TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Department of Software Science

Vjatšeslav Tšetšnev 164480 IAPB

SHORT-TERM LOAD FORECASTING USING ARTIFICIAL NEURAL NETWORK

Bachelor's thesis

Supervisor: Eduard Petlenkov
PhD

Margarita Spitšakova
PhD

Tallinn 2019

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Vjatšeslav Tšetšnev 164480 IAPB

ELEKTRIENERGIA TARBIMISE LÜHIAJALINE ENNUSTAMINE TEHISNÄRVIVÕRKUDE ABIL

Bakalaureusetöö

Juhendaja: Eduard Petlenkov
PhD

Margarita Spitšakova
PhD

Tallinn 2019

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Vjatšeslav Tšetšnev

21.05.2019

Abstract

The purpose of the thesis is to investigate how precisely the short-term consumption of electricity of Alexela Energy AS clients can be forecasted using Artificial Neural Network and implement a User Interface for this application. High level of forecast precision allows to better plan the orders of electricity from the electricity producers and brings a direct economic benefit. Additionally, an overview is made of the State-of-the-Art methods, which scientists use for solving similar load forecasting problems.

Major issues addressed in the thesis include conduction of the practice tests for studying the behavior of Artificial Neural Network, hyperparameter optimization for finding the most beneficial structure of the network and suitable settings, feature selection and data rescaling on various offline and up-to-date online data. The generalization test is conducted to allow a more in-depth analysis of the forecasting capability of the models.

As a result, the average forecast error lowered to 3.62% from initial ca. 4.50%. To allow convenient use of the model the executable forecasting application was created in Python with a User Interface with data visualization.

This thesis is written in English and is 58 pages long, including 6 chapters, 21 figures and 19 tables.

Annotatsioon

Elektrienergia tarbimise lühiajaline ennustamine tehisnärvivõrkude abil

Lõputöö eesmärk seisneb selles, et selgitada välja kuivõrd täpselt saab ennustada elektrienergia lühiajalist tarbimist Alexela Energia AS klientide poolt, kui kasutada selleks tehisnärvivõrku ja realiseerida selle rakenduse jaoks kasutajaliides. Prognoosimistäpsuse kõrge tase võimaldab planeerida optimaalsemalt tellimusi elektrienergia saamiseks elektritootjatelt ja annab otsest majanduslikku kasu. Lisaks sellele on antud ülevaade eesrindlikest meetoditest, mida teadlased kasutavad analoogiliste ülesannete lahendamiseks elektrienergia tarbimise prognoosimisel.

Peamised probleemid, mida lõputöös käsitletakse, sisaldavad endas praktiliste testide läbiviimist tehisnärvivõrgu käitumise tundmaõppimiseks ja mudeli parameetrite optimeerimist kõige soodsama võrgustruktuuri ning selle jaoks sobivate seadete leidmiseks. Samuti on koos automatiseerimismeetodite kasutamisega viidud läbi sisendatribuutide valik ja käsitletud küsimust, mis on seotud erinevate lokaalselt salvestatud ja aktuaalsete online-andmete töötlemise, esitamise ja skaleerimisega. Käsitleti ajaloolisi andmeid ja ilmaprognoose, mis sisaldasid andmeid õhutemperaturi, õhurõhu ja niiskusesisalduse kohta, elektrienergia reaalset tarbimist Eestis, valguspäeva pikkust, elektrienergia tootmist päikesepaneelidega, andmeid riigipühade ja neile eelnevate päevade kohta, genereeritud andmeid, mis puudutasid kellaaega, päeva, nädalat ja kuud. Täiendavalt on läbi viidud üldistav test mudelite prognoosimisvõimaluste sügavamaks analüüsiks.

Selle tulemusena väheneş prognoosi keskmine viga kuni 3.62% esialgse 4.50% võrreldes. Mudeli mugavamaks kasutamiseks loodi Pythonis eraldiseisev rakendus graafilise kasutajaliidesega, milles kasutatakse andmete visualiseerimist.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 58 leheküljel, 6 peatükki, 21 joonist, 19 tabelit.

List of abbreviations and terms

AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application programming interface
ARIMA	Autoregressive integrated moving average
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
CV	Cross-Validation
DES	Double Exponential Smoothing
DNN	Deep Neural Network
GUI	Graphical User Interface
JS	JavaScript
LSSVM	Least-squares Support Vector Machine
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
ML	Machine Learning
MSE	Mean Squared Error
MVC	Model-View-Controller
NaN	Not a Number
ReLU	Rectified linear unit
SARIMA	Seasonal Autoregressive Integrated Moving Average
SPA	Single-Page Application
STLF	Short-term electricity load forecasting
SVC	Support Vector Classification
SVM	Support Vector Machine
SVR	Support Vector Regression
tanh	Hyperbolic tangent function
UI	User Interface

Table of contents

List of figures

List of tables

1 Introduction

1.1 Background

Clients in Estonia can freely select their electricity supplier since the beginning of 2013, the year, when an electricity market became open [1]. One of such electricity suppliers is Alexela Energy AS who provide both private and corporate customers with electrical energy. Electricity suppliers don’t produce electricity; instead, they order it from the electricity producers on a wholesale market, e.g., Nord Pool [2].

1.2 Problem

The main problem with the electricity is that it’s a specific product, which cannot be stored in the amount of electricity supplier like Alexela Energy deals with [3]. Therefore, instead of buying electricity in advance and preserving it, electricity suppliers order the electricity for the next day on a wholesale market, which needs to be generated by electricity producers [2]. The produced energy will be then directly delivered to the end customer.

Ordering the electrical energy one day in advance is also cheaper for electricity suppliers than buying it on short notice. Consequently, they need to estimate their clients’ consumption tomorrow and order the exact amount of electricity required per hour. If they order too much electrical energy, the excesses will be lost. If they order too little, they will have to order additionally needed electricity through other more-expensive channels. In both cases, the poorly planned purchase means a direct monetary loss for the electricity supplier.

The amount of financial loss can be roughly estimated. As an example, I’ll take the amount of electricity, which was totally consumed in Estonia in the year 2018 – approximately 8,4 TWh. The average next day price in 2018 on Nord Pool was 48,43 €/MWh. The cost for 1% of the total consumed electrical energy in Estonia in 2018 can be calculated by multiplying those two numbers in equivalent units and dividing by 100; it is more than €4 million per year.

This estimation clearly shows that only a slight difference of 1% can have such a massive financial impact. Therefore, the forecast precision for the ordering the electricity a day ahead per hour for electricity suppliers like Alexela Energy should be predicted as exact as possible.

1.3 A research project at TalTech

Since the end of 2018, M. Spitšakova, J. Belikov, and E. Petlenkov were engaged in research at TalTech on the Short-term electricity load forecasting (STLF) topic [4]. STLF is the exact name widely used in the research papers for forecasting the electricity consumption for a period of from one to seven days [5]. The goal of this study was to predict a day-ahead electricity consumption for private and business customers of Alexela Energy.

The results of the several month's works included a detailed statistical and correlational analysis of available datasets [4]. The best prediction accuracy was achieved by a combination of both simple and multiple linear regression approaches and the generation of separate models for each day and hour [4].

As will be described in part 2, there exist a wide range of other alternative approaches for prediction of electricity consumption, e.g., a large group of Artificial Intelligence (AI) methods. They were also examined for solving STLF problem, but only preliminary research was conducted on Support Vector Machine (SVM) and Artificial Neural Network (ANN), including the initial models in MATLAB [4].

1.4 Purpose

In contrast to the linear regression model, both AI methods like SVM and ANN are considered to be able to provide better results because of their ability to solve complex non-linear problems [5], [6], one of which is an STLF problem. At the same time, SVM and linear regression used a limited amount of information only from the recent months [4] and therefore, the ability to learn from large amounts of historical data was missing.

The research group made a hypothesis that an ANN is capable of providing better results because of the ability to learn on always growing amounts of data. My assignment was to check this hypothesis. I used the knowledge from a statistical and correlational analysis which had been carried out during the research project and continued the work on an ANN approach for STLF applied to Alexela Energy company with the main focus on further improving the prediction precision.

From the initial model in MATLAB, I used the input vector and structure of ANN, which formed the initial settings for my further work and implementation. Furthermore, the initial model in MATLAB was benchmarked, and I've used its performance as a reference. The Mean Absolute Percentage Error (MAPE) of the model in MATLAB was ca. 4.50%.

The primary objectives for my thesis included:

- Investigate how precisely the electricity consumption can be forecasted using ANN.
- Find the combination of the input vector, hyperparameters, and the structure of ANN that will provide the most precise forecast.
- Implement the model using ANN in Python. This programming language was chosen because of available Machine Learning (ML) libraries like TensorFlow and Keras. Both of them are extensively supported, regularly updated, free, and open-source. In the case of MATLAB, it is proprietary software, and a license should be purchased. The second reason is the greater possibilities for creating a standalone application.

Additional objectives for my thesis included:

- Investigate alternative approaches and techniques and make an overview of State-of-the-Art methods.
- Implement a Graphical User Interface (GUI) for the forecasting application.

1.5 Overview of the thesis

In the beginning, I give an overview of the State-of-the-Art methods scientists use in research papers for solving similar problems. Particular focus I make on the ANN because this is the method I investigate in my thesis.

After that, I give an overview of the methodology, tools, and processes I used.

Then I present the results of my work, from practice tests, hyperparameter optimization, data rescaling, feature selection, generalization test to the creation of the GUI for a standalone application.

In the end, I analyze and discuss the achieved results, outline further development possibilities, and give a summary of the key findings of my work.

2 State of the Art methods

In this part, I present an overview of the State-of-the-Art techniques, which are used in the research papers and journal articles for STLF problems and give a more detailed overview of the concept and principles of an ANN.

There exist another similar to STLF problem – the electricity price forecasting. The methods used for the later are close to ones used for STLF. The primary difference lies in the factors that play a role in forming the predictions, which considerably differs. E.g., the macro-economic factors are critical for predicting electricity prices.

Three main groups of methods are distinguished for solving STLF problems: time series methods, which includes statistical and conventional methods, AI methods, the primary representatives of which are ANN and SVM and hybrid or a combinational approach [6].

2.1 Time Series

Time series group is the oldest one for STLF and consist of statistical and conventional methods.

2.1.1 Linear regression

Linear regression is a statistical method, which describes the dependency between at least two variables: the independent variable or predictor, which is the input, and the dependent variable or prediction, which is the output of the linear regression [7]. The number of independent variables can vary. The linear regression is called simple or univariate if it receives as input only one such independent variable [7], and multiple, if there are two or more input variables, also called features [8].

A linear model does not automatically imply that as a result, we will get a linear function. The term linear in a linear model is used in terms of parameters and not independent variables [9], meaning the function can be non-linear.

Mean Squared Error (MSE) is often used as a cost function. The main idea of linear regression is to minimize this function by changing the parameters to provide the best possible estimate for a dataset [8].

Linear regression is considered as one of the oldest approaches [5] for STLF problems. Multiple linear regression was first applied in November 1966 for finding a relationship between the weather and peak demands of electricity [5], [10].

2.1.2 ARIMA family

Autoregressive integrated moving average (ARIMA) model was proposed in 1970 by Box and Jenkins [11]. This is why it's also often referred to as the Box-Jenkins model [11]. It allows applying regression techniques on the non-stationary data [12]. The data is called non-stationary if it contains some trend [12].

ARIMA model can be described using the following attributes: p, d, and q [11]. P stands for Autoregressive (AR), d for Integrated (I) and q for Moving average (MA) parts [11]. Typically, all these three parameters are put together [11], [12], as shown in Equation 2.1.

$$ARIMA\ (p, d, q) \tag{2.1}$$

The autoregressive part represents a combination of previous values, which take effect on the calculation of a prediction [12]. They are also called lagged values [11]. For example, if the lag is equal to one, we will consider only one last value for making a forecast. Values are also weighed, meaning we can use a proper weight for each of previous values, thereby making some of them more or less relevant [12].

The integrated or differencing part [13] is related to the fact that ARIMA is applied to non-stationary data [12]. Therefore differences between values should be calculated instead of operating with absolute values [13]. As a result, we get the non-stationary data converted to the stationary [13].

The moving average part represents a combination of previous errors we should consider for the calculation of a prediction [12].

There exist several extensions to the ARIMA model. One of them is the Seasonal Autoregressive Integrated Moving Average (SARIMA), which works directly with seasonal data, which forms cycles over time [14]. The notation for SARIMA can be written, as shown in Equation 2.2.

$$ARIMA(p,d,q)(P,D,Q) \tag{2.2}$$

where p, d and q represent the non-seasonal part, just like in the case of ARIMA, and P, D, and Q represent the new seasonal part [11], [14].

Often in the journal articles, an "X" modification of ARIMA-family models can be met. For example, SARIMAX was used in [15]. It additionally to seasonal ARIMA accepts exogenous factors as one of its input parameters [14], e.g., changing weather [3].

2.1.3 Exponential smoothing

While making the predictions, Exponential smoothing method takes directly into consideration the previous values [16]. It assigns a higher weight to the newer values and lower weight to the older values by making the recent values more significant and the older values less essential while forecasting the next value [16]. In contrast to the previously discussed ARIMA model, Exponential smoothing does not use an autoregressive component, which makes it easier to calculate [11].

Three main types of Exponential Smoothing can be outlined [16]:

- Simple Exponential Smoothing
- Double Exponential Smoothing (DES)
- Triple Exponential Smoothing (Holt-Winters)

Simple Exponential Smoothing can be calculated, as shown in Equations 2.3-2.4.

$$F(0) = A(0) \tag{2.3}$$

$$F(t) = F(t-1) + \alpha * (A(t-1) - F(t-1)) \tag{2.4}$$

In Equation 2.3 we assign the first actual value as an initial forecasted value. In Equation 2.4 F(t) is the calculated forecast and A(t) is an actual value at time t. α is the smoothing factor or parameter, where 0 < α < 1 [16].

DES offers support for trends [16] similarly to the ARIMA model, and Triple Exponential Smoothing (Holt-Winters) supports besides trends seasonality [16], just like previously discussed SARIMA.

2.1.4 Kalman filter

Kalman filter is a statistical technique for value estimation. In [17] a moving window technique was applied in conjunction with a Kalman filter for solving an STLF problem, which was used to recursively calculate the forecasts of electricity demand based on the prior consumption data and weather information.

2.2 Artificial Intelligence

An AI group is overall considered as an improvement over the previously discussed Time Series group mainly due to their strength in working with non-linearities [5], which is a necessity for STLF.

2.2.1 Machine Learning

There exist two main types of ML: supervised and unsupervised [18]. The difference consists in the way, how the learning process will be organized. In the supervised case, we always have for each input a right output. In the unsupervised case, the ML algorithm will deal only with the input information without knowing the correct output.

The problems where supervised ML algorithm is applicable can be divided into two groups [19]:

- Classification task's goal is to assign the right class or categorize items. Accuracy is the standard metrics for such problems [19], how accurately was the prediction of a class.

- Regression problems don't have a categorical output, as the classifications problems. Instead, their output scale is continuous [19]. Standard metrics include Mean Absolute Error (MAE) and MSE.

2.2.2 Artificial Neural Network

The structure of the human brain was the inspirational impulse for the development of ANN in the 1950-s [18]. The primary assignment of both biological as well as an artificial neuron is to receive the signal and transmit it further, as shown in Figure 1 and Figure 2 accordingly.

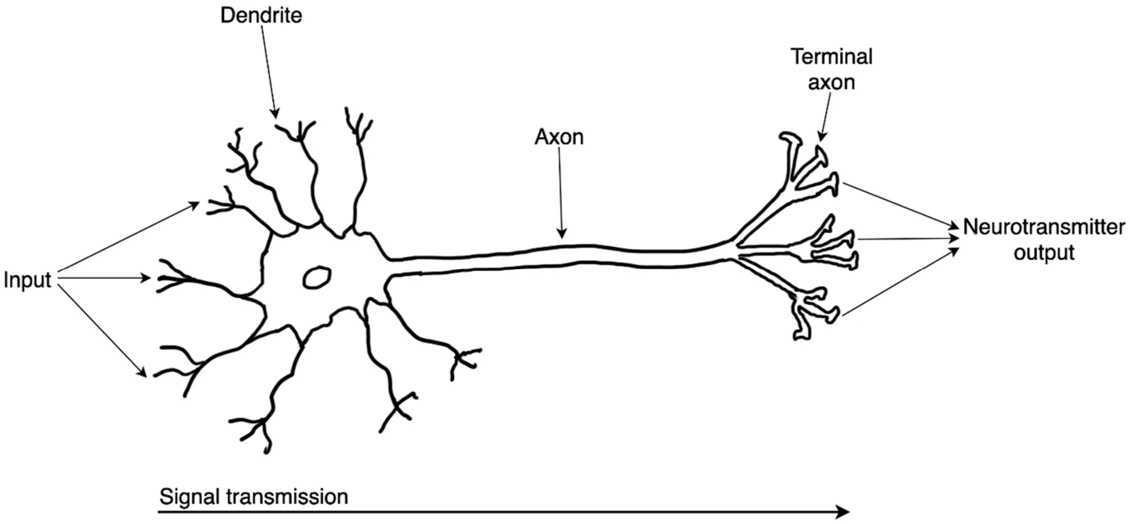


Figure 1. Structure of a biological neuron was the inspiration for the development of an artificial neuron.

ANN consists of neurons. The main workflow of a neuron can be divided into the following steps, as shown in Figure 2. A neuron receives the inputs x_1 and x_2, each of which has a specific weight w_1 and w_2, how valuable it is. Neuron calculates a weighted sum of those inputs and adds a bias *b*, which is specific for each neuron. After this, the activation function is applied to this result. Finally, the output of this activation function is the final output of a neuron.

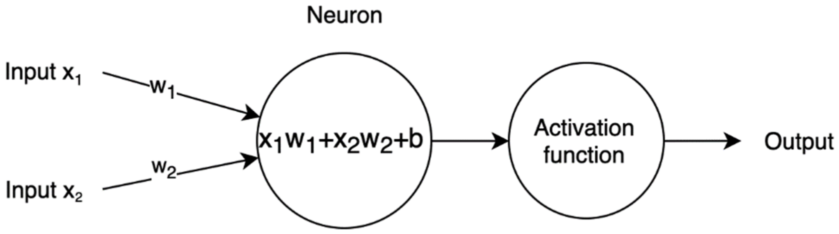
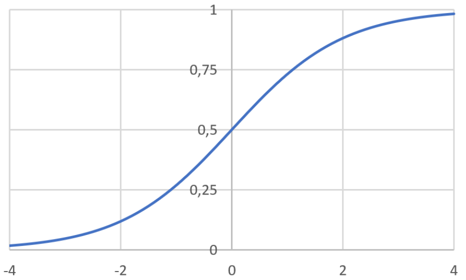
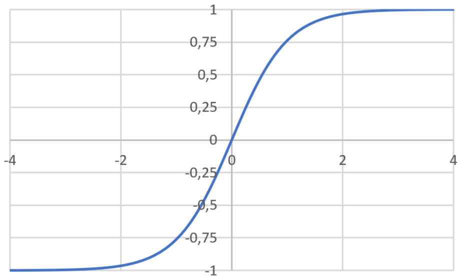
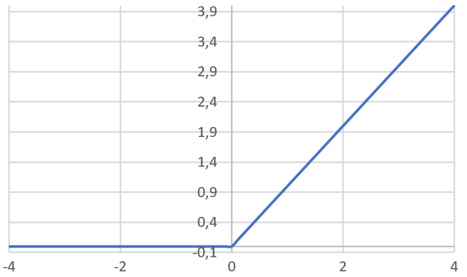


Figure 2. The workflow of an artificial neuron includes calculation of a weighted sum of inputs with the addition of bias and application of an activation function.

The activation functions are vital parts of an ANN because they allow solving non-linear problems [20]. Some of the frequently used activation functions are shown in Table 1. If there were no activation functions in the neuron's workflow, an ANN would perform very similarly to a linear regression model [20].

Table 1. Some of the frequently used activation functions for ANN.

Activation function	**Equation**	**Graph**
Sigmoid (logistic) function	$y = \dfrac{1}{1 + e^{-x}}$	
Hyperbolic tangent function (tanh)	$y = \tanh(x) = \\ = \dfrac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$	
Rectified linear unit (ReLU)	$y = \max(0, \mathrm{x})$	

Both sigmoid and tanh functions have a similar downside: they suffer from the vanishing gradient problem [21]. On the sides, they both have regions, which are almost entirely flat and therefore cause a zero-gradient issue. ReLU solves this problem for x > 0 [21] and is today widely utilized for Deep Learning purposes.

Neurons in an ANN can be divided into the following layers, as shown in Figure 3: input, hidden, and output layers. The number of layers in an ANN is a sum of hidden layers and an output layer [22]. The input layer is only an input vector. Similarly, the number of neurons can be calculated by omitting the input layer; only neurons of the hidden and output layers are counted [22], marked in Figure 3 as grey.

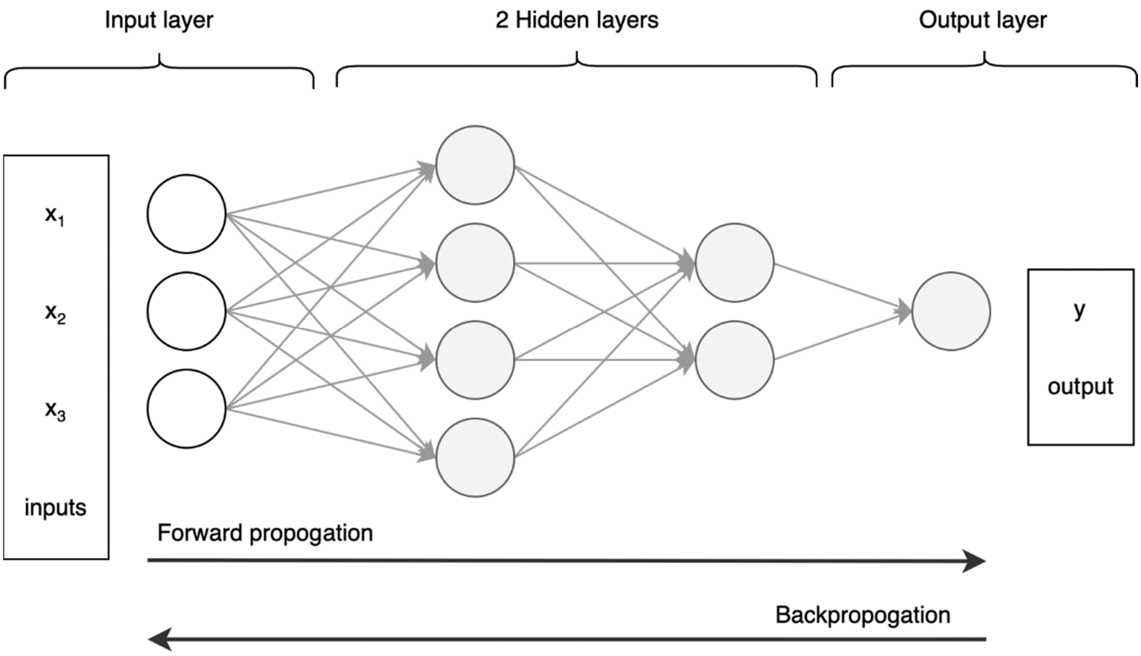


Figure 3. ANN consists of a layered structure and works in cyclical phases called forward and backpropagation.

The work of an ANN can be described in cycles, consisting of two phases: forward propagation and backpropagation. The direction of the forward propagation phase is from left to right, as shown in Figure 3, and the goal is to produce an output for specific input.

When the final output of an ANN has been calculated, in case of supervised learning, it’s essential to calculate the error, how wrong the result is in comparison to the right answer. This is the assignment for a cost or loss function. In the case of a regression problem, one of the possible metrics is an MSE. Optimization algorithm, e.g., gradient descent, helps, in this case, to minimize this loss function by leading it in the right direction to the function minimum [23].

The actual training of an ANN consists of tuning the parameters: weights and biases. In the beginning, an ANN generates those parameters randomly. After the cost function is calculated, the parameters of a network will be adjusted during the backpropagation phase accordingly to an optimization algorithm [23]. Typically, the training instances are processed in batches, which makes the training procedure faster, because the gradient descent calculation and update of the network’s parameters take time. The direction of the backpropagation phase is from right to left, as shown in Figure 3.

One of the primary characteristics of ANN is the capability to generalization [18]. It means ANN should not remember the data it was trained with, but it should understand, extract and learn the patterns, trends, and dependencies so that it would be able to work further on with the new previously unseen data adequately.

Besides shallow-structured feedforward ANN models, other Deep Neural Network (DNN) structures are being utilized for STLF problems. The minimum amount of hidden layers so that the ANN would be called deep is two [18]. DNN subtypes include Recurrent Neural Network (RNN), which better works with pattern information due to the existence of cycles [18].

Two other network types are based on RNN which are used for STLF. The first one is Long Short-Term Memory (LSTM), which was used in [24] for predicting electricity consumption in France. The benefit of LSTM lies in utilizing lagged or previous values [24]. The input for LSTM-RNN should be, therefore, 3-dimensional [25]. The second one is the Echo State Network, which was used in [26]. It's a method which utilized reservoir computing, the primary goal of which was that the training procedure became quicker and less-complex in contrast to RNN [26].

2.2.3 Support Vector Machine

The best way to describe the principle of SVM is to study a classification problem, where we have items which should be divided into two classes [27]. The core idea consists in the case of a two-dimensional data in drawing a line, which would accurately separate those two sets – a hyperplane [27]. The elements of the datasets, which are located the closest to the hyperplane, are called the support vectors [27]. The minimum length of the line from them to the hyperplane, which is perpendicular is called a margin [27]. SVM is an optimization task, meaning we have to find the combination of those margins, where they are both the biggest [27]. This way, we will get the optimal position for the hyperplane and will increase the robustness of the method.

Sometimes the data cannot be split up into two categories with a straight line. For such non-linear cases, SVM offers a kernel trick [28] – a transformation to the higher-dimensional representation of the data, where it becomes separable [27]. It's accomplished using functions, also called kernels [28], responsible for non-linear mappings [27].

Additionally, SVM can be used for the data, which contains outliners [28] – the concept of a soft margin [29].

The SVM can solve not only the classification problems but suits also for regression problems. Therefore it can be divided into Support Vector Classification (SVC) and Support Vector Regression (SVR) subtypes [29]. Both SVC and SVR are powerful tools for solving complex non-linear assignments.

A Least Squares reformulation to the original SVM (LSSVM) which was introduced in 1999 by Suykens and Vandewalle [30] was used in [5] for solving an STLF problem. In contrast to SVM, which is a quadratic programming problem, in LSSVM, it's necessary to find a solution to a linear system of equations [30].

2.3 Hybrid

Hybrid means it consists of several models. The problem with individual models is that they often have method-specific limitations. By combining several models, it's possible to minimize the prediction error even further [6].

There exist two types of hybrid models, as outlined in [6]. The first type includes the models, each of which produces its forecast for electricity consumption independently [6]. The second type requires a prior decomposition of the electricity demand set into some separate portions, each of which is further processed by an independent model [6]. Afterward, their outputs are combined [6]. This decomposition can be accomplished by utilizing wavelet transformation [6].

As an example for a hybrid approach, in [31] for STLF was used a combination of Fuzzy Time Series, another AI forecasting method, and Convolutional Neural Network (CNN), a subtype of DNN, which was developed for processing images [18].

2.4 State of the Art methods summary

Essential approaches for STLF problem are shown as a summary graph in Figure 4.

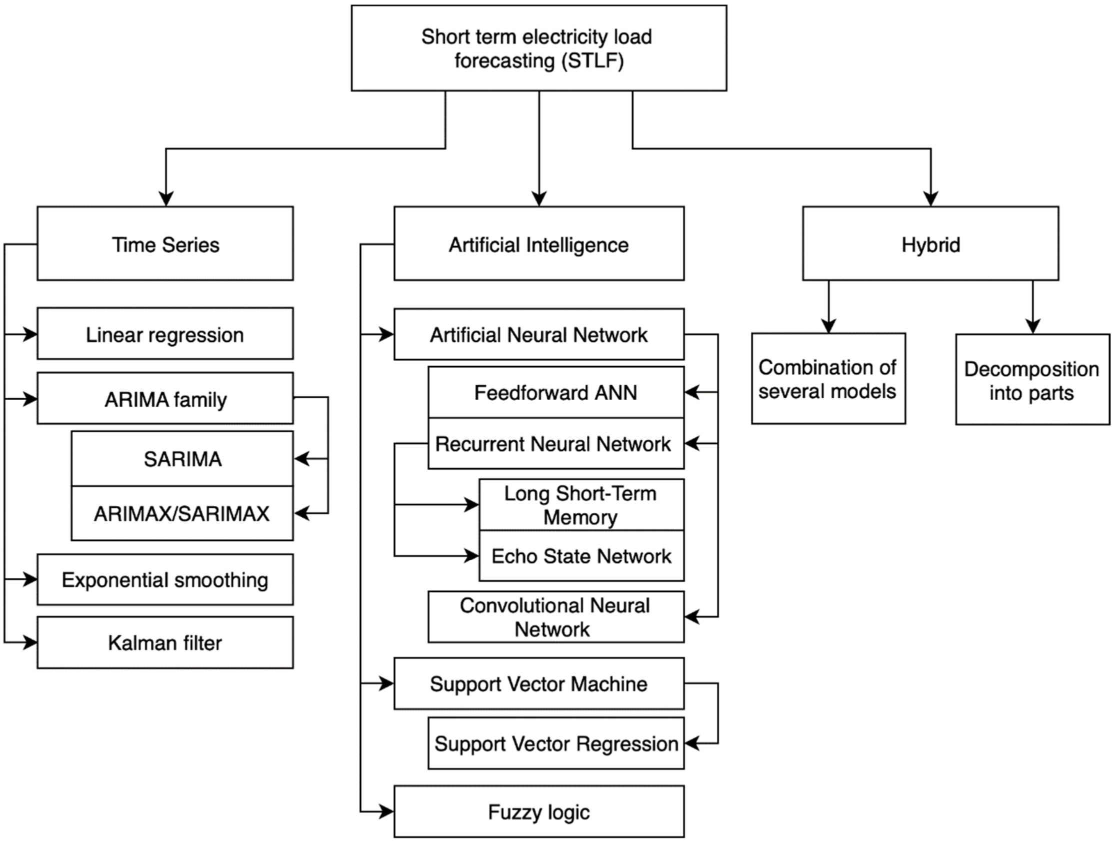


Figure 4. Mostly used State of the Art methods for STLF.

2.5 Conclusions

After conducting my literature research on STLF, I made the following conclusions:

- There exist no gold-standard for solving STLF problems. At the same time, the number of research works on this topic, and different methods, techniques, approaches is tremendous.

- The trajectory of development in this field is apparent. AI methods and hybrid approaches are today often researched due to their vast potential. The first suits the best for non-linear data, whereas the second helps to resolve the weaknesses associated with the individual models [6].

- Each solution is unique, and none of them are universally applicable. Often solutions depend on specific requirements and available datasets.

- Solutions are often not reproducible. Usually, no code or datasets are provided because of the sensitive nature of the information. Frequently key-parts of the development or critical parameters are missing or incomplete.

3 Method

In this part, I give an overview of available datasets, tools I used for the implementation of a forecasting application and discuss the process, including different metrics and sets.

3.1 Overview of the datasets

The datasets are essential because they will be the base for the further input for an ANN. In this part, I'll give an overview of available datasets and discuss their unique properties.

3.1.1 Alexela electricity consumption

Alexela Energy provided two separate datasets with electricity consumption data per hour: the first included the electricity consumption recorded from April 1, 2016, until April 30, 2018, and the second – from September 1, 2017, until October 31, 2018. The difference between these datasets lies in the different selection of the clients. The first dataset included only a part of all clients, whereas the second set contained the consumption statistics regarding all clients.

The datasets included separate information for private and business clients. At the same time, the information about the exact client number was not available. It was one of the requirements from Alexela Energy that the solution would not be dependent on the number of clients.

One of the reasons for such a decision is that clients can be very different. One business client can be, for example, a small office with one air conditioner, where two people work on computers. Another business client can be a large educational institution or a factory, with many large-scale pieces of equipment. The electricity consumption profiles of these two business clients are considerably different.

Another reason is that the client number is not constant and continuously changes. As outlined in the introduction, the customers can change their electricity supplier at any time according to their contract. It means that the average consumption of the electricity calculated per client is not constant and can fluctuate.

3.1.2 Solar panel data

Alexela Energy additionally owns a solar panel park and provided the dataset with an electricity amount they generated. The monitored period was from September 1, 2017, to November 11, 2018. This data shows how intense solar radiation is. If it is high, it leads to an increase in electricity consumption due to the use of air conditioning and ventilation.

Solar panel data had several problems:

- Because its starting date is September 1, 2017, it would result in the inability to use Alexela's consumption data from April 1, 2016, until August 31, 2017, which is a massive reduction in the amount of data I would have for training purposes.

- An additional problem is associated with the predictions of the solar radiation. This kind of information does not usually appear in the weather forecasts, meaning I would have to predict it using some separate model. In the sum, I would make a prediction for electricity consumption based on the forecast for solar radiation, which would lower the model's total accuracy level.

- The data was not completely clean, e.g., sometimes in the night, solar panels continued to produce a small amount of energy. It meant the additional postprocessing of this data was necessary to use it.

Due to these reasons, I decided not to include the solar panel data in the input vector.

3.1.3 Historical weather data

The historical weather datasets were taken from rp5.ru in Excel datasheet format. I've used the weather history data individually for Tallinn and Tartu. It was necessary to take in consideration two cities separately because a significant amount of all clients of Alexela Energy live in these cities. The next reason is that the weather in these cities gives a reasonable estimate of the weather situation in the whole Estonia due to their different locations. For my further tests, I've used three weather parameters: temperature, pressure, and humidity.

3.1.4 Weather forecasts

The weather forecasts for the next two days per hour I've got from yr.no in eXtensible Markup Language (XML) format. Again, similarly to the weather history, the information for Tallinn and Tartu was separately received and processed.

Humidity can be defined in several different units, e.g., rp5.ru used mmHg and yr.no forecast hPa (hectopascal) units. Therefore, an additional conversion using equation was needed for hPa values to convert them to mmHg [32] to make the humidity values comparable.

3.1.5 Estonia electricity consumption

Estonia consumption history per hour is publicly available on Elering.ee.

I've conducted a several days observation on consumption values to find out if they correct the data afterward. It turned out the data was only once slightly adjusted for the most recent hour, making it a reliable source. Another benefit consists of the fact that Estonia consumption data is available with only a small delay, less than one hour.

3.1.6 Timestamps

One of the inputs can be time, day of the week, month, week number, or day of the year.

A typical problem associated with the cyclical parameters, e.g., time and month, is that if it is presented just as a numeric value, an ANN won't recognize that 23 and 0 hours are the neighbors exactly as December and January (12th and the first months). Such a problem can be solved by presenting the data using sine and cosine values, with the help of which it would be possible to preserve the cyclical nature of the data [33].

However, this approach does not necessarily make sense to use universally. For example, all weekdays are unique and have different electricity consumption profiles. Therefore, for weekdays, it is better to use a normalized day number from 1 (Monday) to 7 (Sunday).

3.1.7 Holidays and pre-holidays

There are several days in the year, which have typically completely different consumption profile than others. These are the holidays and the days before holidays, in other word pre-holidays. Their consumption profile can be to some extent compared with Sundays.

I’ve used publicly available holiday calendars in Estonia for 2016-2019 years and flagged accordingly such days.

3.2 Overview of the tools

3.2.1 Programming languages

Python as a programming language which suits the best for building an application for STLF. The main reason is that Python and its libraries cover the complete spectrum of scientific needs – from data preprocessing to ML frameworks.

For creating an application for STLF, I’ve used Python 3.6.8. A GUI was created using JavaScript (JS) framework Aurelia and web framework Bottle.

3.2.2 Machine Learning libraries

In Python, there are several libraries available, which allow building an ANN application. The most popular include TensorFlow and Keras. They are both considered as gold-standard libraries for ML purposes in Python. They are well documented, regularly updated, and supported.

I’ve used Keras, which is based on TensorFlow and offers a higher-level abstraction layer for TensorFlow Application Programming Interface (API). For hyperparameter optimization, I’ve used Hyperas library, which is built on top of Hyperopt.

3.2.3 Libraries for working on datasets

For working with datasets, there exist two main approaches: NumPy and Pandas.

NumPy supports a wide range of matrices operations, like addition, multiplication, and transpose. One of the downsides of using NumPy is that it’s tougher to keep an overview of large two-dimensional matrices, e.g., which contain 20.000 rows and 20-30 columns. Often, it’s needed to operate with columns separately, which also makes debugging more difficult.

An alternative approach for working with datasets is Pandas. It allows operating on datasets as a table, offering scientific operations, similar to Structured Query Language

(SQL) functionalities and Excel-like view options. Additionally, it supports column names, which makes working with many columns at once more straightforward.

Other essential libraries that I've used include:

- scikit-learn, which offers additional scientific features, e.g., splitting the datasets into the train and test sets, metrics evaluations, data standardization, and normalization.

- matplotlib.pyplot for plotting training progress graphs, electricity consumption graphs, and metrics visualizations.

3.3 Overview of the process

3.3.1 Prediction method

Electricity consumption statistics of Alexela Energy clients is not available approximately for the last two days. It's one of the challenges, which considerably limits the possible approaches.

One of the straightforward ways to predict the electricity consumption is the same-day technique – it consists in assuming that the electricity demand will be the same as the previous week. The precision of this forecasting method can be further improved by additionally correcting the predictions based on several factors, e.g., weather conditions like temperature, pressure, and humidity, the length of the day, knowledge of the holidays, Estonia consumption data, knowledge of the time, day of the week, week number and month.

3.3.2 Translation into a Machine Learning problem

In case of electricity prediction for Alexela Energy clients I want to predict the electricity consumption, in particular, the percentage change of the electricity consumption in contrast to the last week, in other words, a number. It means this is a regression problem, as outlined in 2.2.1. Because the historical data is present, the ML algorithm is not going to figure out the consumption change on its own, which means the learning is supervised.

3.3.3 Prediction accuracy evaluation

The approach for forecasting the consumption is based on correcting the previous week consumption. The actual output is the percentage change in electricity consumption in contrast to the last week. Because of such a particular method of calculation, there exist two different errors: model and prediction errors.

The model error shows how wrong the percentage change in electricity consumption in relation to the last week was predicted. This is a method-specific error, as shown in Equations 3.1-3.3, where *c* stands for a consumption. During my further tests, the model error was plotted as a subplot for evaluation purposes.

$$c_{difference} = \frac{c - c_{last\ week}}{c_{last\ week}} \tag{3.1}$$

$$absolute\ error = |\ c_{difference} - c_{difference\ predicted}\ | \tag{3.2}$$

$$model\ error = \frac{1}{n}\sum_{k=1}^{n} absolute\ error_k \tag{3.3}$$

The calculations for a prediction error, which shows how wrong the actual prediction is, are shown in Equations 3.4-3.6. This is the actual metrics, which Alexela Energy uses for measuring the forecast accuracy.

$$c_{prediction} = c_{last\ week} * (1 + c_{difference\ predicted}) \tag{3.4}$$

$$absolute\ percentage\ error = \frac{|\ c_{prediction} - c_{actual}\ |}{c_{actual}} \tag{3.5}$$

$$prediction\ error = \frac{1}{n}\sum_{k=1}^{n} absolute\ percentage\ error_k \tag{3.6}$$

3.3.4 Train, validation and test datasets

The concept of using an ANN includes dividing the complete dataset into three parts [34]:

- The train set is used directly for training an ANN. It helps an ANN learn trends, correlations, and dependencies of the dataset. As described in 2.2.2, the actual training lies in the tuning of the parameters during the backpropagation phase.

- The validation set is used for checking how the training is performed; if it improves the model with each epoch. At the same time, the data in the validation set is not used for training an ANN; it's explicitly for evaluation purposes only. The additional use for a validation set is tuning hyperparameters. Plotting training and validation sets' metrics on one graph allows evaluation of how ANN training is progressing. Several problems can be diagnosed, e.g., overfitting, when an ANN remembers the train set. As a protection for such cases, Keras offers callbacks, in particular, an Early stopping approach, which I utilized during my tests. When there is no further improvement in the validation set, the training will be stopped to prevent overfitting.

- The test set is used to check, how well the ANN has built the possibility to generalization. Equally to a validation set, the test set is not used for training purposes. It includes previously unseen data for an ANN. Even the validation set cannot be used as a test set, because it showed the influence on the training results.

4 Results

In this part, I present the results of my work. First, I investigate the ANN possibilities to build a model which provides reliable forecasts and as precise as possible. This part can be divided into five steps, as shown in Figure 5.

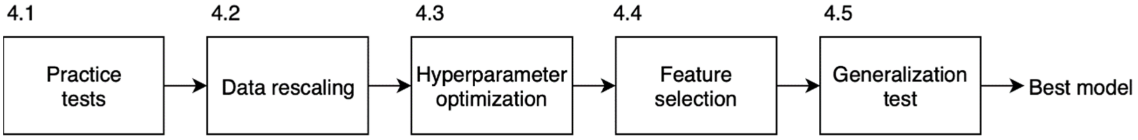


Figure 5. Steps of finding the best model.

Then, after finding the best model, I implement the UI and build an executable for the forecasting application.

4.1 Practice tests

4.1.1 Initial start of an Artificial Neural Network

The first tests were conducted using Keras as an ANN framework using the settings, as shown in Table 2. The input vector, number of layers, neurons, and activation functions were initially set according to the best results of preliminary research described in 1.3 [4].

Table 2. Initial settings of ANN taken from the preliminary research work.

Input vector with seven inputs		• The average temperature of Tallinn and Tartu • The average temperature of Tallinn and Tartu a week ago • The hour in sine representation • The hour in cosine representation • Normalized form of a number of a weekday (day fraction) • Month number in sine representation • Month number in cosine representation
First hidden layer	**Number of neurons**	6
	Activation function	tanh

Second hidden layer	**Number of neurons**	2
	Activation function	Sigmoid
Output layer	**Number of neurons**	1
	Activation function	Linear

Other ANN parameters and test-specific settings were set, as shown in Table 3. An optimization algorithm was set to Adam because it uses an adaptive learning rate and demonstrates reliable and quick convergence [35].

Table 3. Other ANN parameters and test-specific settings.

Holidays	Removed
Epochs	200
Batch size	200
Optimizer	Adam
Loss function	MSE
Metrics	MSE, MAE
Validation split	10%
A test set	2 last months (September and October 2018)

One of the obstacles I've faced at the beginning was the need for the reproducible results. ANN always at the beginning initializes its weights and biases randomly, as outlined in 2.2.2, which leads to different outcomes. To make the results repeatable and thereby comparable during my experiments, I fixed the seeds and made some additional changes to Keras backend configuration according to [36].

The model error lied between 4.23%-5.50% depending on the number of epochs, as shown in Table 4. With the epoch number equal to 400, I've achieved a model error of 4.23%, which was the best model performance seen so far.

However, it needs to be noted that choosing the epoch number based on the performance of the testing set is not the right approach. Instead, the validation set should be used for this task. Initial different settings of epochs were necessary to get an initial feeling on how the ANN behaves and what to expect.

Additionally, Keras offers a callback feature which gives a possibility to stop the training when no improvement in the validation set occurs over a specific number of epochs. The training was interrupted by this technique at the epoch 181, as shown in Table 4, meaning that from epoch 81 to 181, there were no improvements achieved in the loss measured on the validation set. At the same time, if I set the epoch time equal to 100, it was apparent that the training was not yet completed.

Table 4. First tests of ANN using Keras and initial settings with varying epoch number.

Epochs	**Model error (the lower, the better)**
15	5.50%
50	5.18%
100	4.79%
181 (Early stop, patience=100)	4.49%
200	4.36%
300	4.33%
400	4.23%
500	4.44%

Therefore, I’ve chosen as a starting point number of epochs equal to 200, which was the closest to the point, where the Early Stop algorithm originally fired. Further results with more epochs should be therefore treated cautiously.

For a visual representation of an output directly produced by an ANN, a graph with the percentage change of electricity was generated, as shown in Figure 6.

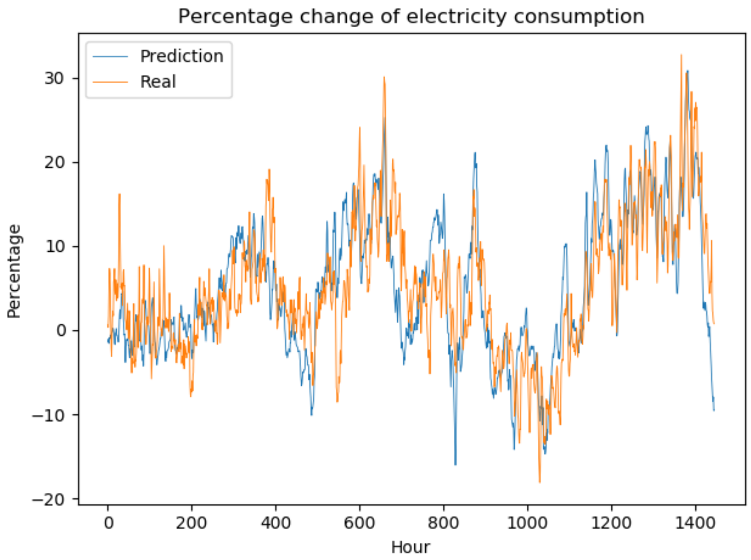


Figure 6. The direct output of ANN (prediction) is plotted against the known percentage change of electricity consumption compared to the last week (real); two test months, September and October 2018 are plotted on the x-axis hourly.

Another graph was built to evaluate the calculated predictions of electricity consumption compared to a real usage profile for a week, as shown in Figure 7.

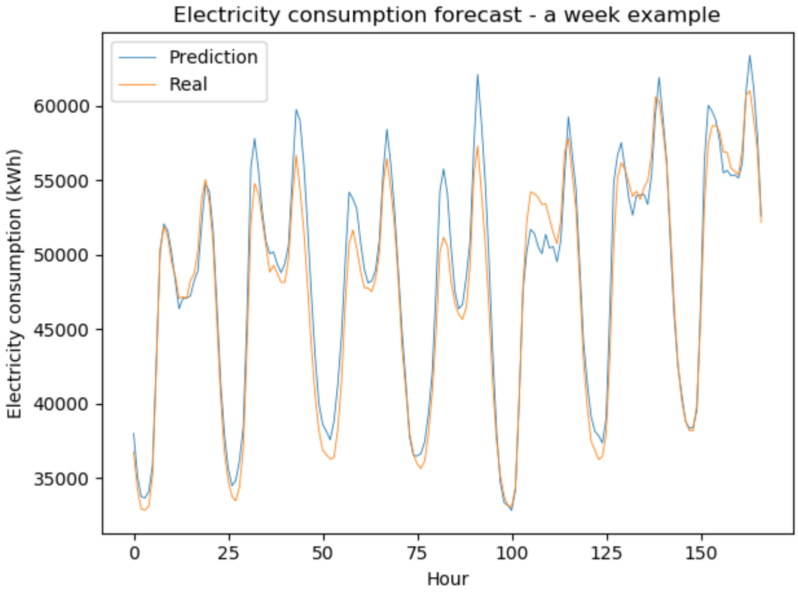


Figure 7. The actual prediction of the electricity consumption in kWh is presented for one week.

4.1.2 Same-day approach

If only the same-day approach were used by not correcting last week consumption, the model error would be 7.18%. However, this does not mean that ANN will produce always better results than this baseline performance. The model error can be higher than 7.18% if the predictions are made in the wrong directions. Only the ones that correct the percentage change of consumption in the right direction can make the forecasts more precise by bringing both model and forecast errors closer to the target of 0%.

4.1.3 Day duration

To the initial input vector, the new feature was added – the day duration of the specific day. The day length was calculated using CBM-model [37]. This is the equation, which uses the number of the day of the year and the latitude coordinate of the specific city [37].

I conducted experiments in both standardized and not-standardized forms of day duration, as shown in Table 5. Overall there were no significant improvements observed.

Table 5. Results of adding a day duration to the input vector.

Epochs	**Model error (the lower, the better)**		
	Non-standardized day duration	**Standardized day duration**	**Reference**
200	4.48%	4.51%	4.36%
300	4.77%	4.55%	4.33%
400	4.59%	4.33%	4.23%
500	4.53%	4.36%	4.44%

4.1.4 Removal of temperature from the input vector

The straightforward way to check if the temperature is playing a role for forecasting is to remove a current temperature and temperature a week ago columns and examine if this change affected the forecast precision. The hypothesis was that time, day of the week, and the month is enough information to produce a forecast.

The model error of this test was 6.36%, significantly worse than with temperature columns (4.36%). In Figure 8 is revealed, how the quality of the forecast degraded dramatically. The day and week consumption trends were only recognizable in the miniature form. In the middle, there was a jump in the prediction line, exactly where the month changed from September to October 2018. Most importantly, the overall consumption trend change was missing.

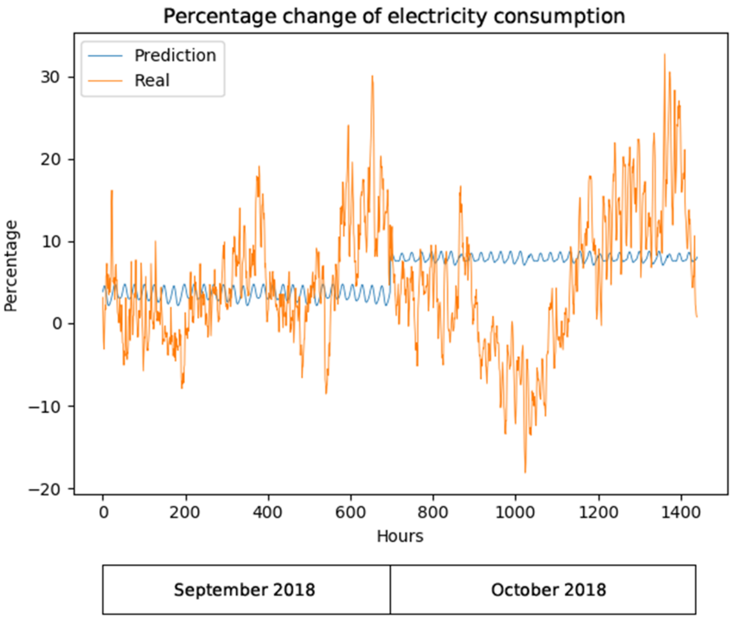


Figure 8. The temperature was removed entirely from the input vector, which caused a significant decrease in the quality of the forecasting. A jump between two months in the prediction is visible.

4.1.5 Degree-hour

The temperature is one of the most frequently used features for STLF. The core idea lies in the fact that in the summertime, when the temperature is higher, clients use ventilation and air conditioners. Similarly, in the winter time, the temperature is lower, and clients use heating.

Between those two phases, there is a transitional period, when it's whether too hot nor cold, so there is not an increased need whether for air conditioners nor heating. During the research project, a degree-hour analysis was conducted, and the following results were received: the lower bound at 14°C, and the upper bound was at 28°C [4]. The hypothesis is that the temperature change between those values does not lead to a weather-based electricity consumption change and therefore, can be omitted [38]. Temperatures higher than upper bound and lower than lower bound will be transformed to the relative temperatures by using Equation 4.1 [4].

$$t_{degree\ day} = \max(bound_{lower} - t, 0) + \max(t - bound_{upper}, 0) \qquad (4.1)\ [4]$$

The utilization of the degree-hour technique showed an improvement for the forecasting results made by a linear regression model [4]. When I applied the degree-hour approach to ANN, the precision of the electricity consumption forecasts decreased; the model error became 4.61-4.70%. As shown in Figure 9, on the top graph (a), the prediction line is nearly flat during the first 200-300 hours. On the bottom graph (b) is visible, that the temperature information during this period is almost entirely removed because of degree-hour. It shows that indeed, the temperature attribute is more critical then firstly believed.

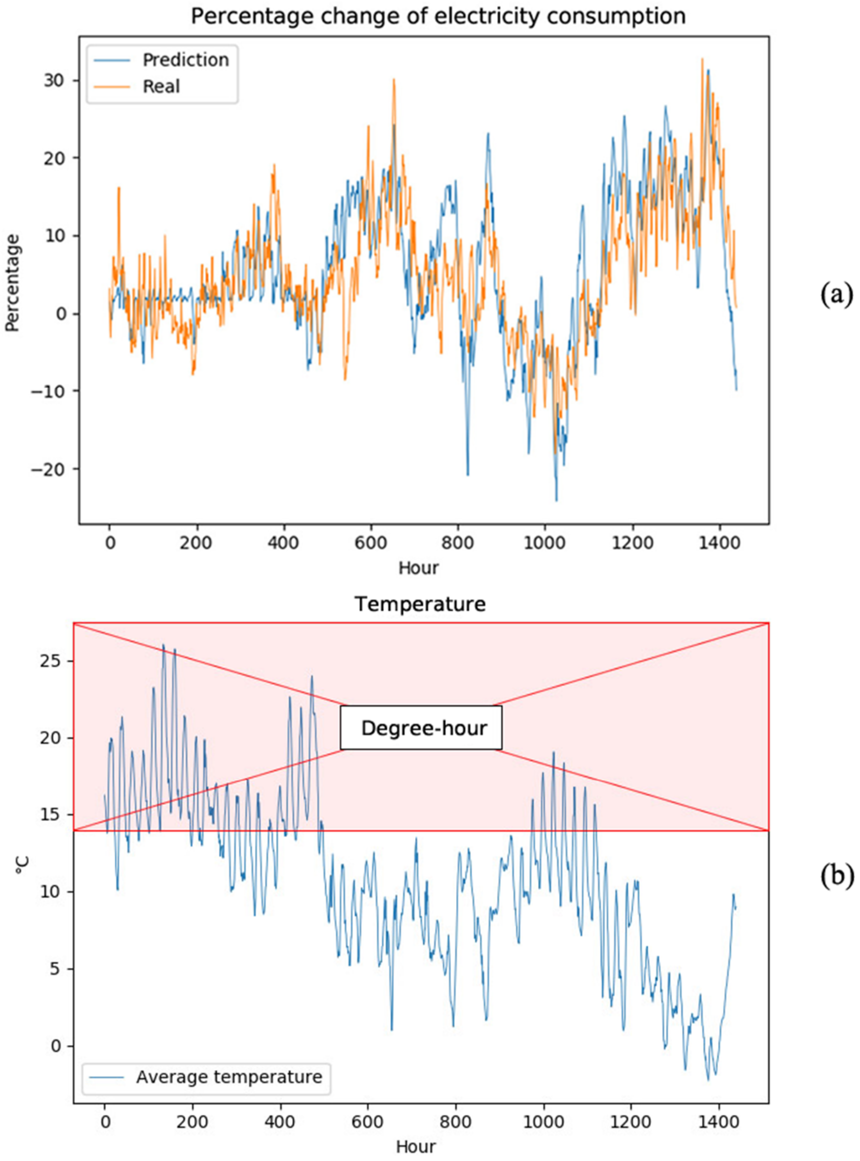


Figure 9. Graphs of the percentage change in electricity consumption (a) and temperature (b). The first 200-300 hours showed that the missing temperature information between 14°C and 28°C led to a forecast quality degradation.

4.1.6 Weighted temperature

As previously outlined in 3.1.1, the Alexela Energy clients are distributed within Estonia. Still, weather history and forecasts are made only at specific spots. Just a particular measurement of temperature in Tallinn does not imply, that the same temperature is precisely the same across all the Estonia. This fact led to a decision to use Tallinn and Tartu temperatures as a mean value, to give a rough estimate of the temperature in Estonia.

However, the number of clients is not equal in Tallinn and Tartu. There are more Alexela Energy clients in Tallinn, which shows that the temperature in Tallinn might play a more significant role in the forecasts than the temperature in Tartu. This is another hypothesis from the research project [4], which needed to be checked on the ANN.

Coefficients were calculated: for temperature in Tallinn was chosen 0.62 and for Tartu – 0.38 [4]. Instead of a mean value, now the temperature in Estonia would be a weighted sum of temperatures in Tallinn and Tartu, as shown in Equation 4.2 [4].

$$t_{weighted} = 0{,}62 * t_{Tallinn} + 0{,}38 * t_{Tartu} \qquad (4.2)\ [4]$$

The results of calculating a weighted sum instead of a mean value for temperatures are shown in Table 6.

Table 6. Tests on weighted temperature.

Epochs	Model error (the lower, the better)	
	Weighted temperature	Mean temperature (Reference)
200	4.39%	4.36%
300	4.39%	4.33%
400	4.24%	4.23%

The weighted temperature produced a little worse outcome than the mean version.

4.1.7 Weekday coefficients

The electricity consumption profile is unique for each of the weekdays. Weekdays also have a repetitive pattern; this is why a same-day approach described in 4.1.2, produced surprisingly good results for such a simple prediction way.

The hypothesis, proposed during the research project [4], was that the normalized days of the weeks (day fractions) could be replaced with the weekday coefficients. They were calculated during the statistical analysis and represent the relative electricity consumptions on these days. As shown in Table 7, the weekday coefficient is set to 1 on Thursdays, because the electricity consumption profile in the middle of the week is the highest. On weekends coefficients are accordingly smaller.

Weekday coefficients preserve the uniqueness of each weekday because each factor is used only once, but at the same time delivers additional information about the expected relative consumption on this day.

Table 7. Weekday fractions and coefficients for representation for a number of the weekday.

	Monday	Tuesday	Thursday	Wednesday	Friday	Saturday	Sunday
Weekday fraction	1/7	2/7	3/7	4/7	5/7	6/7	7/7=1
Weekday coefficient	0.97	0.98	1	0.96	0.95	0.82	0.78

The result of adding or exchanging weekday coefficient instead of weekday fractions is shown in Table 8.

Table 8. The result of tests related to weekday coefficients.

Epochs	**Model error (the lower, the better)**		
	Day coefficient instead of day fraction	**Day coefficient in addition to day fraction**	**Reference**
200	4.62%	4.54%	4.36%
300	4.82%	4.85%	4.33%
400	4.62%	4.71%	4.23%

In comparison to the reference, the precision of the models decreased.

4.1.8 Role of datasets

As outlined in 3.1.1, I had two different Alexela electricity consumption datasets available. Although the percentage change was calculated within each of those datasets, the number of clients that were tracked was considerably different. The first dataset covered only a small sample of all Alexela Energy clients. It was necessary to check, whether this first dataset provided an improvement in the prediction precision or it would be better to remove it and work only with the second dataset. The result is shown in Table 9.

Table 9. Test results on the importance of the Alexela electricity consumption datasets.

Datasets	**Model error (the lower, the better)**
Only the first dataset	4.95%
Only the second dataset	4.73%
Both datasets	4.36%

I’ve found out, that using both datasets, ANN provided a better forecasting result.

4.1.9 With or without holidays and pre-holidays

Holidays and pre-holidays are the most difficult days to predict and work with. The main problem is that the profile of electricity consumption on holiday is different in contrast to the previous and next weeks. For calculating the electricity consumption on the next week, it's necessary to take as a baseline the electricity usage on holiday, which will lead again to a lower precision.

One of the solutions to these problems is to deal with the holidays separately. It's possible to map each holiday as such using holidays datasets, and accordingly do the same with the pre-holiday list. This will allow an ANN to learn the unusual behavior of holiday days. I've conducted four tests, as shown in Table 10.

Table 10. Tests on datasets with or without holidays and pre-holidays.

Dataset	**Model error (the lower, the better)**
All included	4.66%
Non-holiday	4.36%
Non-preholiday	4.70%
No holidays and no preholiday	4.56%

By not adding holidays, it was possible to improve the forecasting precision.

4.1.10 Standardization vs. normalization

Standardization or normalization are both techniques for removing the units from the features. It is often applied for datasets because it allows an ML algorithm to better map relationships due to unitless attributes. These two different approaches are described in Table 11.

Table 11. Comparison between standardization and normalization.

Rescale method	**Equation**	**Output**
Standardization	$x_{standardized} = \frac{x - \mu}{\sigma}$ μ – mean value σ – standard deviation	(-∞,∞)
Normalization (Min-Max scaling)	$x_{normalized} = \frac{x - x_{min}}{x_{max} - x_{min}}$	[0,1]

Normalization is generally recommended to be used with ANN [39], but it does not make sense to use it universally. E.g., temperature values are never constant, and nobody guarantees that the maximum temperature this year would not be higher than the maximum temperature in the training data. Normalization is an excellent choice for, e.g., Red-Green-Blue (RGB) colors, where it's known that the values can only lie between [0, 255]. Because the maximum and minimum parameters are not constant for such data like temperature, humidity, pressure, and Estonia consumption and can vary, standardization is a more suitable approach, because it works better on the data, which shows a Gaussian distribution [39] or includes natural outliners.

Another critical fact to note, normalization or standardization techniques should be used in the right order. First, the scaler should be applied to the training data and scaler settings, like mean and standard deviation in case of standardization or x_{min} or x_{max} in case of normalization, should be calculated and saved. Only then it's correct to apply the same scaler with the saved parameters on a testing set. If the scaler parameters are calculated on the whole dataset, we directly influence the outcome [40]. The reliable results can only be guaranteed if the test set is not used in any way during the training or validation phase.

The first tests included the complete data standardization of all input columns; it resulted in a model error from 4.70% to 5.26% with varying epoch number. Then I've conducted the tests on the standardization on only the first two columns – temperature columns: the model error was in the range between 4.52% and 4.73%. Despite the expectations, the results were worse than without standardization.

4.1.11 Sensitivity to the temperature forecast accuracy

For building an electricity consumption prediction for the future, I used the weather forecasts. As with every prediction, these are never 100% accurate.

Although 1-2-day weather forecasts are typically close to reality, small errors and uncertainties might occur. The mean absolute deviation for such predictions is equal to approximately 1.67°C [41]. Therefore, it's necessary to check, how the electricity consumption forecasting accuracy will react if the temperature is shifted within ±2°C. The temperature shift in the complete learning data showed a slight improvement while changing in the positive direction within 1°C, as shown in Table 12. Shifting only in the test set at the same time showed this tendency much clearer – the forecast accuracy varied within 4.09-4.80% range. It shows that the weather forecast accuracy can influence a lot the electricity consumption forecast precision in a positive as well as negative ways.

Because of such a high fluctuation of 0.71% between the best and the worst models' errors, I decided to conduct the same test combined with the standardization to study, how much stability rescaling method will add. Whereas the partial standardization of only the temperature columns had almost the same high results difference of 0.67%, the complete standardization offered much more stable results at 0.27% but also less precise, as shown in Table 12.

Table 12. Test results on sensitivity to the temperature forecast accuracy within ±2°C range.

Temperature shift	**Model error (the lower, the better)**			
	In all learning data	**Only in the test set**		
	No standardization	**No standardization**	**Temperature columns standardized**	**All columns standardized**
-2 °C	4.38%	4.80%	4.96%	5.39%
-1 °C	4.44%	4.56%	4.75%	5.33%
-0,75°C	4.62%	4.50%	4.70%	5.31%
-0,50°C	4.45%	4.45%	4.65%	5.29%
-0,25°C	4.38%	4.40%	4.61%	5.28%
0°C	4.36%	4.36%	4.56%	5.26%
0,25°C	4.34%	4.31%	4.52%	5.24%
0,50°C	4.34%	4.27%	4.48%	5.23%
0,75°C	4.35%	4.23%	4.45%	5.21%
1°C	4.36%	4.20%	4.41%	5.19%
2°C	4.66%	4.09%	4.29%	5.12%

An additional more-realistic test was conducted on randomly shifting the temperature only in the test set, as shown in Table 13.

Table 13. Test results on randomly shifting the temperature only in the test set.

Temperature shift	**Average model error (the lower, the better)**
±0.5°C	4.73%
±1°C	4.78%
±2°C	5.22%

The error increased due to noisy temperature compared to the model error reference of 4.36%. The more random noise, the less accurate the model.

4.2 Hyperparameter optimization

The practice tests conducted in 4.1 gave some valuable hints, in what direction to continue the work. However, the main problem with this approach, it’s tough to get a complete understanding of what’s happening, because practice tests don’t give a full picture. To move forward, I needed some automatization approach to study the not-flat landscape of the cost function.

I decided to study if there is a room for improvement concerning the structure of ANN, in particular numbers of hidden layers and number of neurons, activation functions, batch size, number of epochs. All these parameters are often called hyperparameters.

According to [42], there exist four different methods to accomplish this assignment. The simplest one is the Manual Search, which involves trying by hand every single combination of hyperparameters [42]. A Grid Search is an automatization approach, which consists in building a grid or table of parameters, all combination of which will be tested [42]. A better way is to use a Random Search because it allows to speed up the process [42]. Seldom it is required to find a global minimum; this is why Random Search will provide a good enough solution in a reasonable time [42]. However, there exist even better and more advanced method which uses heuristics, which will lead to the suitable hyperparameters – a search using Bayesian optimization algorithm, also called Sequential Model-Based Optimization [42].

I’ve chosen the last method and found a suitable Python library, which offers such functionality – Hyperas. I’ve set up a search space, as shown in Table 14. The number of trials was set to 200. For the possible number of neurons, I’ve chosen to use all powers of 2, with a maximum amount of 64 neurons per hidden layer. As the main metrics, I've used a loss parameter measured on a validation set because this is the correct set for adjusting hyperparameters. Additionally, I've tested the performance of each model on the test set in both MAE and MSE metrics.

Table 14. A search space for hyperparameters.

Hidden layer 1	**Optional hidden layer 2**	**Output layer**	**Epochs**	**Batch size**
1, 2, 4, 8, 16, 32, 64	1, 2, 4, 8, 16, 32, 64	1	10, 20, 40, 60, 100, 200, 350, 500	32, 64, 128
Sigmoid, tanh, ReLU	Sigmoid, tanh, ReLU	Linear		

The best validation loss performance was achieved with two neurons in the first hidden layer with a sigmoid activation function. For finding the optimal neuron number in the second hidden layer, I've built a top list based on the multiplication value of all three metrics I've measured. The best overall performance was achieved with eight neurons in the second hidden layer again with the sigmoid activation function. In general, the sigmoid activation function performed very well in contrast to tanh and ReLU.

Another important observation was made on the number of epochs required. Between epochs 10 and 500, almost no improvement in validation loss was achieved, as shown in Table 15, leading to the conclusion, that there is no need in a very high number of epochs when the number of neurons in hidden layers is tuned precisely for the assignment. At the same time, the testing loss MAE was getting worse with increasing epoch number. The main criteria for choosing the suitable epoch number is always the observation of performance in the validation set. If no further improvements are achieved, then it's recommended to stop the training process to prevent overfitting. As a result, I concluded that the optimal number of epochs for this assignment was between 10 and 40 epochs.

Table 15. Dependency between the number of epochs and validation and testing loss parameters.

Epochs	**Validation loss MAE**	**Testing loss MAE**	**Testing MSE**
10	3.50%	4.06%	0.27%
40	3.50%	4.10%	0.27%
100	3.48%	4.23%	0.29%
350	3.48%	4.34%	0.29%
500	3.52%	4.34%	0.31%

The batch size is the parameter, which can be set up without doing such an advanced search. The general logic is, the more the batch size, the more memory will be utilized for training and the faster the training. At the same time, the more the batch size, the more epochs are required to achieve the same training results because recalculations of weights and biases occur less frequently. As a default, Keras uses the batch size equal to 32. Because Hyperas did all the test on the configuration with 2-8 neurons in the hidden layers with the batch size equal to 128, I decided to continue working with this size.

At epoch 30, the ANN with new hyperparameter configuration produced a model error equal to 4.03%. I conducted again a test on including and not including holidays and pre-holidays and found out, that including everything produced only slightly worse model error of 4.07%, which lead to a decision to continue working on the approach, which included both holidays and pre-holidays.

4.3 Data rescaling

Now I had the new hyperparameters which better matched the assignment. However, before moving on, it was necessary to make a decision, which columns to standardize. This is why I run additional, this time automated tests, and tracked the differences in the model performance.

First, I conducted a test, which investigated, if there is any difference in the model performance depending on the method used for data rescaling. Additionally, to get the complete picture of the influence of standardization or normalization, I decided to try all 2^7=128 combinations of 7 input features, on which I applied a transformation function.

The result has shown that applying standardization or normalization results in very close model performances with minor differences. In the second test, I've discovered, that using rescaling function on all the columns produced considerably decreased forecast precision then if not applying anything at all – 4.47% and 4.03% accordingly. Moreover, models performed better if no rescaling was used to the temperature columns.

4.4 Feature selection

With improved hyperparameters and understanding of the data rescaling, I moved to the next step – feature selection. Features are the attributes in the input vector.

In the assignment, there are a lot of possible input parameters: temperature, humidity, pressure, electricity consumption in Estonia, day duration, timestamps. However, more importantly, all this information can be presented differently, e.g., the temperature can be used for the time point of the forecast, a week ago value or even an absolute difference between those temperatures. At the same time, it's not correct to calculate a percentage difference between two temperatures using Celsius axis: the increase from 1°C to 2°C does not mean a 100% increase in temperature.

As a result, I ended up with 23 possible columns, as shown in Table 16, which meant there were more than 8 million possible combinations (2^{23}). Some of the columns belonged together, like sine and cosine representations of time, month or day of the year, because sine or cosine alone brings only half of the cyclical information. Therefore, I've counted them as one column. Humidity wasn't included in the vector for combinations, because it did not show promising performance improvements in the practice tests I conducted and it was a trade-off to make a total number of combinations smaller.

Table 16. Possible features or input columns.

Feature group	**Features**	**Rescaling**
Temperature	The average temperature of Tallinn and Tartu for a time point of prediction	
	The average temperature of Tallinn and Tartu a week ago	
	The absolute difference between the average temperature of Tallinn and Tartu for a time point of prediction and a week ago	standardized
Pressure/ humidity	Pressure/humidity for a time point of prediction	standardized
	Pressure/humidity a week ago	standardized
	The absolute difference between pressure/humidity for a time point of prediction and a week ago	standardized
	The percentage difference between pressure/humidity for a time point of prediction and a week ago	
Day duration	Day duration for a time point of prediction	standardized
	Day duration a week ago	standardized
	The absolute difference between day duration for a time point of prediction and a week ago	standardized
	The percentage difference between day duration for a time point of prediction and a week ago	

Estonia electricity consumption	Electricity consumption in Estonia two days ago	standardized
	Percentage difference in electricity consumption in Estonia two and seven days ago	
Holiday database	Holidays	
	Pre-holidays	
Consumption statistics	Day coefficient	
Timestamps	The hour in a 24-hour system in sine and cosine representation	
	Normalized form of a number of a weekday (day fraction)	
	Day number in the month	standardized
	Day number in the year in sine and cosine representation	
	Day number in the year	standardized
	Week number in the year	standardized
	Month number in sine and cosine representation	

Over 100 thousand of random input vector combinations were generated with a total column number between 6 and 12, where some of the columns, like sine and cosine of time, were counted as one. For every combination, a separate ANN model was built and benchmarked. As metrics, I used not only the validation loss and model error but also calculated the real forecast error. This allowed me to select one thousand of the best performing input vectors.

In this test, the forecast error was always better than the model error; the difference was on average 0.2-0.3%. The best combination produced a model error of 3.79% and forecast error of 3.61%.

4.5 Generalization test

At this point, I had a thousand of promising models. The only problem with them, they were tested on only two months: September and October 2018. As outlined in 2.2.2, one of the critical characteristics of ANN is a possibility to generalization – how well the model can work in new situations. For such assignment like STLF, this is crucial, that the model delivers reliable forecasts.

Because no new data was available, I decided to adopt a principle from a Cross-Validation (CV) method to this assignment, as shown in Figure 10. The classical validation approach consists in dividing the dataset into the fixed train, validation, and test sets. At the same time, CV doesn't have a separate validation set. Instead, it divides the whole dataset into the equally sized parts. A CV consists of iterations or folds. During the first fold, the first part is the test set and all others – the train set. During the second fold, the second part becomes the test set, and accordingly, all others form the train set. The benefit lies in the fact that in the sum, all the information in the dataset will be utilized for testing purposes.

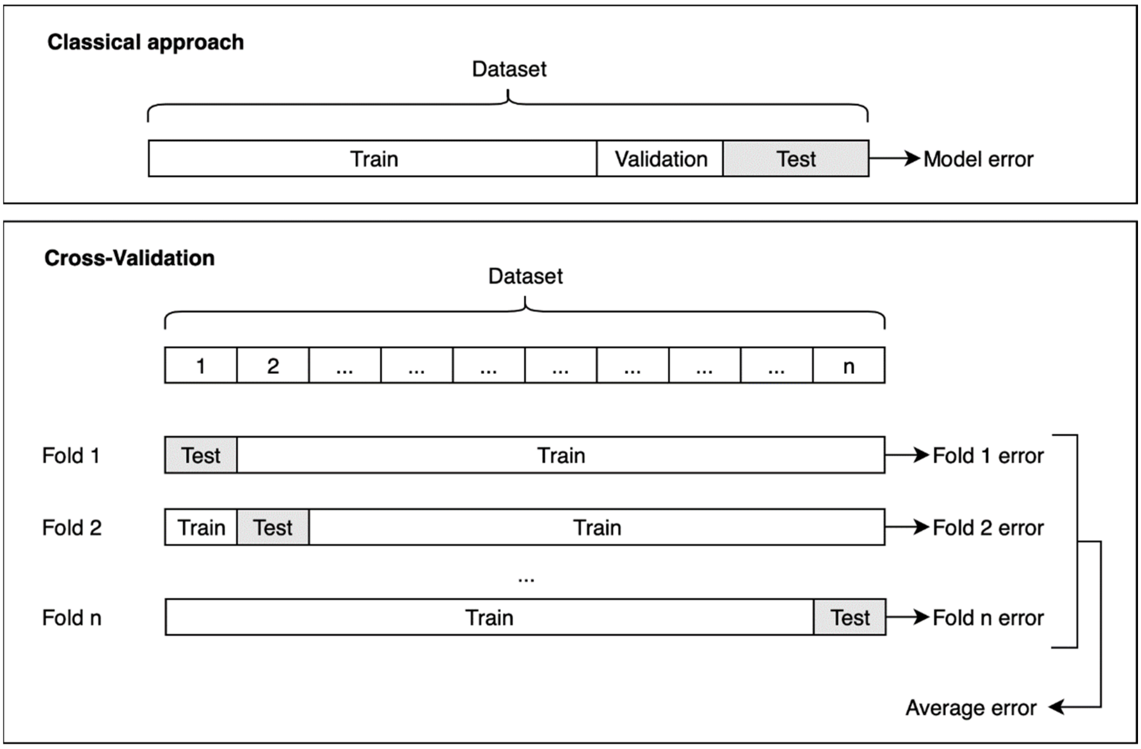


Figure 10. Principle of a classical approach with train, validation and test sets and CV technique.

CV principle suits perfectly for testing generalization capability. I ran this test using 1000 best-performing input vectors from the previous step and compared the average final results. The raw results in this test are not representative, at least due to the fact that Alexela dataset consisted of two different samples. However, the trial was essential for verifying, how well the model could behave in different situations. This test allowed me to narrow the selection of the best models, from which I've chosen one with 13 inputs, as shown in Table 17. The model with the new vector performed on average 0.17% better in CV set then the reference version.

Table 17. Features in input vectors.

New input vector (13 inputs)	**Reference (7 inputs)**
The average temperature of Tallinn and Tartu for a time point of prediction The average temperature of Tallinn and Tartu a week ago Day duration for a time point of prediction The percentage difference between day duration for a time point of prediction and a week ago Day coefficient Normalized form of a number of a weekday (day fraction) Holidays Electricity consumption in Estonia two days ago Percentage difference in electricity consumption in Estonia two and seven days ago The hour in a 24-hour system in sine representation The hour in a 24-hour system in cosine representation Month number in sine representation Month number in cosine representation	The average temperature of Tallinn and Tartu for a time point of prediction The average temperature of Tallinn and Tartu a week ago The hour in a 24-hour system in sine representation The hour in a 24-hour system in cosine representation Normalized form of a number of a weekday (day fraction) Month number in sine representation Month number in cosine representation

I conducted the benchmark using a new vector which contained 13 inputs and compared the results with the previous 7-input version on similar settings, as shown in Table 18. The model showed here also better performance – 3.80 against 3.98%.

Table 18. Comparison of model and forecast errors tested on 15 epochs.

Error	**New (13 inputs)**	**Reference (7 inputs)**
Model error	3.80%	3.98%
Forecast error	3.62%	3.78%

4.6 Graphical User Interface

There exist several methods of how a GUI can be created for a Python console application. One of the approaches consists in using toolkits like Tkinter or PyQt, which allow creating a User Interface (UI) directly in Python. Another method enables using Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), and JS framework capabilities to build an interface. The latter variant has a significant advantage because it allows creating a more interactive UI.

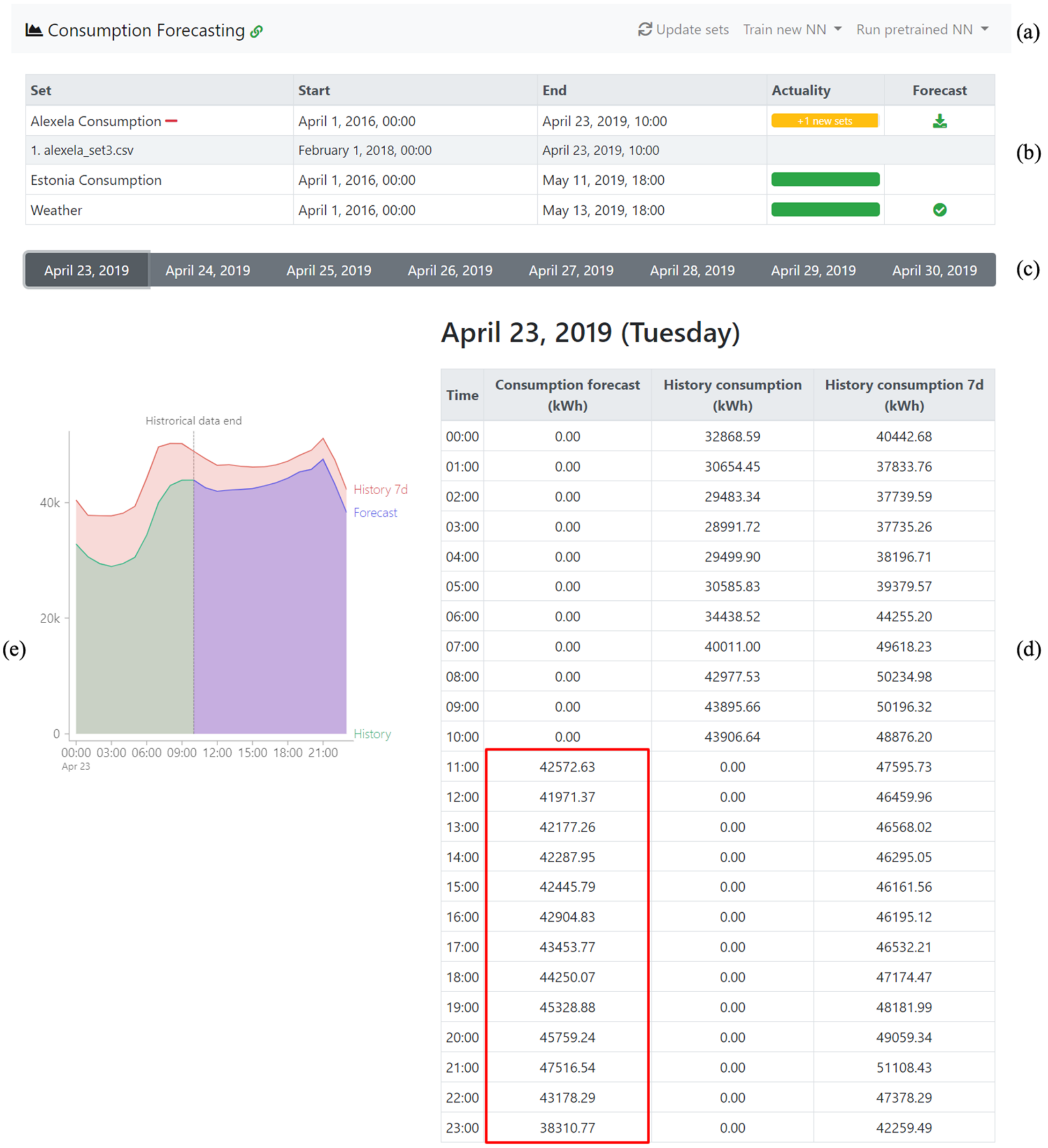
For building a GUI, I’ve decided to use the second approach. As a back-end side, I’ve used Bottle framework, which served as a controller for my ANN application. As a front-end side, I’ve used Aurelia as a Single-Page Application (SPA) JS framework and Bootstrap 4.3.1 as a CSS framework. MetricsGraphics.js was used for making graph representations. A comprehensive analysis of the architecture of the application can be found in 5.9.

One of the requirements was that the program would be a standalone application, which would be possible to use without installing Python. For accomplishing this, I used cx_Freeze. A configuration script was composed for building the package, which included all the needed libraries. As a result, an EXE file was generated, which can be executed on every Windows machine. With the start of the TensorFlow backend, the GUI opens automatically in the browser window.

The functionality of the program included:

- automatic update of weather history, weather forecast and Estonia consumption from API-s, their postprocessing, and merging with existing offline local data
- automatic merging of additional Alexela datasets using specified way and calculating the percentage differences only within each set; explicit representation of the order of the new datasets processed within the GUI.
- possibility to train new model or use a pre-trained model on both input vectors: with 7 inputs (reference) and 13 inputs (new)
- option to download a forecast in Comma-Separated Values (CSV) format
- a graph generation for each day, which visualizes the forecast and last week consumption

The screenshot of UI is shown in Figure 11. The GUI contains the menu, the information about loaded datasets, the possibility to choose the day for which the forecast will be shown, and the prediction table with the data visualization.



Set	Start	End	Actuality	Forecast
Alexela Consumption —	April 1, 2016, 00:00	April 23, 2019, 10:00	+1 new sets	
1. alexela_set3.csv	February 1, 2018, 00:00	April 23, 2019, 10:00		
Estonia Consumption	April 1, 2016, 00:00	May 11, 2019, 18:00	🟩	
Weather	April 1, 2016, 00:00	May 13, 2019, 18:00	🟩	✓

April 23, 2019 (Tuesday)

Time	Consumption forecast (kWh)	History consumption (kWh)	History consumption 7d (kWh)
00:00	0.00	32868.59	40442.68
01:00	0.00	30654.45	37833.76
02:00	0.00	29483.34	37739.59
03:00	0.00	28991.72	37735.26
04:00	0.00	29499.90	38196.71
05:00	0.00	30585.83	39379.57
06:00	0.00	34438.52	44255.20
07:00	0.00	40011.00	49618.23
08:00	0.00	42977.53	50234.98
09:00	0.00	43895.66	50196.32
10:00	0.00	43906.64	48876.20
11:00	42572.63	0.00	47595.73
12:00	41971.37	0.00	46459.96
13:00	42177.26	0.00	46568.02
14:00	42287.95	0.00	46295.05
15:00	42445.79	0.00	46161.56
16:00	42904.83	0.00	46195.12
17:00	43453.77	0.00	46532.21
18:00	44250.07	0.00	47174.47
19:00	45328.88	0.00	48181.99
20:00	45759.24	0.00	49059.34
21:00	47516.54	0.00	51108.43
22:00	43178.29	0.00	47378.29
23:00	38310.77	0.00	42259.49

Figure 11. GUI for a forecasting application contains the menu (a), information about available datasets (b), the possibility to choose a specific day for showing the predictions (c), the table with the hourly electricity consumption forecasts in the first column (d) with the data visualization (e). The forecasted electricity consumptions are marked with a red rectangle.

In the menu, there are three buttons: update sets, train new ANN, and run pre-trained ANN. Update sets button reloads all locally saved datasets and updates them with the information available from three API-s: weather history, weather forecast, and Estonia consumption. From the other two dropdown menus, it's possible to specify, which input vector should be used for an ANN – 7 or 13 input vectors.

Next is the table with the information about the datasets currently loaded, their start and end dates. For Alexela consumption there is a button which shows or hides the information about new datasets, which Alexela Energy can add in the particular folder. All the files in this folder will be scanned, sorted by the start date and merged in a specified way so that only the differences within each of the sets will be calculated.

Next is the bar, which offers the days, for which the forecasts were calculated. The forecast is generated for the maximal time possible, which depends on the input vector (2 or 7 days ahead), e.g., if there is used electricity consumption in Estonia 2 days ago, the maximum amount of forecast will never exceed this number.

The main focus in GUI was made on the explicit representation of the primary information Alexela Energy needs – the electricity consumption forecasts in kWh. The table contains three columns: the forecast column, Alexela real consumption or historical column, and Alexela consumption seven days ago. Every day is presented separately with information per hour. In the graph, there is the same information represented visually as in the table for better evaluation.

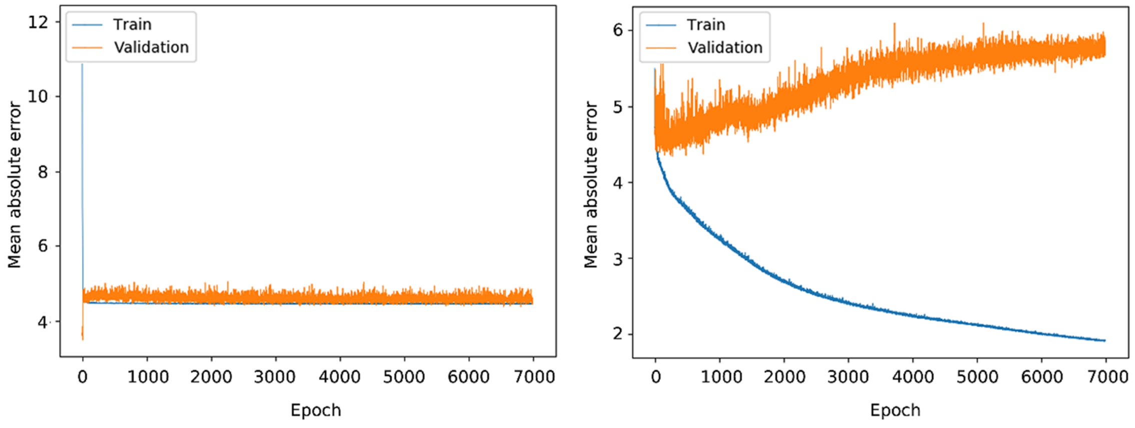
5 Analysis and discussion

In this part, I analyze and discuss the results. In the previous section, I increased the precision of the forecasting model. The first significant improvement was achieved by changing the structure of ANN and finding the suitable for the assignment hyperparameters and the second – by changing the input vector from 7 to 13 inputs.

5.1 Training evaluation

The common practice to evaluate the progress of the training is to plot train and validation errors on a graph. This way it's possible to diagnose problems associated with the ANN, e.g., overfitting. Instead of building a generalization capability, the ANN remembers the examples from the training set and fails when it comes to predicting new values.

The more complex the ANN structure, the more pronounced the overfitting typically can occur. I simulated the overfitting by adding one extra hidden layer and increasing the neurons numbers to 64 on every layer, as shown in Figure 12. On the left graph (a), the train and validation lines converge together, and even such large epoch number of 7000 does not cause significant signs of overfitting. On the right chart (b) is the classical overfitting, when training error is steadily decreasing, but validation error rises.



(a) (b)

Figure 12. Train and validation error graphs: 3-hidden layer with 2-8-1 neurons configuration (a) and 4-hidden layer 64-64-64-1 neurons configuration with simulated overfitting (b).

Too little neurons and a small number of hidden layers are also not rational because it will prevent ANN from showing its full potential in solving non-linear problems.

5.2 Forecast quality evaluation

The new vector included not only the temperature and timestamp information like the original input vector but also day duration, day coefficient, Estonia electricity consumption, and holiday information as previously described in Table 17.

For evaluation purposes, I generated plots with two graphs, where percentage change of electricity consumption and absolute model error, the difference of forecasted and actual percentage change, were plotted, as shown in Figure 13.

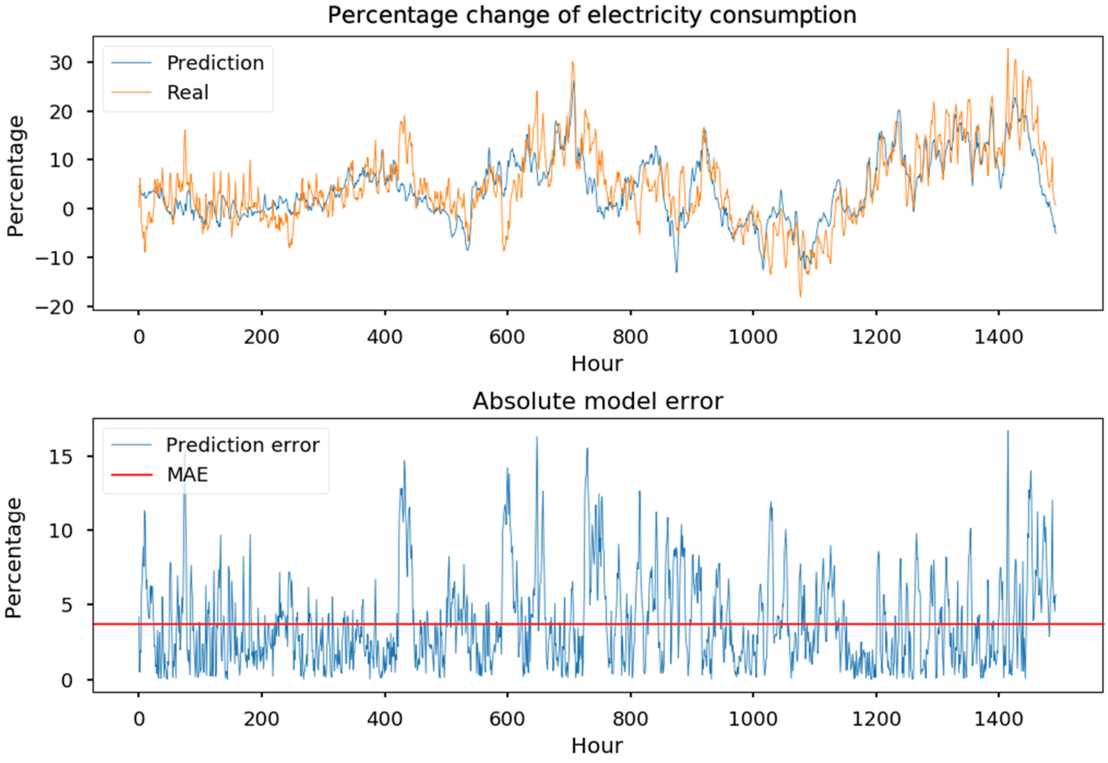


Figure 13. A detailed evaluation of the predicted values by ANN on two test months.

The weekly average forecast error using input vector with 13 inputs varied in the range from 2.22% to 4.87% with the average forecast error of 3.62%. The input vector with 7 attributes resulted in the forecast error from 2.23 to 5.06% with an average of 3.78%. Not only on those two test months, the 13-input vector performed better than with the 7-input, but the same behavior was also observed during the generalization test, which utilized the whole dataset.

To evaluate how correctly the forecasting model is predicting, a plot was generated for every week, which contained two graphs: forecasted electricity consumption and the absolute forecast error. The best forecast error is 2.22%, as shown in Figure 14.

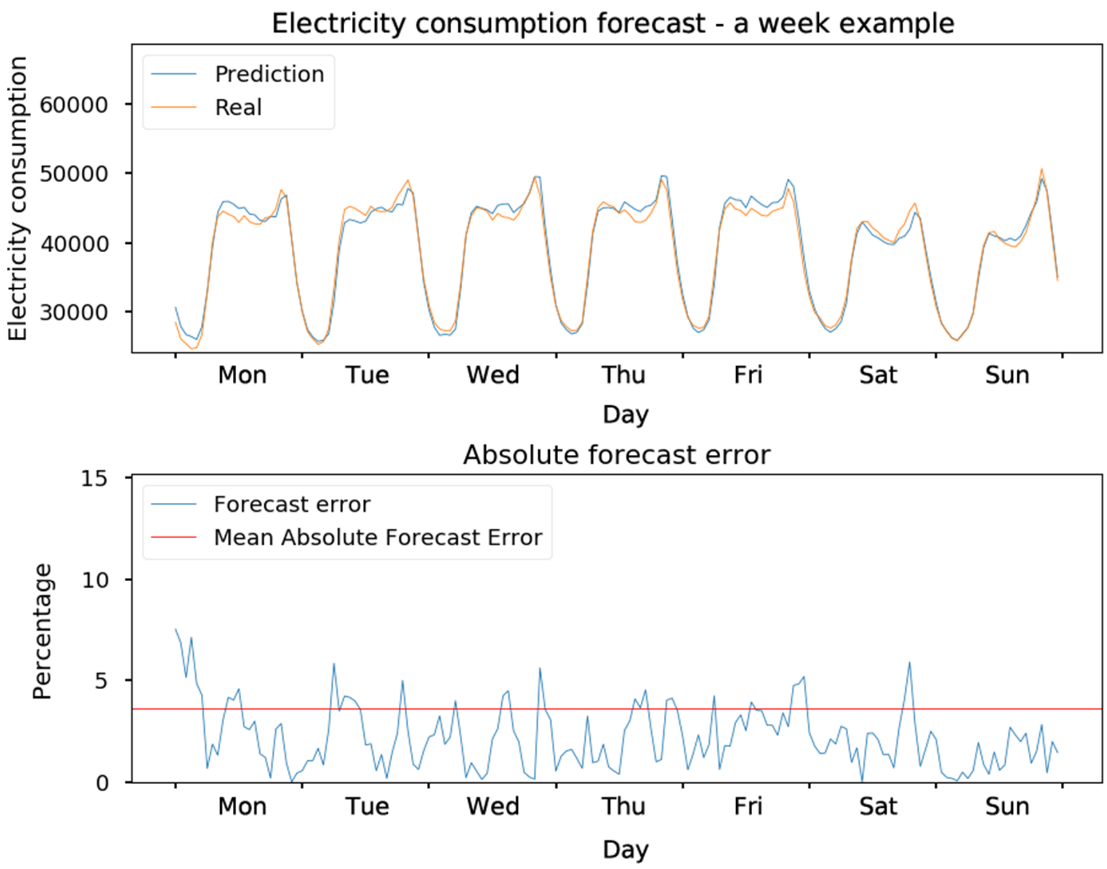


Figure 14. Calculated forecasts for one of the weeks with the best forecast error of 2.22%.

Some weekdays are predicted more precisely, like Thursdays and Saturdays, and some are more difficult to forecast like the beginning of the week and Sundays, as shown in Table 19.

Table 19. Day wise forecast error analysis.

	Vector with 13 inputs	**Vector with 7 inputs**
Monday	4.14%	4.16%
Tuesday	4.11%	4.18%
Wednesday	3.53%	3.94%
Thursday	2.86%	2.87%
Friday	3.55%	3.86%
Saturday	2.99%	3.15%
Sunday	4.11%	4.19%

The reason for varying forecast errors lies not only in the fact that the input vectors does not include the necessary knowledge. Another cause is in the method itself used as the basis for the forecast calculations – same-day approach. As the basis, the previous week's consumption is taken and corrected according to the input parameters. The problem occurs if the electricity consumption is unexpectedly considerably different from what was predicted. Not only it results in the momentarily increased forecast error on that particular day, but also leads to forecast deviation the same day next week because as the basis for its calculations previous week's consumption is used as illustrated in Figure 15.

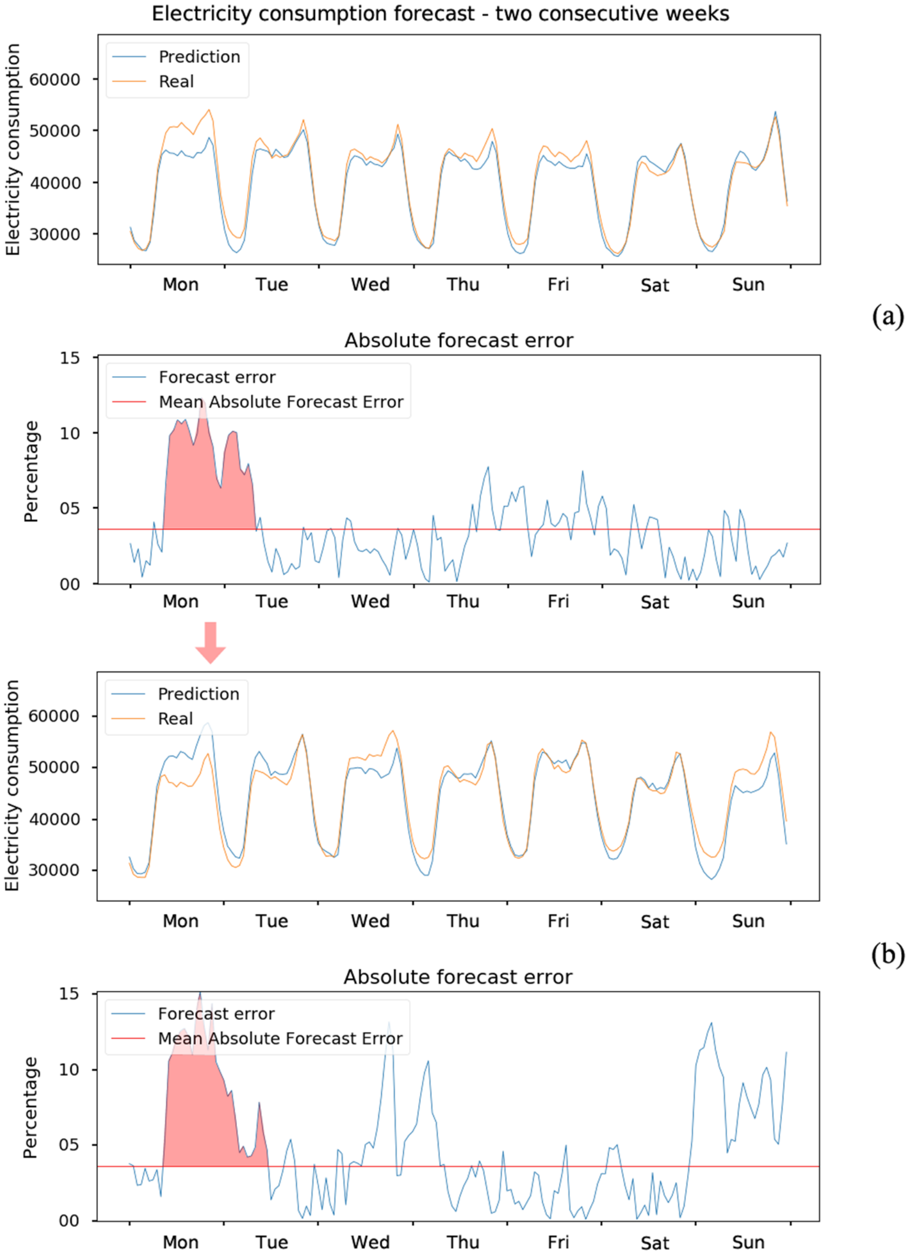


Figure 15. Similar-day approach transfers a forecast error on Monday and Tuesday from the first week (a) to the next week (b).

5.3 Cycles in the electricity consumption profiles

The electricity consumption profiles are complex and are non-linear by nature. In general, three central components in them can be distinguished: intraday, intraweek, and intra-year or seasonal cycle [43].

The intraday cycle describes the best behavior of the people during the day, as shown in Figure 16.

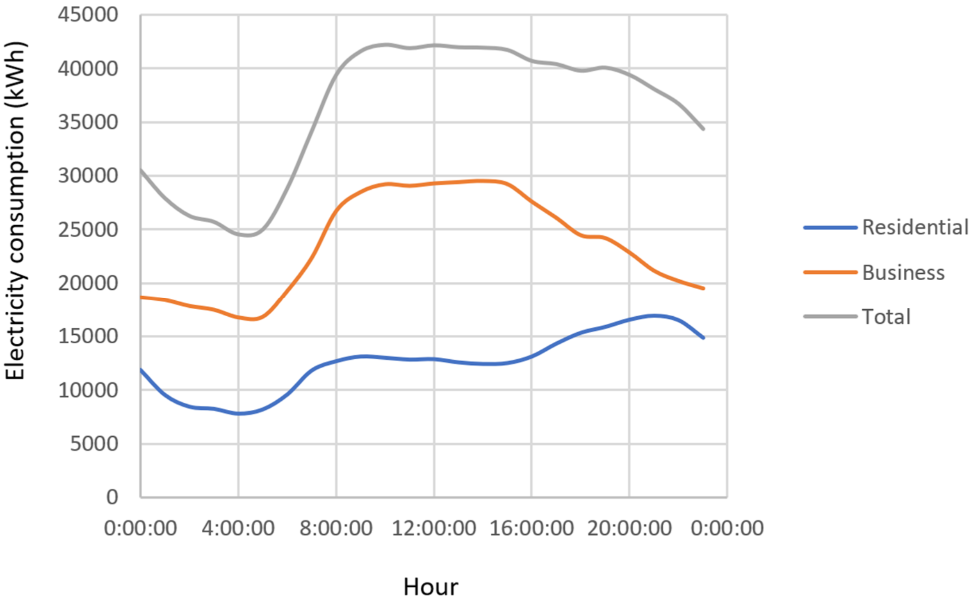


Figure 16. Electricity consumption profile of one of Wednesdays.

During the day hours, the total electricity consumption is considerably higher, when most of the people work than during the night when most of the people sleep. The rise of total electricity consumption begins at approximately five o ‘clock and has a plateau during 9-15 hours. Whereas business clients use more electricity during the day, residential customers consume more electrical power in the evening hours. However, even during the night, the electricity consumption is not near a zero; instead, it has some baseline, which won’t be crossed.

The intraweek consumption describes the electricity profiles on the different weekdays, as shown in Figure 17. As outlined in 4.1.7, it’s known that electricity usage during the work days is higher than on the weekend. The highest use during the week is on Wednesdays, which is located precisely in the middle of the working week. Some exceptions apply to an intraweek cycle, e.g., holidays and days, before the holidays.

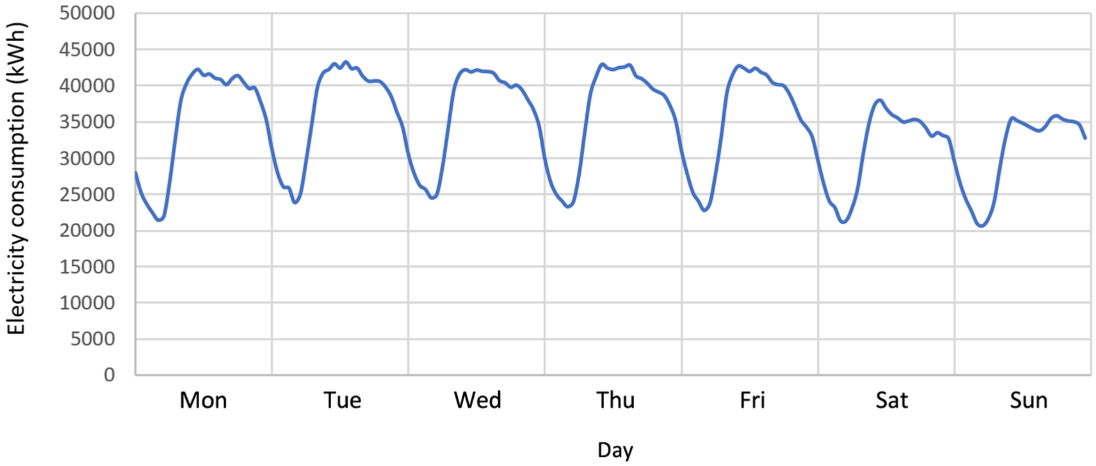


Figure 17. Electricity consumption during the week.

Intra-year cycle covers the seasonal part of electricity consumption profiles associated with the use of ventilation, air conditioning, heating, lighting, and people behavior, which differs much depending on the month and day length, as shown in Figure 18. From January to March – these are the most difficult months to predict mainly due to the large fluctuations in weather.

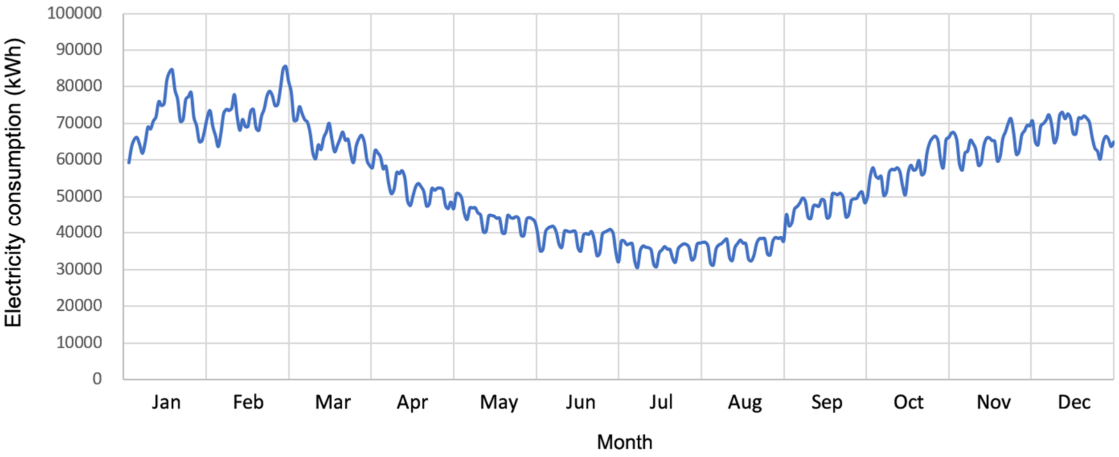


Figure 18. Total daily average electricity forecasting during the year.

5.4 Role of the weather

5.4.1 Theory

The temperature has a significant impact on an electricity consumption profile. In the summertime, when the temperature is higher, clients use ventilation and air conditioners. Similarly, in the winter time, the temperature is lower, and clients use heating. Within a day, logic is the following: the majority of people works during the day resulting in higher electricity consumptions and sleeps during the night resulting in lower usage of electricity. However, the temperature alone does not give a complete picture of the actual weather conditions.

Additionally, to the temperature, the atmospheric pressure plays a significant role in weather. Higher pressure is associated with no clouds and sunshine whereas lower pressure correlates with the clouds and rain. The denser the clouds are, the more they protect us from high atmospheric pressure leading to lower pressure values. Analogously, when there are no clouds, nothing protects from high atmospheric pressure, resulting in a pressure increase.

Humidity is another essential weather parameter. There exists a specific correlation between the humidity and our perception of the temperature. If the humidity is high, the temperature in the winter feels cooler and, in the summer, warmer [44] than only the temperature would suggest.

5.4.2 Practice

A comprehensive investigation was performed on the role of temperature. I want to outline three experiments. In 4.1.4, I've tried to delete the temperature from the input vector, which led to a significant degradation in the prediction quality. It looked like some skeleton or base structure was missing; therefore, the forecast was forming a flat line.

In 4.1.11 and 4.3, I've investigated how the model behaved when the temperature was shifted, and only columns containing temperature information or all columns were standardized. Against my prediction, the standardization produced worse forecast precision that if no rescaling was applied.

The likely reason is the importance of the temperature vector. When a rescaling technique is applied, different input vectors become unitless and consequently comparable. At the same time, an ANN cannot distinguish, which parameter plays a more significant role and which is less critical. Therefore, when I tried to standardize only the temperature columns or all columns, the temperature became less significant from the ANN perspective.

Some of the experiments were based on the results received from creating a linear regression model [4], e.g., degree-hour in 4.1.5 and weighted temperature in 4.1.6. Even though for a linear regression model, these techniques provided a great success, in the case of ANN, they did not give any improvement. The degree-hour experiment showed again that by removing some of the temperature information, the forecast precision significantly decreased.

The temperature had the most crucial role in the forecasting model of all-weather parameters I had. Temperature includes not only intraday information but also an intra-year or seasonal information and to some extent the day length information what makes the temperature so powerful yet simple. Both pressure and humidity were not capable of competing with the temperature and can be considered only as additional information.

5.5 Role of day length

Although some component of day length is contained in the weather information, the day length shows it more directly. Day length has a straight impact on people's behavior. The longer the day length, the more people spend time outside and not at home, because it's the summertime resulting in the smaller electricity consumption profiles. The shorter the day is, the more people spend time inside, the more they use room lighting, electronics, and heating.

5.6 Representation of the input vectors

Not only there are numerous different features available, but data for an ANN can be represented very differently. For example, the day duration; it's possible to include in the input vector the day duration of the day of the forecast, the historical day duration from the same day last week, their absolute difference or even percentage change of those two values. Additionally, there are rescaling capabilities to make different input vectors comparable with each other. It's more beneficial to apply a rescaling technique on columns, where data contains actual values or absolute differences, then where already a percentage change is calculated.

By analyzing the results, I've found a pattern, which input vector led to better results. Besides temperature columns, which were present almost in all combination which performed well, the percentage difference in electricity consumption in Estonia two and seven days ago was also consistently leading to better results. This parameter gave the overall trend, in which direction recently the use of electricity changed.

Another potent attribute was day number in the year in sine and cosine representation, which was present almost in all models, which performed well in the generalization test. It had some advantage over the classical month and day representation because it continuously covered the whole year without jumps. Vectors with it had shown even better performance than the one I chose. The main problem lies in the leap years, e.g., the year 2020 will be with February 29. Because the total number of days in the year will be different, it will lead to all modified sine and cosine values, which might affect the forecast precision. Some additional processing is needed to overcome this obstacle, e.g., convert February 29 to the second February 28. It's possible because the year was an attribute which was not included in the model or search space.

5.7 Size of historical datasets

The amount of training data for an ANN is significant. The more data, the better. In some research papers [45], even decades of historical information are used to make the predictions.

The next problem, Alexela Energy had different datasets, which covered only a specific sample of clients, which the company had at the beginning of the particular dataset. It means that the client number and electricity consumption pattern can change. Therefore, the forecast principle should be based on working with the differences rather than absolute numbers, and the same-day approach should be applied only within the same dataset.

To make it possible to add recent electricity consumption data or new datasets, in the forecasting application, I've implemented the particular way of merging the datasets, which calculates the differences only within the same dataset. The method consists in preserving of the one extra week of data from the previous set, which leads to the intentional generation of duplicate values in the date and time column, processing sets as usual and deletion of the duplicates date and time values while keeping the first occurrences. This way, the model is not dependent on the number of clients and clients from the different datasets are not compared with each other leading to comparable and consistent percentage changes.

Additionally, I investigated in 4.1.8, how ANN would work with a limited amount of data. The forecasting performance decreased when the dataset was limited and performed better with information, which included all available time. To sum things up, it’s crucial for an approach using ANN to provide the maximum amount of information available.

5.8 Preprocessing of the datasets

Often there was missing data in datasets, e.g., gaps in weather forecasts and Estonia consumption history. ANN is extremely sensitive to those problems. If there is even one value somewhere, which is Not a Number (NaN), ANN won’t work. Several cases can be outlined:

- completely missing timestamps
- missing individual values
- data values containing zero values, e.g., in Estonia consumption

Another problem is the Daylight-Saving Time. When the clock is moving backward in October, it leads to duplicate timestamp values. Similarly, when the clock is moving forward in March, a missing timestamp occurs. For a similar-day approach using ANN, it’s vital that each day has precisely 24 hours.

I’ve found solutions to these problems. The preprocessing data includes several steps of filtering out invalid data, like zero or NaN values. Additionally, resampling happens on the date and time column, and missing values are interpolated. On the terminal ends of the columns, at the very beginning and the end, no interpolation should be applied.

In some cases, the missing data problem was more prominent and could not be fixed with interpolation. E.g., two weeks were lost entirely in case of Estonia consumption from November 22, 2016, until December 7, 2016. The possible solutions included the following variants:

- not to add the first Alexela dataset until December 7, 2016, altogether
- leave the 2-week gap in Estonia consumption history, which would effectively split the first Alexela set into two independent sets
- copy day-wise consumption history from the previous 2015 year

It was not possible to include a forecasted consumption instead of real consumption, because it was also missing. Therefore, I've chosen the third variant because having all the data possible is crucial for ANN. The data fitted ideally because the profile of consumption was in these two years very close to each other.

5.9 Architecture of the application

The input for ANN is the data or input vectors, as shown in Figure 19. The raw data is stored locally in the form of CSV or Excel files. The updated files for Alexela electricity consumption are processed when placed in a particular folder. Other data, like electricity consumption of Estonia, weather history, and forecast, are downloaded from the Internet by user request. Some of the data, like timestamps and daylength, get generated by the program code.

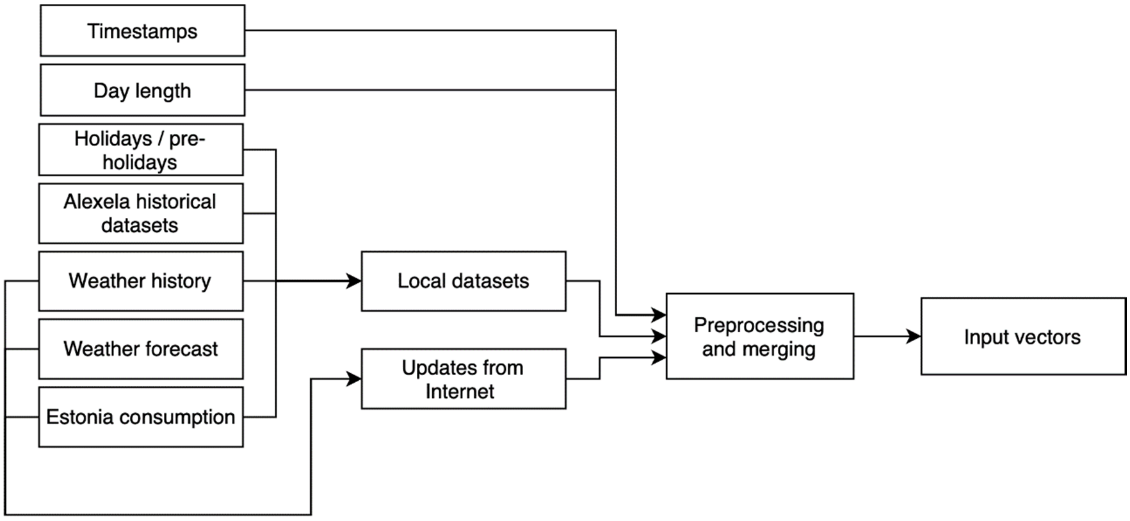


Figure 19. Combining data from different local and online sources and preprocessing.

All the data should be preprocessed, e.g., some columns should be shifted by 168 hours forward like temperature the same day and hour last week, some should be converted to sine and cosine representation. Noteworthy is also merging the data so that each day would have only 24 hours. There should be no duplicates, no missing or wrong values.

Before the input can be presented to ANN, some more steps should be accomplished, as shown in Figure 20. From this large table, which contains all possible columns processed from different sources, it's possible to select only the columns needed by a specific input vector. For different stages, I used different splitting techniques. E.g., while researching the capabilities, I used the classical three set approach – train, validation, and test approach. For a final application, there's no more need for validation and test sets. Next, the rescaling can be applied on specific columns.

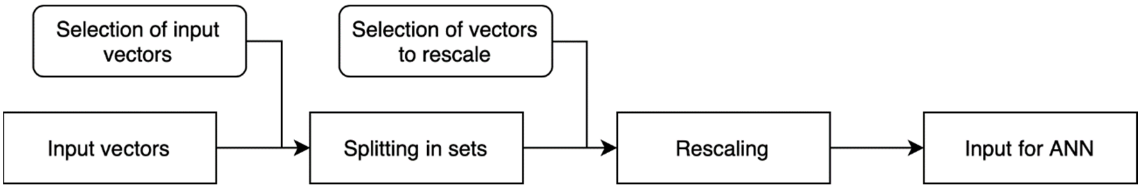


Figure 20. Preparation of the input for ANN includes selecting proper columns, splitting into sets, and rescaling.

The application was designed by preserving the Model-View-Controller (MVC) architecture, as shown in Figure 21. The benefit of using MVC structure is that each its component can easily be replaced without the need to rewrite the other two.

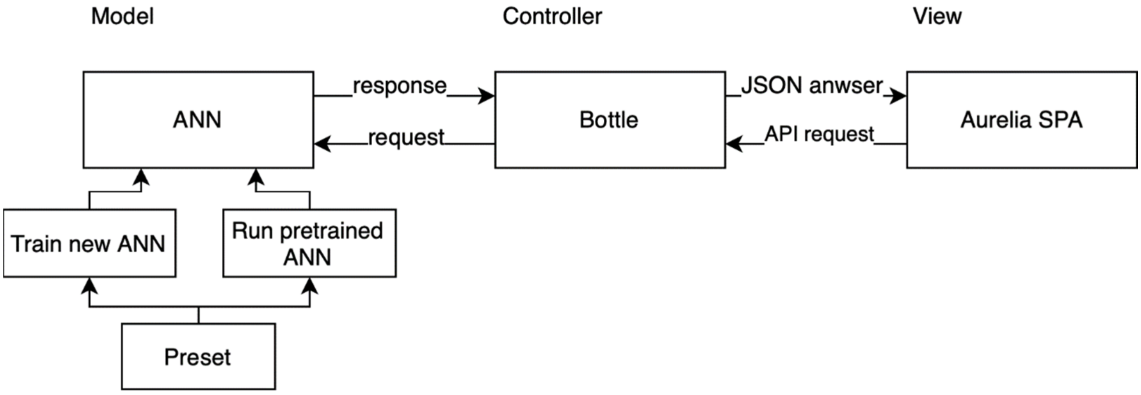


Figure 21. The architecture of an application is following an MVC structure.

The model is ANN with all the business logic. Two main functions are supported: the new model can be trained, or a saved version or pre-trained model can be executed. Different models, e.g., with various input and rescaled vectors, can be created using presets. For each preset, four files have been previously saved: Keras h5 file for recreating the ANN, scaler file, lists of input columns and columns to rescale. Those files were used differently, according to the user's choice to create a new ANN or run the pre-trained model. The controller is a Bottle web framework which offers server capabilities, coordinates the communication between the model and view. The view is created using SPA Aurelia. It represents the UI, which the user sees and can operate with.

5.10 Further development

Some forecasts were more off than others, and the manual investigation didn't always reveal the exact cause for some deviations. Nevertheless, it's apparent, that also other factors can influence the electricity consumptions besides the ones, I've tried. By finding the exact cause for those unexpected fluctuations, it's possible to make the model even more precise. It could be accomplished by, e.g., adding some extra post-processing step for further correcting the output of an ANN.

By combining different models, it's also possible to improve the precision – called model ensemble. During the search for an input vector, I've tried all possible combination of 6 inputs. I've selected the best models and decided to combine those results by calculating the average of results. However, the test was not able to reveal the more exact result than provided by the single model.

A better approach would be to combine models, which were built using different techniques – a hybrid, described in 2.3. Every single method has its method-specific limitations. For example, ANN tends to stick with a local minimum and miss the global one. Even a more advanced technique exists, which involves decomposing the datasets into several parts. The specific model will process each of those portions. Afterward, the results will be combined.

5.11 Conclusions

- ANN behaved differently than the linear regression model, e.g., fine-tuning of temperatures like weighted temperature or degree-hour approach showed degradation in performance.

- The amount of data is crucial for the success of ANN; one year of information is not enough to teach the ANN.

- Besides temperature, the electricity consumption of Estonia and day of the year were potent attributes.

- Not only the input vectors but their representation way for ANN are essential, e.g., instead of using absolute values, a percentage difference can be significantly more beneficial.

- Rescaling applied to all columns showed degradation in performance. On some of the columns, e.g., temperature, it may be helpful not to use a rescaling technique, although some loss of stability may be observed, e.g., in case of an inaccurate weather forecast.

- Alexela Energy new datasets should not be compared with each other, because the number of clients can vary. Instead, only within a dataset relative differences in electricity consumption compared to the previous week should be calculated.

- Hyperparameter optimization used for tuning the structure of ANN can not only improve the performance but can also considerably speed up the training procedure by lowering the epoch number needed and improve convergence.

- For hyperparameter optimization, it was beneficial to use a number of neurons equal to the factor of two, which allowed easier to find the structure with decent performance.

- To further improve the performance of ANN, additional vectors with different extra information are needed, or modification should be applied to same-day approach.

- From research papers, it’s known, that a combination of different techniques can further improve the performance – a hybrid approach.

6 Summary

The STLF is a topic, which is intensively researched since the year of 1966. By looking back in history, it all started by applying the statistical time-series methods and then moved to the AI methods. An ANN, a subgroup of AI, has a central role in building the forecasting models, because of its capabilities of creating complex non-linear models and possibility to learn on always growing sets of data.

Predicting the electricity consumption is crucial for electricity suppliers like Alexela Energy because the ordering of the electricity needed for the next day happens in advance. Therefore, it’s critical to have the forecast per hour with the highest precision possible. Misplanned order can lead not only to wasted resources but also to economic loss and need to order additional electricity on short notice through higher-priced channels.

In my thesis, I’ve lowered the average forecast error to 3.62% from ca. 4.50%. This improvement was achieved by research, experiments, and results’ evaluation. The first breakthrough brought a hyperparameter optimization, which gave a suitable ANN structure fitted for the assignment. The second improvement was achieved by an enhanced input layer, which included besides temperature also day duration, day coefficient, Estonia electricity consumption, and holiday information in the most beneficial representation way. A generalization test was conducted to make sure the results are stable.

To allow convenient use the ANN, I’ve created a forecasting application with GUI which offers running pre-trained models as well as training new ones. Alexela Energy can add new datasets which will be merged with other local datasets and updated weather history, weather forecast and electricity consumption of Estonia datasets from online API-s. For better evaluation of results, a visualization of the predictions within the GUI and download function for forecasts was implemented.

References

[1] "Choosing an electricity supplier | Eesti.ee," *Estonian government information portal | Eesti.ee*. [Online]. Available: https://www.eesti.ee/en/housing-and-environment/services-related-to-housing/choosing-an-electricity-supplier/. [Accessed: 08-Apr-2019].

[2] "Electricity market - Eesti Energia." [Online]. Available: https://www.energia.ee/en/elekter/elektriturg. [Accessed: 08-Apr-2019].

[3] R. Weron, "Electricity price forecasting: A review of the state-of-the-art with a look into the future," *Int. J. Forecast.*, vol. 30, no. 4, pp. 1030–1081, Oct. 2014.

[4] M. Spitšakova, J. Belikov, and E. Petlenkov, "Alexela Energy AS applied research for development of electricity consumption prediction model," LEP18083, 2019.

[5] A. Yang, W. Li, and X. Yang, "Short-term electricity load forecasting based on feature selection and Least Squares Support Vector Machines," *Knowl.-Based Syst.*, vol. 163, pp. 159–173, Jan. 2019.

[6] J. Zhang, Y.-M. Wei, D. Li, Z. Tan, and J. Zhou, "Short term electricity load forecasting using a hybrid model," *Energy*, vol. 158, pp. 774–781, Sep. 2018.

[7] L. Sullivan, "Correlation and Linear Regression." [Online]. Available: http://sphweb.bumc.bu.edu/otlt/MPH-Modules/BS/BS704_Correlation-Regression/BS704_Correlation-Regression_print.html. [Accessed: 06-Apr-2019].

[8] B. Fortuner, "Linear Regression — ML Cheatsheet documentation." [Online]. Available: https://ml-cheatsheet.readthedocs.io/en/latest/linear_regression.html. [Accessed: 06-Apr-2019].

[9] H. Lohninger, "Linear vs. Nonlinear Models," *Fundamentals of Statistics*. [Online]. Available: http://www.statistics4u.com/fundstat_eng/cc_linvsnonlin.html. [Accessed: 06-Apr-2019].

[10] G. Heinemann, D. Nordmian, and E. Plant, "The Relationship Between Summer Weather and Summer Loads - A Regression Analysis," *IEEE Trans. Power Appar. Syst.*, vol. PAS-85, no. 11, pp. 1144–1154, Nov. 1966.

[11] E. y Stellwagen and L. Tashman, "ARIMA : The Models of Box and Jenkins," *Foresight Int. J. Appl. Forecast.*, no. 30, pp. 28–34, 2013.

[12] F. Malik, "Understanding Auto Regressive Moving Average Model — ARIMA," *Medium*, 19-Sep-2018. [Online]. Available: https://medium.com/fintechexplained/understanding-auto-regressive-model-arima-4bd463b7a1bb. [Accessed: 06-Apr-2019].

[13] J. Brownlee, "How to Create an ARIMA Model for Time Series Forecasting in Python," *Machine Learning Mastery*, 08-Jan-2017. [Online]. Available: https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/. [Accessed: 06-Apr-2019].

[14] J. Brownlee, "A Gentle Introduction to SARIMA for Time Series Forecasting in Python," *Machine Learning Mastery*, 16-Aug-2018. [Online]. Available: https://machinelearningmastery.com/sarima-for-time-series-forecasting-in-python/. [Accessed: 06-Apr-2019].

[15] A. Savelainen, “Construction of SARIMAX- models using MATLAB,” p. 20, 2009.

[16] S. Glen, “Exponential Smoothing: Definition of Simple, Double and Triple,” *Statistics How To*, 07-Jun-2018. [Online]. Available: https://www.statisticshowto.datasciencecentral.com/exponential-smoothing/. [Accessed: 06-Apr-2019].

[17] H. M. Al-Hamadi and S. A. Soliman, “Short-term electric load forecasting based on Kalman filtering algorithm with moving window weather and load model,” *Electr. Power Syst. Res.*, vol. 68, no. 1, pp. 47–59, Jan. 2004.

[18] P. Nakamoto, *Neural Networks and Deep Learning: Neural Networks and Deep Learning, Deep Learning, Big Data*. CreateSpace Independent Publishing Platform, 2018.

[19] J. Brownlee, “Difference Between Classification and Regression in Machine Learning,” *Machine Learning Mastery*, 10-Dec-2017. [Online]. Available: https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/. [Accessed: 14-Apr-2019].

[20] A. S. Walia, “Activation functions and it’s types-Which is better?,” *Towards Data Science*, 29-May-2017. [Online]. Available: https://towardsdatascience.com/activation-functions-and-its-types-which-is-better-a9a5310cc8f. [Accessed: 16-Apr-2019].

[21] B. Fortuner, “Activation Functions — ML Cheatsheet documentation.” [Online]. Available: https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html#id6. [Accessed: 16-Apr-2019].

[22] A. Karpathy, “CS231n Convolutional Neural Networks for Visual Recognition.” [Online]. Available: http://cs231n.github.io/neural-networks-1/. [Accessed: 16-Apr-2019].

[23] B. Fortuner, “Gradient Descent — ML Cheatsheet documentation.” [Online]. Available: https://ml-cheatsheet.readthedocs.io/en/latest/gradient_descent.html. [Accessed: 17-Apr-2019].

[24] S. Bouktif, A. Fiaz, A. Ouni, and M. Serhani, “Optimal Deep Learning LSTM Model for Electric Load Forecasting using Feature Selection and Genetic Algorithm: Comparison with Machine Learning Approaches †,” *Energies*, vol. 11, no. 7, p. 1636, Jun. 2018.

[25] J. Brownlee, “How to Reshape Input Data for Long Short-Term Memory Networks in Keras,” *Machine Learning Mastery*, 29-Aug-2017. [Online]. Available: https://machinelearningmastery.com/reshape-input-data-long-short-term-memory-networks-keras/. [Accessed: 06-Apr-2019].

[26] A. Deihimi and H. Showkati, “Application of echo state networks in short-term electric load forecasting,” *Energy*, vol. 39, no. 1, pp. 327–340, Mar. 2012.

[27] N. Bambrick, “Support Vector Machines for dummies; A Simple Explanation,” *AYLIEN*, 24-Jun-2016. [Online]. Available: http://blog.aylien.com/support-vector-machines-for-dummies-a-simple/. [Accessed: 06-Apr-2019].

[28] S. Ray, “Understanding Support Vector Machine algorithm from examples (along with code),” *Analytics Vidhya*, 12-Sep-2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/. [Accessed: 06-Apr-2019].

[29] M. Kanevski, A. Pozdnukhov, S. Canu, M. Maignan, P. M. Wong, and S. A. R. Shibli, “Support Vector Machines for Classification and Mapping of Reservoir Data,” in *Soft Computing for Reservoir Characterization and Modeling*, vol. 80, P.

Wong, F. Aminzadeh, and M. Nikravesh, Eds. Heidelberg: Physica-Verlag HD, 2002, pp. 531–558.

[30] J. A. K. Suykens and J. Vandewalle, “Least Squares Support Vector Machine Classifiers,” p. 8.

[31] H. J. Sadaei, P. C. de Lima e Silva, F. G. Guimarães, and M. H. Lee, “Short-term load forecasting by using a combined method of convolutional neural networks and fuzzy time series,” *Energy*, vol. 175, pp. 365–377, May 2019.

[32] “Hectopascals to millimeters of mercury [hPa to mmHg] pressure conversion.” [Online]. Available: https://www.aqua-calc.com/convert/pressure/hectopascal-to-millimeter-of-mercury. [Accessed: 13-Apr-2019].

[33] I. London, “Encoding cyclical continuous features - 24-hour time,” *Ian London’s Blog*, 2016. [Online]. Available: http://ianlondon.github.io/blog/encoding-cyclical-features-24hour-time/. [Accessed: 17-Apr-2019].

[34] V. Ravichandran, “MATLAB Neural Network Toolbox Workflow,” 11-Apr-2018. [Online]. Available: https://www.youtube.com/watch?v=tc36mc1xcEY. [Accessed: 18-May-2019].

[35] A. S. Walia, “Types of Optimization Algorithms used in Neural Networks and Ways to Optimize Gradient Descent,” *Towards Data Science*, 10-Jun-2017. [Online]. Available: https://towardsdatascience.com/types-of-optimization-algorithms-used-in-neural-networks-and-ways-to-optimize-gradient-95ae5d39529f. [Accessed: 19-Apr-2019].

[36] “FAQ - Keras Documentation.” [Online]. Available: https://keras.io/getting-started/faq/. [Accessed: 19-Apr-2019].

[37] W. C. Forsythe, E. J. Rykiel, R. S. Stahl, H. Wu, and R. M. Schoolfield, “A model comparison for daylength as a function of latitude and day of year,” *Ecol. Model.*, vol. 80, no. 1, pp. 87–95, Jun. 1995.

[38] “Degree Days - Handle with Care!” [Online]. Available: https://www.energylens.com/articles/degree-days. [Accessed: 20-Apr-2019].

[39] J. Brownlee, “How to Normalize and Standardize Your Machine Learning Data in Weka,” *Machine Learning Mastery*, 04-Jul-2016. [Online]. Available: https://machinelearningmastery.com/normalize-standardize-machine-learning-data-weka/. [Accessed: 20-Apr-2019].

[40] “machine learning - StandardScaler before and after splitting data,” *Data Science Stack Exchange*. [Online]. Available: https://datascience.stackexchange.com/questions/38395/standardscaler-before-and-after-splitting-data. [Accessed: 20-Apr-2019].

[41] R. Robbins and C. + C. of U. W. Strategies, “How Accurate Are Weather Forecasts?,” *HuffPost*, 10:47-500. [Online]. Available: https://www.huffpost.com/entry/how-accurate-are-weather-_b_6558770. [Accessed: 20-Apr-2019].

[42] A. Gozzoli, “Practical guide to hyperparameter searching in Deep Learning,” *FloydHub Blog*, 05-Sep-2018. [Online]. Available: https://blog.floydhub.com/guide-to-hyperparameters-search-for-deep-learning-models/. [Accessed: 21-Apr-2019].

[43] M. Bessec and J. Fouquau, “Short-run electricity load forecasting with combinations of stationary wavelet transforms,” *Eur. J. Oper. Res.*, vol. 264, no. 1, pp. 149–164, Jan. 2018.

[44] P. Heckert, “Why Does Humidity Make You Feel Colder In Lower Temperatures?,” *Decoded Science*, 30-Nov-2011. [Online]. Available:

https://www.decodedscience.org/why-does-damp-cool-weather-make-it-feel-colder/6736. [Accessed: 13-Apr-2019].

[45] G. Oğcu, O. F. Demirel, and S. Zaim, “Forecasting Electricity Consumption with Neural Networks and Support Vector Regression,” *Procedia - Soc. Behav. Sci.*, vol. 58, pp. 1576–1585, Oct. 2012.