of other classes were falsely classified as instead of class 1. Class 2 is one of the classes with the most amount of missing values as 96.2% of its empty values were replaced by backward filling. This makes it likely, that data points of other classes with large amounts of missing values could be easily misunderstood by the classifier algorithm as class 2.

Overall worse results than previously achieved could party be attributed to the reduced number of data points in the data set.

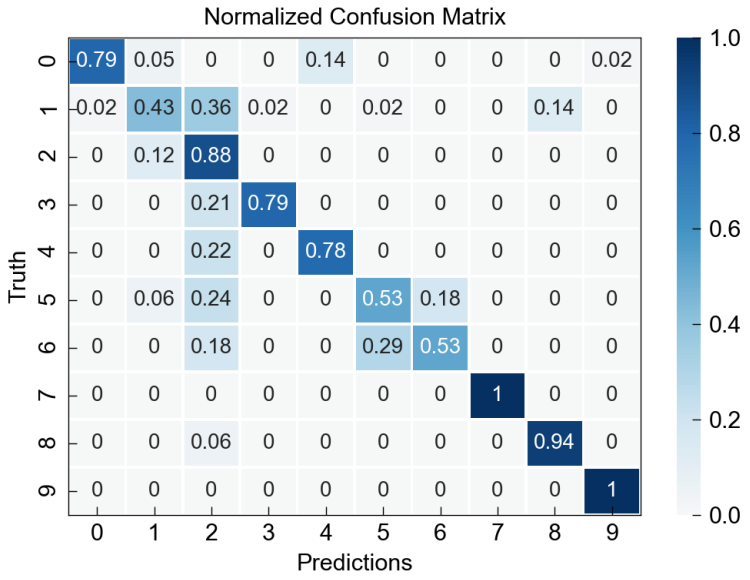

Figure 5.5: Normalized confusion matrix of trial 3a

5.4.2 Trial 3b

The data set of trial 3b represents a less radical approach in trying to mitigate the effects of over-representation. In this trial, half of all data points of class 0 were removed from the training data set compared to trial 2, but all other classes remained unchanged. Normally, class 0 represented about 30% of all data points in the training data (see figure 4.3). The new distribution is show in figure 5.6, x-axis was kept the same for easier comparison.

The details of the data set used in trial 3b are shown in table 5.13.

The MiniRocket RFC showed the overall best performance in all metrics other than precision macro, where the MiniRocket ridge reached a 1.6% higher score. The MiniRocket SGD lagged behind in all performance indicators while being the slowest, similar to previous trials. On average, the MiniRocket tested with the RFC classifier in trial 3b outperformed the same algorithms tested in trial 2b, showing an average performance increase of 1,48%. Among all trials so far, this one showed the overall best performance.

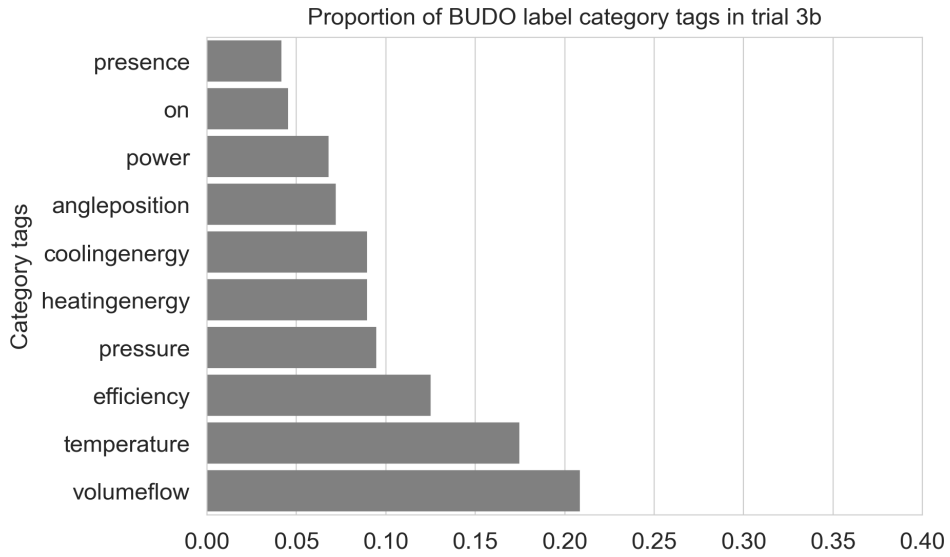

Figure 5.6: Altered proportions of classes in trial 3b data set

Table 5.13: Data set for trial 3b

Name	Trial 3b
Pre-processing	Backward filling
Classes used	0,1,2,3,4,5,6,7,8,9
Time start	2020-01-01 00:00:00
Time end	2021-01-01 00:00:00
Sample rate	1 hour
Number of data points	1715
Test / train ratio	515 / 1200

The best results in trial 3b were achieved during the third run of the MiniRocket RFC classifier. The results are shown in figure 5.7

The normalized confusion matrix of trial 3b shows improvement over, but overall a similar behavior, to trial 2b. Data points from the classes 0, 1, 4, 7, 8 and 9 were all more than 86% of the time correctly classified and up to 100% with class 7. Most classes were falsely classified as class 1 and the similarities between classes 5 and 6 are well-pronounced.

5.4.3 Conclusions - trial 3

- Mitigation of over-representation of a class in the training data showed improvement in the overall performance of the MiniRocket RFC and ridge classifiers.
- Reducing the number of data points in the training data to completely eliminate over-representation can be counter-productive depending on the number of data points re-

Table 5.14: Average metrics of the MiniRocket algorithm in trial 3b, five tests per classifier

Classifier	MiniRocket RFC	MiniRocket Ridge	MiniRocket SGD
F1 macro	0.7877	0.7749	0.6499
F1 micro	0.7825	0.7728	0.6190
Precision macro	0.8274	0.8434	0.7506
Precision micro	0.7825	0.7728	0.6191
Recall macro	0.7801	0.7601	0.6761
Recall micro	0.7825	0.7728	0.6191
Accuracy	0.7825	0.7728	0.6191
MCC	0.7530	0.7450	0.6118
Duration (s)	10.55	9.38	13.62

maining in the data set.

5.5 Trial 4 - Influence of merging classes

Previous studies investigating unsupervised learning algorithms on a similar data sets have shown, that data point clusters found by artificial intelligence often do not match with the human-defined classification of the same data points [Bode et al., 2019]. Data points of two classes, that are often falsely classified as members of the other class likely point to a similar phenomena when using supervised time series classification.

Table 5.15: Data set for trial 4

Name	Trial 4
Pre-processing	Backward filling
Classes used	0,1,2,3,4,-,6,7,8,9
Time start	2020-01-01 00:00:00
Time end	2021-01-01 00:00:00
Sample rate	1 hour
Number of data points	2010
Test / train ratio	603 / 1407

The most pronounced cases in this thesis that could indicate the behaviour described above are shown in figure 5.3 and 5.5. and 5.7. Two pairs of classes: Classes 0 (temperature) and 4 (on) and classes 5 (coolingpower) and 6(heatingpower) were in large proportions falsely classified as each other. In practical applications, it could be beneficial to merge two such classes into one to improve the overall performance of classification algorithms, as long as the merging do not interfere with the overall logic and purpose of the classification problem at hand. In case of this thesis classes 0 (temperature) and 4 (on) show no similarity in terms

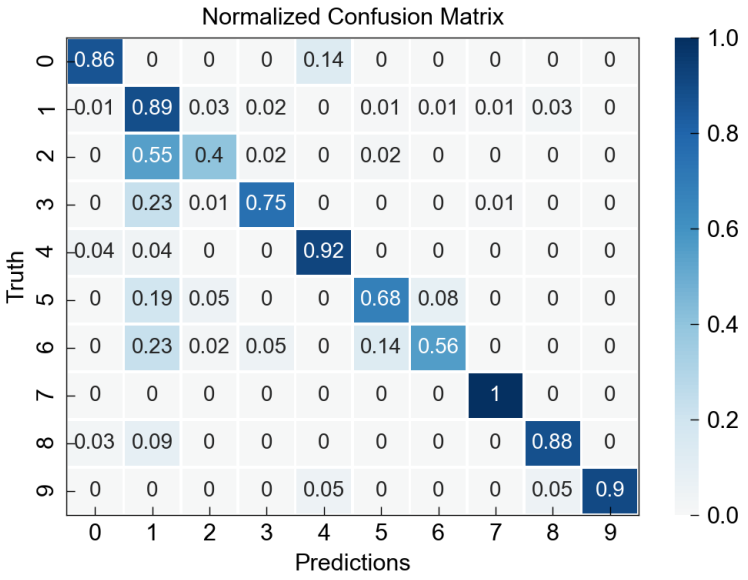

Figure 5.7: Normalized confusion matrix of trial 3b

of the human-defined classification. Classes 5 and 6, however, describe very similar data points and sensors measuring the same type of physical value. To investigate the influence of merging classes on the outcome of supervised time series classification, the data points of these two classes are merged into one single class in the data set (shown in table 5.15) and the outcome is compared to trial 2b, where the data set was prepared, other than the merged classes, identically.

Table 5.16: Average metrics of the MiniRocket algorithm in trial 4, five tests per classifier

Classifier	MiniRocket RFC	MiniRocket Ridge	MiniRocket SGD
F1 macro	0.7755	0.7710	0.6523
F1 micro	0.7878	0.7867	0.6988
Precision macro	0.8289	0.8396	0.7863
Precision micro	0.7877	0.7867	0.6988
Recall macro	0.7509	0.7453	0.6421
Recall micro	0.7877	0.7867	0.6988
Accuracy	0.7877	0.7867	0.6988
MCC	0.7458	0.7455	0.6499
Duration (s)	12.44	11.34	14.36

Table 5.16 shows the average metrics achieved by the classifiers in trial 4. As in the previous trials, MiniRocket RFC delivered on average the best results, only falling behind in one metric, the precision macro, by about 1% compared to MiniRocket ridge. The MiniRocket SGD classifier delivered the lowest metrics and slowest computational time. Overall, the

merging of classes 5 and 6 did not the result in the anticipated performance increase, The overall performance of the MiniRocket RFC classifier was on average better than that of the MiniRocket RFC in trial 2b, but worse than trial 3b, in both cases remaining within 1% margin of difference.

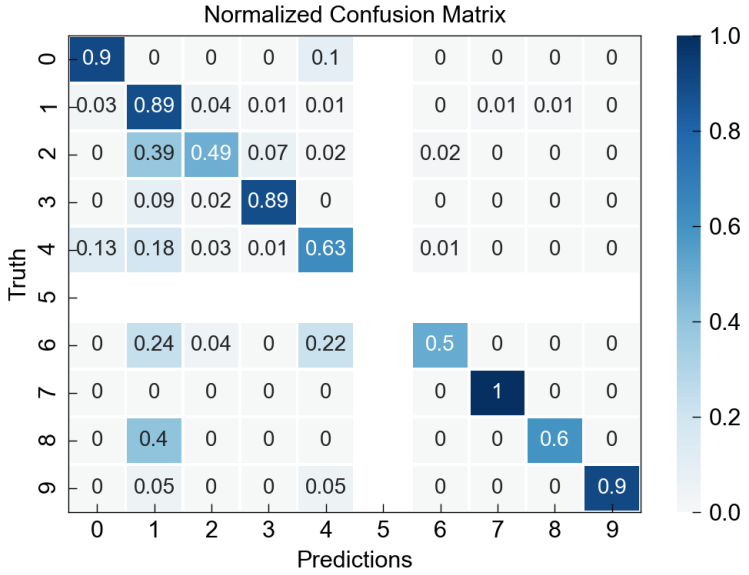

Figure 5.8: Normalized confusion matrix of trial 4

The best results in this trial was achieved in the fourth run of MiniRocket RFC. The results are shown in the normalized confusion matrix in figure 5.8. In terms of the rate of correct classification of the newly merged class 6, no improvement can be observed when compared to the trials 2b and 3b.

5.5.1 Conclusions - trial 4

- Merging of the classes 5 and 6 did not result in a significant increase in performance in terms of metrics, the ratio of data points correctly classified as the new merged class also decreased compared to trial 2 or trial 3 where those classes were kept separately.
- Most likely higher degree of similarity is required between two classes to justify merging.

5.6 Trial 5 - influence of sampling rate - large data

The impact of data set size and data set quality are deterministic factors in the success of machine learning algorithms [Sessions and Valtorta, 2006],[Althnian et al., 2021]. Due to the high computational power requirement of conventional supervised time series classification

algorithms, concessions in similar studies were often made regarding the amount of data points as well as the time span and sampling rate used in the data sets [Fütterer et al., 2017]. The MiniRocket algorithm alleviates this restriction to a large degree due to its vastly superior computational time.

Expanding the sample rate and the time span of a data set allows the classifier to pick up on nuances and more detailed data point behavior that would not be possible with a shorter time span or larger sample rate, but in the same time it enlarges the data the classifier has to process.

In trial 5, the previously used hourly sample rate was changed to 15 minutes in both tested data sets to examine its effects. The three iterations of the MiniRocket classifier were ran five times on each data class.

5.6.1 Trial 5a

Trial 5a was identical in every respect to trial 2b except for the sample rate. Table 5.17 shows the details of the data set. Changing the the sample rate to 15 minutes meant, that every data point consisted of 35,136 individual data stamps instead of the previous 8,784 when using hourly values.

Table 5.17: Data set for trial 5a

Name	Trial 5a
Pre-processing	Backward filling
Classes used	0,1,2,3,4,5,6,7,8,9
Time start	2020-01-01 00:00:00
Time end	2021-01-01 00:00:00
Sample rate	15 min
Number of data points	2010
Test / train ratio	603 / 1407

Table 5.18 shows the results of trial 5a after five runs with each classifier. On average the MiniRocket RFC classifier outperformed the ridge and SGD classifiers by 3% and 12% respectively, dominating the others in every metric as well as computational time. When compared to trial 2b, an increase in performance of 1.24% could be observed, however, trial 3b outperformed trial 5a by an average of 0.24% while finishing only in 13.62 seconds instead of 55.56 seconds.

The best results during trial 5a were achieved on the first run of the MiniRocket RFC algorithm, the results are shown on the normalized confusion matrix in figure 5.9. The normalized confusion matrix shows very similar characteristics when compared to that of trial 2b (figure

Table 5.18: Average metrics of the MiniRocket algorithm in trial 5a, five tests per classifier

Classifier	MiniRocket RFC	MiniRocket Ridge	MiniRocket SGD
F1 macro	0.7703	0.7027	0.6106
F1 micro	0.7983	0.7901	0.6879
Precision macro	0.8154	0.7615	0.7163
Precision micro	0.7983	0.7901	0.6879
Recall macro	0.7551	0.6815	0.6111
Recall micro	0.7983	0.7905	0.6879
Accuracy	0.7983	0.7905	0.6879
MCC	0.7615	0.7503	0.6432
Duration (s)	55.56	55.84	57.32

5.3). The classes are overall well differentiated from one another, except for class 1, as also previously observed.

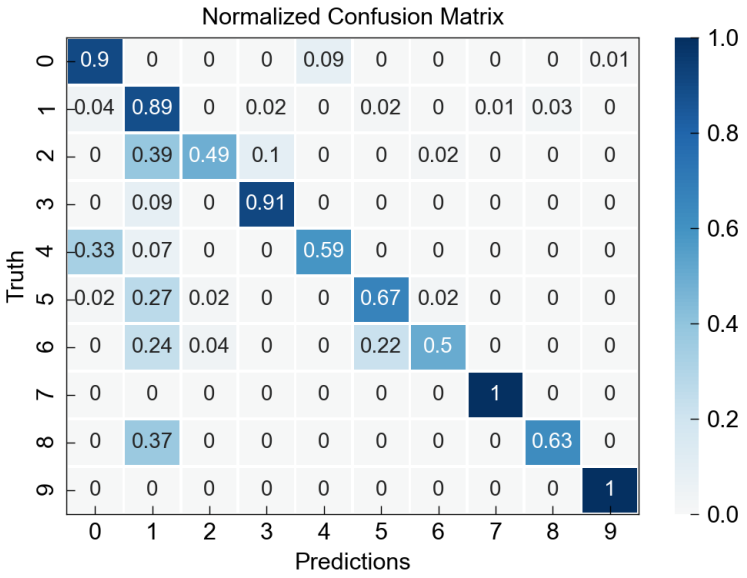

Figure 5.9: Normalized confusion matrix of trial 5a

5.6.2 Trial 5b

In trial 5b, the last trial conducted in this thesis, an attempt was made to include all modifications on the data set that have proven to be positively effecting the outcome of classifier algorithms in all previous trials. The sample rate was kept at 15 minutes, backward filling was used to pre-process the data set and the bias of over-representation was partially removed by halving the number of data points of class 0. In addition to these changes, class 1 was completely removed from the data set. This last step might not be a viable option in practical applications, when the goal is to classify all available data points. Details of the data set used in trial 5b is shown in table 5.19.

Table 5.19: Data set for trial 5b

Name	Trial 5b
Pre-processing	Backward filling
Classes used	0,-,2,3,4,5,6,7,8,9
Time start	2020-01-01 00:00:00
Time end	2021-01-01 00:00:00
Sample rate	15 min
Number of data points	1356
Test / train ratio	407 / 949

Figure 5.10 shows the proportions of classes after the modifications to the data set were made. X-axis is kept constant across figure 5.10, 5.6 and 4.3 to allow for easier comparison.

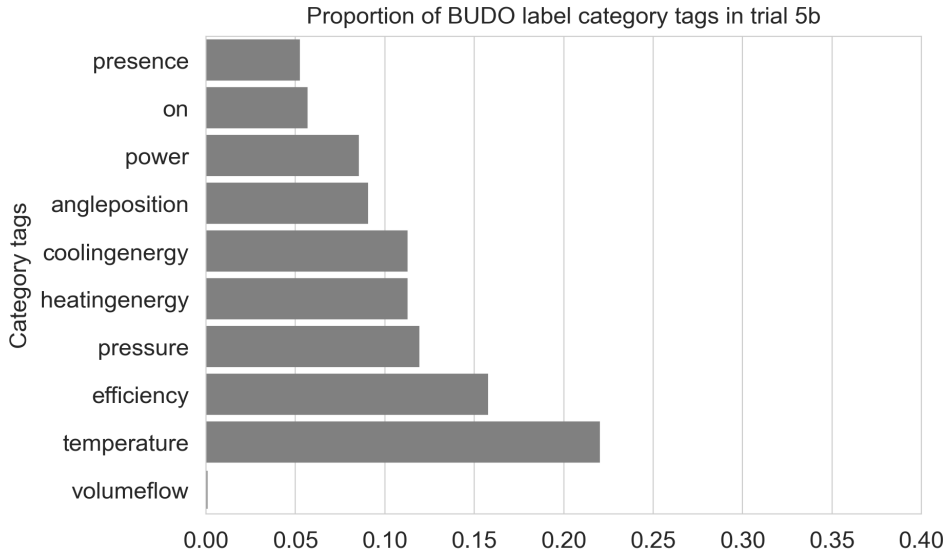

Figure 5.10: Altered proportions of classes in trial 5b data set

The MiniRocket RFC algorithm produced the best results on average, outperforming the

MiniRocket ridge and SGD in all metrics while being also the fastest. Among all trials in this thesis, trial 5b performed on average with 6,13% better than the first comparable trial with backward filling on the main data set (trial 2b)

Table 5.20: Average metrics of the MiniRocket algorithm in trial 5b, five tests per classifier

Classifier	MiniRocket RFC	MiniRocket Ridge	MiniRocket SGD
F1 macro	0.8245	0.8035	0.6498
F1 micro	0.8447	0.8216	0.7233
Precision macro	0.8372	0.8211	0.7134
Precision micro	0.8447	0.8216	0.7233
Recall macro	0.8350	0.8243	0.6691
Recall micro	0.8447	0.8216	0.7233
Accuracy	0.8447	0.8216	0.7233
MCC	0.8109	0.7872	0.6794
Duration (s)	38.11	47.75	39.07

The overall best result was achieved on the fourth run of the MiniRocket RFC classifier, which is visualized on the normalized confusion matrix in figure 5.11. When compared to trial 2b, the proportions of data points correctly classified had increased in all classes. After removing all data points of class 1, similar false classification occurred between class 2 and the classes 3, 5 and 6, as a large proportion of them were falsely classified as class 2. This is to be expected, since in all previous trials class 2 was between 37% - 73% of time falsely classified as class 1, showing a large degree of similarity.

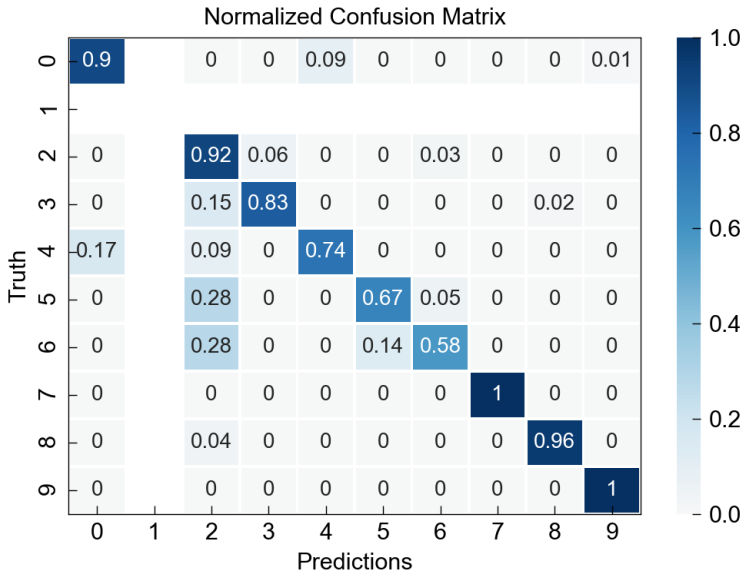

Figure 5.11: Normalized confusion matrix of trial 5b

5.6.3 Conclusions - trial 5

- Higher frequency sampling rate increased the overall performance of the classifiers, however alleviating over-representation bias in trial 3b produced even better results.
- Combining all alterations on the data set that were shown to increase overall performance in previous trial resulted in the best overall performance in trial 5b.

5.7 Comparison with AIKIDO

In this chapter two completely different approaches to classifying data points according to BUDO categories are compared.

The AIKIDO tool is a machine learning algorithm based on natural language processing (NLP), that was designed to read the label of any data point and translate it according to the machine-readable BUDO schema. In previous studies the AIKIDO could achieve between 0.82 - 0.97 accuracy when tested on 5 different buildings [Stinner et al., 2019]. In a recent masters's thesis at the EBC institute, Krieger [Krieger, 2021] further developed the AIKIDO tool to achieve better results by up to 2%. For the comparison in this thesis, this most recent version of the AIKIDO tool is used.

The data sets and pre-determined classes in this thesis are all based on the manual translation of the data point labels of the E.ON ERC building, using the category tags belonging to the "signal type" BUDO category.

In this comparison, the AIKIDO tool and the MiniRocket RFC classifier were both employed on the same data points as in trial 2, with identical training and testing data sets (see table 5.21). The aim was to see how accurately the two methods are able to predict the same class to a data point.

Table 5.21: Data set for trial AIKIDO comparison

Name	Aikido
Pre-processing	Backward filling
Classes used	0,1,2,3,4,5,6,7,8,9
Time start	2020-01-01 00:00:00
Time end	2021-01-01 00:00:00
Sample rate	1 hour
Number of data points	2010
Test / train ratio	603 / 1407

The AIKIDO tool was trained on three subsets of the training data, including 1200, 600 and 300 randomly selected data point labels. The predictions then were made on all the 603 data

points of the test data set. Table 5.22 shows the results of the AIKIDO tool.

Table 5.22: Average metrics of the AIKIDO tool on the comparison data set

Training labels	300	600	1200
F1 macro	0.9957	0.9984	0.9961
Precision macro	0.9946	0.9984	0.9953
Recall macro	0.9969	0.9984	0.9969

The AIKIDO tool reached an overall performance above 99% in terms of every metric when trained on either of the three subsets of data point labels. This success, however, can be partially attributed to the fact that the selection of the main data set in this thesis was based on the manual translation of data point labels according to the BUDO schema. This means, that all data point labels used the test data of this trial had a category tag label within the "signal type" BUDO schema category in the manual translation. The AIKIDO tool was successful at assigning the appropriate tags (classes) in almost all cases.

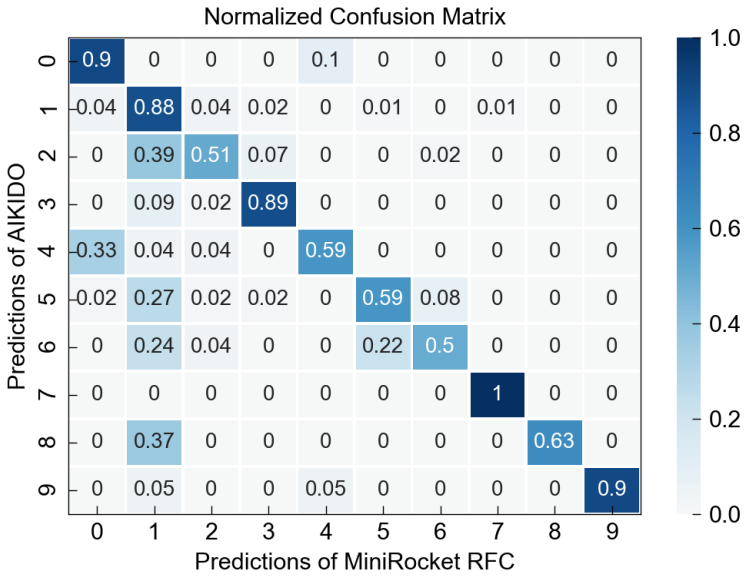

Figure 5.12: Confusion matrix of AIKIDO comparison

In figure 5.12, the results of the comparison are shown in a confusion matrix. In this confusion matrix, both axes correspond to a prediction. The rows show the predictions of the AIKIDO tool using a subset of 600 training labels on the data set, the columns show the predictions of the best result of the MiniRocket RFC classifier from trial 2b.

The results show, that the AIKIDO and the MiniRocket RFC were successful at predicting the same category tags (or classes) more than 88% of the time in case of classes 0, 1, 3 and

9 and up to 100% of the time in case of class 7. The overall results are almost identical to that of trial 2b, as the AIKIDO tool was overwhelmingly efficient at predicting the correct class of each data point.

5.7.1 Conclusions - comparison with AIKIDO

- The AIKIDO tool was capable of predicting the classes in the data set 99% of the time, therefore the comparison with the MiniRocket RFC was almost identical to a comparison to the ground truth of the test data set.
- Future experiments could be done on data points whose labels could not be manually translated according to the BUDO schema to see how successful the two approaches are at finding the same category tag (class).

6 Summary, discussion and conclusions

Throughout this thesis, approaches for increasing the performance of supervised time series classification algorithms were discussed and tested on the practical use case of assigning BUDO schema category tags to data points of the E.ON ERC building.

The summary and discussion section explains the methods used for preparing the data, conducting the tests and details the incremental changes in performance throughout the trials. The conclusion section details the overall performance, best results and opportunities for improvement.

6.1 Summary and discussion

During the course of this thesis two state-of-the-art supervised time series classification algorithms (ROCKET and MiniRocket) were tested and compared on a data set obtained from the E.ON ERC building. The overall goal was to be able to assign the appropriate BUDO schema category tag to any data point, using time series data of the data point instead of its label.

First task, discussed in chapter 4, was acquiring the data and the pre-definition of classes for the classifier algorithm using the the BUDO schema category tags. Thirteen different categories of the BUDO schema were present among the data points of the base data set. From these categories, the "signal type" category was selected, as its tags differentiate between different time series data types, which make up the entirety of data used in the thesis. This category type was represented in 62% of all data points in the data set according to the manual translation (shown in figure 4.1).

For the main data set, which was used later in the trials, ten pre-defined classes were selected from the category tags of the signal type category to represent a wide range of different types of data points. The final results show, that this approach of selecting the pre-defined classes is adequate for supervised time series classification.

After the definition of the classes, the issue of data quality and pre-processing was tackled in section 4.4 through section 4.5. In the main data set, which covered 2010 data points and a year's worth of data between 01.01.2020 - 01.01.2021, overall 72% of all time stamps of the data points contained empty (or "NaN") values. This, as was later shown in trial 2a, made the unprocessed data inadequate for using as input for supervised time series classification.

All trials with empty values either resulted in immediate termination due to an error, or the trial completed but resulted in all data points predicted as one class (trial 2a, figure 5.2). This highlighted the need for data pre-processing.

In chapter 5, a set of five trials were conducted with a focus on testing two state-of-the-art supervised time series classification methods, the ROCKET and the MiniRocket, coupled with three linear classifiers, the random forest classifier (RFC), the ridge classifier and the stochastic gradient descent (SGD) classifier.

Trial 1 confirmed the superiority of the MiniRocket and the ROCKET algorithms over conventional classifier algorithms in terms of computational speed. The MiniRocket algorithm finished between 40 to 551 times faster on the trial 1 data set compared to the other classifiers.

In trial 2, the issue of empty time stamp values were tackled. Two data filling methods, forward filling and backward filling, were investigated. When comparing trial 2b (backward filling, table 5.7) and trial 2c (forward filling, table 5.9), the results of trial 2c were on average 0.4% point higher, but the confusion matrices (figure 5.3 and 5.4) revealed that using forward filling caused two previously distinct classes (2 and 4) to blend into a third class (1), therefore backward filling was chosen to be used for the remaining trials. The best overall result with backward filling was reached by the MiniRocket RFC with an average of 77.45%

In trial 3, the influence of over-representation of a class in the training data was investigated. Class 0 made up close to 30% of all data points in the main data set (figure 4.3). In trial 3a, the overall number of data points of each class in the training data set were reduced until they were all equally represented. In trial 3b the number of data points of class 0 were reduced by half, but all other classes remained the unchanged. Overall the second approach in trial 3b proved to be more beneficial, resulting in an overall increase in performance compared to trial 2b by 1.4% points. Given enough training data, it is likely that the approach used in trial 3a would have been more beneficial, but the reduction of the overall number of data points for training negatively effected the result.

In trial 4, the possibility of merging similar classes was explored. Although the merging of two similar classes (5 and 6) resulted in a marginal increase in overall performance of 0.7% point compared to trial 2b, correct classification of the newly created class (6) did not increase compared to when the two classes were handled separately (figure 5.8). The results of previous trials suggested, that merging other classes such as 1 and 2 could have resulted in better performance, however, these classes do not represent similar data points from a human perspective and merging them would offer little practical use, therefore it was not investigated.

In trial 5, the influence of increased sampling rate was investigated. With the reduced computational requirements of the ROCKET and MiniRocket algorithm, supervised time

series classification tasks of previously not feasible time spans and sampling rates can be tested on simple desktop computers. In trial 5a the time span of the main data set was changed from one hour to 15 minutes, quadrupling the amount of time stamps of each data point in the data set. This resulted in an overall better performance of 1.24% points. In trial 5b, the final trial, all changes on the training data that have previously been proven to be beneficial have been implemented, as well as the class with the largest amount of false classifications (class 1) have been removed. This resulted in an overall average performance of 83.58% using the MiniRocket RFC, which is a 6.13% point increase compared to the first successful classification trial in trial 2b.

Comparison with the AIKIDO tool showed, that using a data set, that has previously been manually translated according to the BUDO schema made the predictions of the AIKIDO tool 99.84% accurate (in case of 600 training labels). Therefore the comparison with the supervised time series classification was almost identical to the ground truth in the test data of trial 2b.

6.2 Conclusion

Supervised time series classification algorithms used on the time series data of BACS data points and the BUDO signal type category tags as classes was successful on the tested data sets. The MiniRocket RFC classifier reached the overall best performance in each trial and dominated all other tested classifiers in almost every single metric each time. The overall best performance was achieved in trial 5b with the average metrics of 83.58%. Data quality issues and the relatively low number of data points belonging to certain classes negatively effect the outcome of any machine learning method including supervised time series classification. Tested methods for modifying the data collection and improving the input data, such as backward filling, alleviating over-representation, and increasing sampling rate were shown to positively effect the overall result.

When compared to similar studies examining supervised time series classification of BACS data points (discussed in chapter 3), the overall results show a definitive improvement. The MiniRocket algorithm allows for testing of much larger time spans, potentially multiple years, with higher sampling rates than previously possible. This allows the classifiers to capture nuances in data point behavior, that a one-week or one-day sample simply cannot. Judging from the results, it is highly probable, that the supervised time series classification approach for classifying data points according to the BUDO schema could supplement the language-based approach, especially in case of data points where the data point label does not include any indication of its signal type category. To confirm this theory, further test on different data sets are required.

7 Future work

This study has shown, that using supervised time series classification to classify data points of BACS according to the BUDO schema category tags is possible with relatively high accuracy. There are certain aspects, however, where the investigations of this thesis could be expanded to achieve even better results:

- The data set used in this study only included data points from only one building, as a result the number of data points available for each class was limited. With enough data points, over-representation bias could be eliminated while not suffering from having too few data points in the training data from any given class.
- Further increase in sampling rate could enhance the results.
- Only ten category tags (classes) of the signal type BUDO category were tested, these could be expanded to all category tags and eventually to other categories as well.
- In addition to backward filing and forward filling, interpolation or other pre-processing methods could be tested.
- Overall better input data quality could enhance the results.

In practical use cases, it could be beneficial to concentrate on data points of a certain system of a building (such as air handling, heating, etc.) instead of the entire building as a whole when applying supervised time series classification. Data points of these systems are usually handled by the same programmable logic controller, therefore they are easy to separate from the remaining data points of different systems. Results of such classification could indicate more information about a data point than its signal type.

Bibliography

Alhanoof Althnian, Duaa AlSaeed, Heyam Al-Baity, Amani Samha, Alanoud Bin Dris, Najla Alzakari, Afnan Abou Elwafa, and Heba Kurdi. Impact of dataset size on classification performance: An empirical evaluation in the medical domain. *Applied Sciences*, 11(2):796, 2021.

A Amidon. A brief survey of time series classification algorithms, 2020.

Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery*, 31(3):606–660, 2017.

Bharathan Balaji, Chetan Verma, Balakrishnan Narayanaswamy, and Yuvraj Agarwal. Zodiac: Organizing large deployment of sensors to create reusable applications for buildings. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pages 13–22, 2015.

BMU. Climate action plan 2050. https://unfccc.int/process-and-meetings/the-paris-agreement/the-paris-agreement, 2016.

Gerrit Bode, Thomas Schreiber, Marc Baranski, and Dirk Müller. A time series clustering approach for building automation and control systems. *Applied energy*, 238:1337–1345, 2019.

Jason Brownlee. Tactics to combat imbalanced classes in your machine learning dataset. *Machine Learning Mastery*, 19, 2015.

CA. The layers of modern building automation system architecture. https://control.com/technical-articles/the-layers-of-modern-building-automation-system-architecture/, 2021.

Duncan S. Callaway. Can smaller loads be profitably engaged in power system services? In *2011 IEEE Power and Energy Society General Meeting*, pages 1–3, 2011. doi: 10.1109/PES.2011.6039890.

Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21 (1):1–13, 2020.

Eugenio Culurciello. Neural network architectures. https://towardsdatascience.com/neural-network-architectures-156e5bad51ba, 2017.

Angus Dempster, François Petitjean, and Geoffrey I Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.

Angus Dempster, Daniel F Schmidt, and Geoffrey I Webb. Minirocket: A very fast (almost) deterministic transform for time series classification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 248–257, 2021.

Li Deng and Xiao Li. Machine learning paradigms for speech recognition: An overview. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(5):1060–1089, 2013.

Keith D. Foote. A brief history of machine learning. https://www.dataversity.net/a-brief-history-of-machine-learning/, 2019.

Johannes Fütterer, Maksymilian Kochanski, and Dirk Müller. Application of selected supervised learning methods for time series classification in building automation and control systems. *Energy Procedia*, 122:943–948, 2017.

Jingkun Gao, Joern Ploennigs, and Mario Berges. A data-driven meta-data inference framework for building automation systems. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pages 23–32, 2015.

Ralf Gitzel. Data quality in time series data: An experience report. In *CBI (Industrial Track)*, pages 41–49, 2016.

GL. Types of neural networks and definition of neural network. https://www.mygreatlearning.com/blog/types-of-neural-networks/, 2020.

Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for multi-class classification: an overview. *arXiv preprint arXiv:2008.05756*, 2020.

Dezhi Hong, Hongning Wang, Jorge Ortiz, and Kamin Whitehouse. The building adapter: Towards quickly applying building analytics at scale. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pages 123–132, 2015.

Michal Hrabia. Deep learning vs. machine learning. https://towardsdatascience.com/deep-learning-vs-machine-learning-e0a9cb2f288, 2020.

IBM. Neural networks. https://www.ibm.com/cloud/learn/neural-networks, 2020.

Michelle Knight. What is machine learning? https://www.dataversity.net/what-is-machine-learning/, 2017.

Kai Krieger. Optimizing a lstm-convolutional model for categorizing data point labels from bacs using a uniform meta data scheme. Bachelor thesis, RWTH-Aachen, 2021.

Y Park. Point naming standards: a necessary evil for building information integration. *ISA Automation Week*, pages 33–34, 2012.

Valerie Sessions and Marco Valtorta. The effects of data quality on machine learning algorithms. *ICIQ*, 6:485–498, 2006.

Boaz Shmueli. Matthews correlation coefficient is the best classification metric you've never heard of. https://towardsdatascience.com/the-best-classification-metric-youve-never-heard-of-the-matthews\-correlation-coefficient-3bf50a2f3e9a, 2019.

Florian Stinner. Budo schema: Standardized designation of time-resolved building data. https://www.ebc.eonerc.rwth-aachen.de/cms/E-ON-ERC-EBC/Forschung/OPEN-SOURCE/~qajk/Standardisierte-Bezeichnung-zeitaufgeloe/?lidx=1, 2020.

FLORIAN Stinner, ALINA Kornas, MARC Baranski, and DIRK Müller. Structuring building monitoring and automation system data. *The REHVA European HVAC Journal-August*, 2018:10–15, 2018.

Florian Stinner, Paul Neißer-Deiters, Marc Baranski, and Dirk Müller. Aikido: Structuring data point identifiers of technical building equipment by machine learning. In *Journal of Physics: Conference Series*, volume 1343, page 012039. IOP Publishing, 2019.

UN. Paris agreement. https://unfccc.int/process-and-meetings/the-paris-agreement/the-paris-agreement, 2015.

Weimin Wang, Michael R Brambley, Woohyun Kim, Sriram Somasundaram, and Andrew J Stevens. Automated point mapping for building control systems: Recent advances and future research needs. *Automation in Construction*, 85:107–123, 2018.

Yagang Zhang. *Application of machine learning*. BoD–Books on Demand, 2010.

Appendix

Table .1: Python packages used in the thesis

Package	Version
ebcdata	0.0.1
EbcSupervisedLearning	0.1.0.4
keras	2.6.0
matplotlib	3.3.1
numpy	1.19.5
openpyxl	3.0.6
pandas	1.1.1
scikit-learn	0.23.2
scipy	1.5.4
seaborn	0.11.2
sktime	0.5.3
tensorflow	2.6.0
tsfresh	0.18.0
xlrd	2.0.1
xlwt	1.3.0
Python	3.6.12

A list of all installed python packages are included in the digital documentation in the file "requirements_env.txt".

Declaration of Originality

I hereby declare that this thesis and the work reported herein was composed by and originated entirely from me. Information derived from the published and unpublished work of others has been acknowledged in the text and references are given in the list of sources. This thesis has not been submitted as exam work in neither the same nor a similar form. I agree that this thesis may be stored in the institutes library and database. This work may also be copied for internal use.

Aachen, Monday 6th December, 2021

Balázs Tóth