Lappeenranta University of Technology
School of Engineering Science
Computational Engineering and Technical Physics
Computer Vision and Pattern Recognition

Aleksandr Lukoshkin

FOREST STAND PARAMETER ESTIMATION BY USING NEURAL NETWORKS.

Master's Thesis

Examiners: Associate Professor Arto Kaarna
Associate Professor Olga Gorbaneva

Supervisors: Associate Professor Arto Kaarna
Associate Professor Virpi Junttila
Assistant Professor Ilya Loshkarev

ABSTRACT

Lappeenranta University of Technology
School of Engineering Science
Computational Engineering and Technical Physics
Computer Vision and Pattern Recognition

Aleksandr Lukoshkin

Forest stand parameter estimation by using neural networks.

Master's Thesis

2019

55 pages, 18 figures, 17 table, 1 appendix.

Examiners: Associate Professor Arto Kaarna
Associate Professor Olga Gorbaneva

Keywords: neural network, airborne laser scanning, forest inventory, species-specific estimates

Wood is a vital material used everywhere. And it is very important to monitor the state of forests. One of the most common ways to estimate different forest parameters is to make the model based on the data obtained by airborne light detection and ranging (LiDAR). But, for building an accurate model on large areas, it is necessary to use hundreds of expensive and time-consuming field sample measurements. Besides that, species-specific estimations require a deeper study of the issue and additional measurements. All these data forms poorly structured sets with a high correlation between independent variables. That makes a problem to build a model with an acceptable level of error for using it in practice. Recent studies in the machine learning area illustrate that Artificial Neural Networks(ANN) are good at modeling complex relationships between data. This thesis provides an analysis of the possible use of neural network algorithms for estimating forest stand parameters on Finnish forest sites for both general and species-specific analysis.

PREFACE

This thesis has been written as a part of a double degree program with the collaboration of Lappeenranta University of Technology and Southern Federal University. I would like to thank my supervisors Arto Kaarna and Virpi Junttila for their patient and guidance of my research work and Ilya Loshkarev, Olga Gorbaneva, and Konstantin Nadolin for helping with the Southern Federal University matters. I would like to express my sincere gratitude to all my friends and family for always supporting me. Thank you for always believing in me.

Lappeenranta, December 13, 2019

Aleksandr Lukoshkin

CONTENTS

LIST OF ABBREVIATIONS

2-D	Two-dimensional
3-D	Three-dimensional
ALS	Airborne Laser Scanning
ANN	Artificial Neural Network
BLUP	Best Linear Unbiased Predictor
Cnf	Coniferous trees
CNN	Convolutional Neural Network
FCN	Fully Convolutional Networks
GPR	Gaussian Process Regression
GRNN	Generalized Regression Neural Network
Hwd	Hardwood
IoU	Intersection over Union
k-NN	k-Nearest Neighbor
k-MSN	k-Most Similar Neighbor
LDA	Linear Discriminant Analysis
LiDAR	Light Detection And Ranging
LME	Linear Mixed-Effects
LR	Linear Regression
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MLP	Multilayer Perceptron
MSE	Mean Square Error
MSN	Most Similar Neighbour
NN	Neural Network
OA	Overall Accuracy
PCA	Principal Component Analysis
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
RF	Random Forest
RMSE	Root-Mean-Square Error
SD	Standard Deviation
SOM	Self-Organizing Map
SVD	Singular Value Decomposition
SVM	Support Vector Machine
SVR	Support Vector Regression
TLS	Terrestrial Laser Scanning
UQ	Uncertainty Quantification

1 INTRODUCTION

1.1 Background

Wooden production takes a big part in economics, manufacture, in our everyday life. Deforestation is our goldmine of opportunities and the sword of Damocles at the same time. On the one hand, wood is a biomaterial. Products from wood, paper, cardboard in many areas can successfully replace harmful, hardly degradable items from usage, e.g. plastic cups. On the other hand, this harms the ecology of the planet, decreases oxygen production, damages the forest ecosystem, etc. Unfortunately, forests are hard to replenish resources. Therefore, using modern technology, it is possible to monitor the parameters of forests and to plan deforestation with the least damage to the environment.

Forest inventory management is a dynamic system integrating chains of wood production, marketing, and conservation. For the analysis of forest resources, modern types of measurements and assessments are used. It is very important to keep the bio balance of the forests while using their resources. Thus, the important task is the optimization of measurements of forest stand parameters.

In compartment-based forest inventory, the characteristics of forest areas, obtained by field measurements, are stored as values of individual parameters, e.g. stem number, basal area, and volume. These parameters could be separated as total and species-specific characteristics. The estimation of these parameters is intended for research and inconvenient for field measurements.

Field measurements of forest inventory variables are very expensive and time-consuming. It has been shown that estimating the parameters of forest stands based on light detection and ranging at the compartment level gives good results, especially in the case of total parameters of forest stands. These estimates were obtained using various mathematical approaches, such as the usual least-squares regression with stepwise selection of variables [1], [2], k-neighbor methods such as k-nearest neighbors or k-most similar neighbors with stepwise variable selection [3], and Bayesian regression with automatic variable selection [4]. These methods used LiDAR data and field measurements of stand parameters to train the models. Then, the forest stand parameters of the target areas were estimated by the trained model using LiDAR measurements of the target areas as input data.

However, the LiDAR data is not strongly correlated with the parameters of species-specific forest stands. Therefore, features derived from digital aerial photographs are also added to the model. The precision of estimates for specific species is generally much lower than the precision of total parameter estimates. The main reason for this is that LiDAR primarily measures the height of trees. The differences by species appear only in the statistically low-significant features of the LiDAR histogram and in the information of the color of aerial photographs.

Due to the increased power of computing devices, there is a high interest in using machine learning algorithms in various applications. Every year their complexity, the level of automation, abstraction, and accuracy of calculations are growing. One of the most famous algorithms is the artificial neural network. Based on the mathematical representation of biological neurons, these algorithms are able to automatically learn the subtle data relationships; various types and implementations of the ANNs are capable of processing any data representation.

This work is aimed at studying the possibility of using algorithms of artificial neural networks to calculate the forest stand parameters based on the LiDAR measurements and features derived from digital aerial photographs.

1.2 Objectives and delimitations

This thesis is focused on the estimation of the forest stand parameters by using the LiDAR data plots and features derived from digital aerial photographs. The aimed variables are total and species-specific wood volume. Artificial neural networks are used as a computational model. The calculation of the values of the desired parameters is a regression task. The results are compared with a linear model to recognize the necessity of using these methods.

The specific objectives of the project are as follows:

1. Building a neural network to solve the regression problem.
2. Selection and comparison of architectures of the ANN.
3. Analysis of the possible use of various datasets.
4. Analysis of the effect of reducing the dimensionality of input data.

5. Comparison of results with previous studies.

1.3 Structure of the thesis

This work is aimed at studying the use of neural networks to estimate the forest standard parameters by using LiDAR data and aerial photographs. Chapter 2 provides a review of previous studies. Different implementations of laser scanning data and different techniques of estimation are reviewed. An introduction to the artificial neural network is provided in chapter 3. It contains an overview of two types of neural networks(NN): Multilayer Perceptron (MLP) and Generalized Regression Neural Network (GRNN). Also, some practical applications for regression tasks with these models are reviewed. The fourth chapter includes the description of implemented algorithms for fully-connected NN (MLP models), generalized regression neural network and Linear Regression (LR) model, which was added in order to compare the results of ANNs. Chapter 5 describes the dataset, used metrics, experiments, and their results. At chapter 6 benefits, drawbacks, and possible improvements of the algorithms are reviewed. Chapter 7 includes the conclusion.

2 Related Work

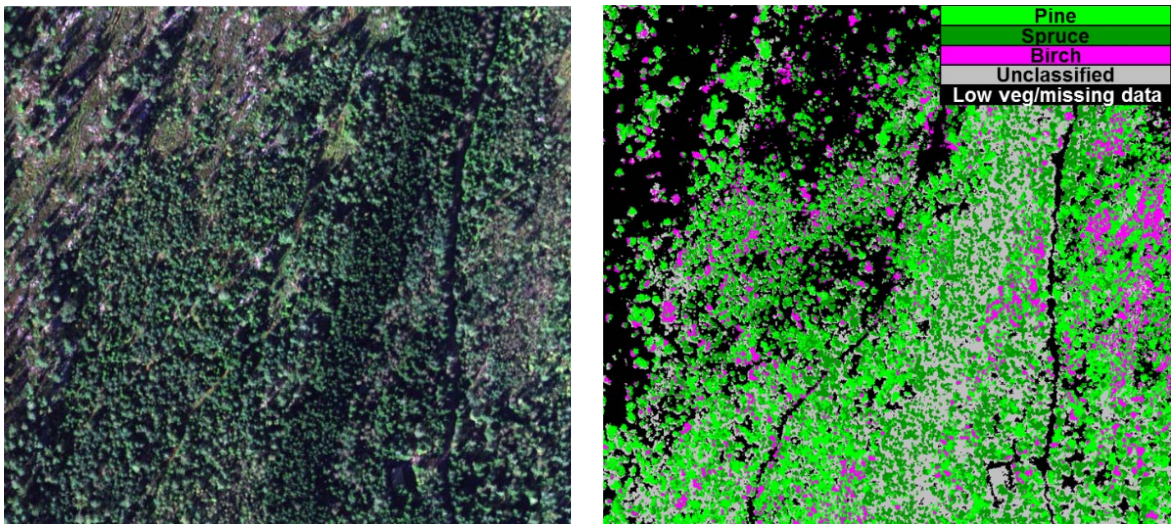
This chapter provides information about previous studies and approaches. Subsection 2.1 describes the species classification tasks in and subsection 2.2 illustrates techniques for solving regression tasks for forest stand parameter estimations.

2.1 Species classification in forest inventory

The processing of forest parameter data includes different types of tasks. One of them is the classification of tree species based on the laser scanning data. The analysis of the classification task could help realize the species-specific estimation. This section considers some of the previous solutions.

At [5] the task was to classify tree species using Airborne Laser Scanning(ALS). For this classification task such several methods were implemented and compared: linear discriminant analysis (LDA), k-nearest neighbor (k-NN), k-most similar neighbor (k-MSN), and random forest (RF). In their study, authors used an extensive dataset consisting of 13 890 trees in 118 plots and two LiDAR campaigns to investigate the problem of classification different tree species in Finland forests by using two different LiDAR scanners and specialized approaches for data normalization. The achieved accuracy was 88-90% in the classification of Scots pine, Norway spruce, and birch. The authors normalized for each scanner dataset using their specific parameters.

Article of Rudjord et.al. [6] illustrates a method for the classification of spruce, pine, and birch in Norwegian forests. For this purpose, an airborne laser scanning and hyperspectral data were used. The ALS data provides information about vegetation height, while the hyperspectral data contains information about biophysical and biochemical parameters and species composition. The authors built a decision tree to classify the dataset into three different tree species categories based on the spruce index and the conifer index, obtained from the color information from spectral data. The classification accuracies for all three classes are in the range of 83-86%. Fig. 1(a) illustrates the color representation of a portion of the hyperspectral data, fig:RudjordImg(b) shows the results of its classification.



(a) (b)

Figure 1. (a) Color representation; (b) Classification results [6]

Zou et.al. [7] presented a deep learning-based model for the classification of different tree species. In their study, authors used a Complex Forest 3-D Point Clouds obtained from a Terrestrial Laser Scanning (TLS). Unlike ALS, TLS systems provide richer information, but also spend more time and money. The whole classification method was derived in three steps. At the first stage, each tree was extracted based on the center density, followed by data preprocessing including noise removal. At the second stage, a three-dimensional point cloud was projected onto a two-dimensional plane, which contains the contours of the trees. To increase the number of train samples, the projection was repeated by turning for every 10 degrees of 3-D object. Then the Deep Belief Network is used for classification. The achieved average accuracies were 93.1% and 95.6% for two data sets.

2.2 Regression tasks in forest inventory

Regression task in the forest inventory could be implemented to estimate such parameters as forest biomass, stand volume, tree height, tree trunk diameter, etc. The works described below used the data of airborne laser scanning to calculate the necessary parameters.

This master thesis is aimed at solving the regression problem for estimating the forest stand parameters. Previous studies of the same problem implemented the following mathematical approaches: least-squares regression with stepwise variable selection

[1, 2], k-neighbor methods (k-nearest neighbors or k-most similar neighbors) with stepwise variable selection [3], and Bayesian regression with automatic variable selection [4]. The samples of LiDAR scanned areas with corresponding field measurement-based forest stand parameters were used for training the models, then forest stand parameters of the target area were estimated by using these models and LiDAR scans as an input. Usually, the more data these models receive for training the better results it will show.

In [8], Nothdurft et.al. applied a non-parametric most similar neighbor approach. The studied area was the municipal forest of Waldkirch, 13 km north-east of Freiburg, Germany. In this approach, the same type of inventory plots and laser scanner data were used for predicting the stand parameters. The neighbor distances are expressed by the similarity between auxiliary variables - laser scanning data - in a forest stand and those observed on the sample plot. The authors calibrated the prior predictions by means of sample plot data within the forest stands and then made the global bias corrections via best linear unbiased predictors (BLUPs). Besides that, for each forest stand, estimations of the target variables were separated for each tree species. The mean relative error (that was calculated as ratio of the half 90% confidence-interval range to the mean estimate) for total volume per ha achieved by the pure most similar neighbor (MSN) prediction averages 18.7%, and is reduced to 16.6% by the calibration.

Silva et.al. [9] presented a Random Forest model, a supervised machine learning algorithm, which allowed to estimate total, commercial, and pulp volumes at industrial forests in southern Brazil. Based on ALS data, they found that the model, based only on the height of the top of the canopy and the skewness of the vertical distribution of LiDAR points, has a very strong and unbiased predictive power. In this work, the authors made a fifth-degree polynomial model of total and assortment volumes based on on-site measurements of individual tree height and diameter at breast height. Then they selected the best lidar metrics for modeling stem volumes based on Pearson's correlation between variables and Model Improvement Ratio, a standardized measure of variable importance. And then built a RF model based on chosen LiDAR variables and compare these two models. The achieved Root-Mean-Square Error (RMSE) values are 7.83%, 7.71% and 8.63% for predictions of the total, commercial, and pulp volume respectively.

At [10], Varvia et.al. describe the estimation of species-specific stand attributes like tree height, stem diameter, stem number, basal area, and stem volume and uncertainty quantification (UQ) by using Gaussian process regression (GPR), which is a nonlinear and nonparametric machine learning method. The authors illustrate the performance advantage over kNN even when smaller training sets are used. For training and estimations, the

authors used the ALS data and data from aerial images. From the aerial images, the mean values of each color channel were used along with two spectral vegetation indices. The results were compared with kNN and Bayesian linear regression. As it is shown below (Fig. 2), the GPR improved the results of species-specific and total volume estimations.

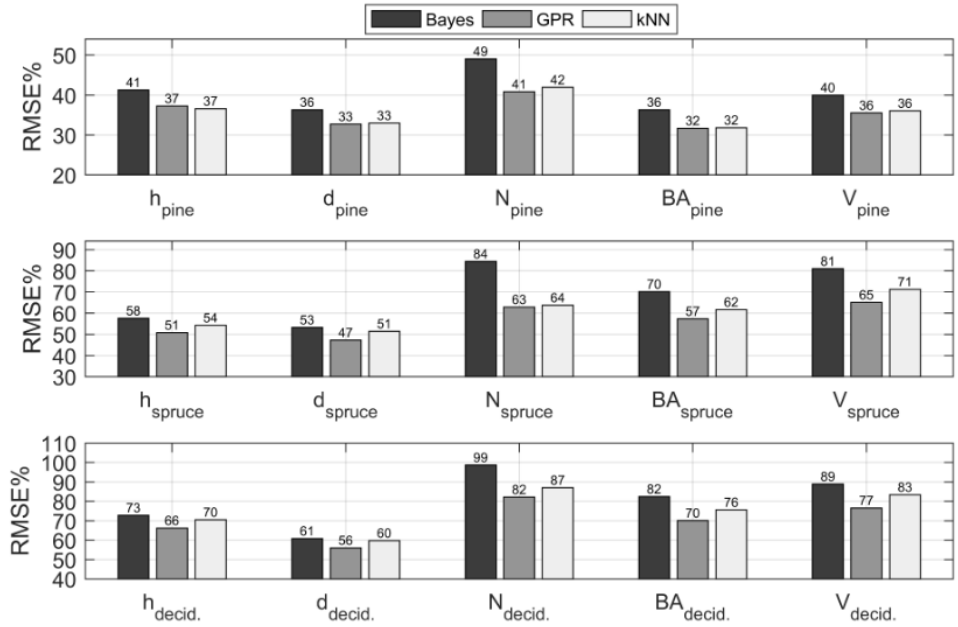


Figure 2. Relative RMSE for pine (top), spruce (middle) and deciduous (bottom) [10].

Gleason et.al. [11] compared methods such as linear mixed-effects (LME) regression, random forest, support vector regression (SVR), and Cubist. They estimated biomass in moderately dense forest at both tree and plot levels. As a result, SVR produced the most accurate biomass model. Behind that, at this work the method of delineation of tree crowns (that was previously developed and illustrated by the authors at [12]) was implemented. As a result it was shown that LME has a RMSE% between 82% and 119%; for RF: 32.4% at a plot level, 70 - 95% at the tree level; for SVR RMSE% at a plot level is 13 - 18%, at the tree level: 68 - 82%; for Cubist it is between 31% and 34% at the plot level and 70 - 96% at the tree level.

Niska et.al. [13] provided three machine learning models: multilayer perceptron, support vector regression, and self-organizing map (SOM). At this work, the authors compared models with the nonparametric k-most similar neighbor method. The estimation was based on the LiDAR and aerial photograph data and a set of field-sample plots. The

authors implemented a multiobjective genetic algorithm to reduce the number of input parameters. In this study, MLP and SVR illustrated the best results for prediction: 41.73 and 40.24 RMSE% respectively.

Garcia et.al. [14] illustrated the comparison of different linear regression techniques with machine learning approaches such as ANNs, SVR, nearest neighbors, and RF. The results confirm that classic multiple linear regression is outperformed by machine learning techniques.

Junttila et.al. [4] provided explanations of the difference between using two different scanners and how it is possible to linearly convert both of LiDAR datasets to the same structure. Further data processing involves bringing the dataset to a normal distribution and using the Bayesian regression algorithm to estimate forest stand parameters. Further works include the species-specific estimations for the same forest stand parameters by using additional information obtained from digital aerial photography [15] and reducing of features by using singular value decomposition (SVD) [16].

3 Artificial Neural Networks

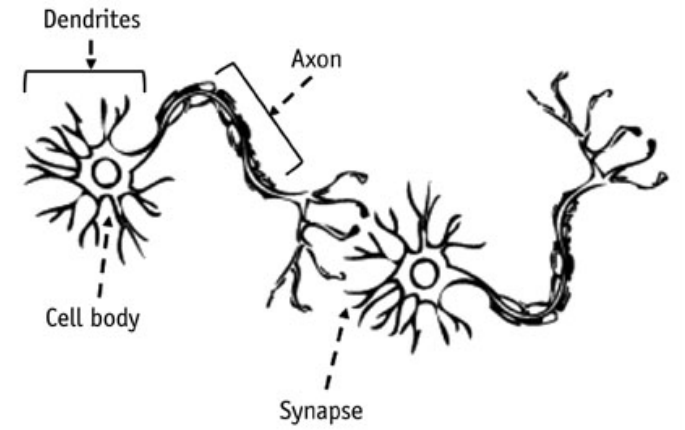
3.1 Background

Artificial neural networks have become more and more popular in different areas of our life. Despite the fact that they were invented back in the 1940s of the last century [17], a huge breakthrough in their development and application has occurred in recent decades. This is justified by an increase in computing power and an increment of the amount of data freely available.

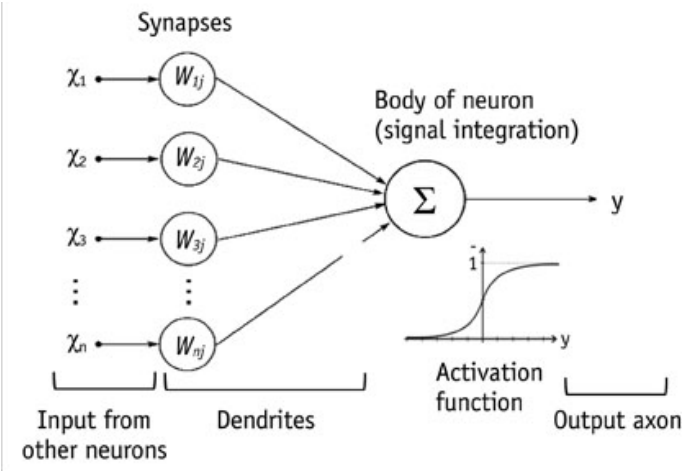
Artificial neural networks are computing systems inspired by biological neural networks of living beings (Fig. 3). ANN is based on a set of related units, called artificial neurons (analog of biological neurons in the biological brain). Each connection (synapse) between neurons can transmit a signal to another neuron. The receiving (postsynaptic) neuron can process the signal(s) and then signal the neurons connected to it. Neurons can have a state that is usually represented by real numbers, usually between 0 and 1. Neurons and synapses can also have a weight that changes during the learning process, which can increase or decrease the strength of the signal that it sends downstream.

An artificial neuron is a simple structural element that computes a linear combination of weights and inputs and applies activation function to the result. Using activation functions, neurons can introduce non-linearity to the function. Neurons are organized into sequential layers and output signals of neurons from one layer are passed to the next layer neurons. Fig. 4 illustrates these connections and relations.

ANNs learning is the process of calculating valid values of the weight matrix W. Usually, the backpropagation algorithm is used for this purpose. The main idea of this algorithm is to calculate the loss function and adjust the weight coefficients in "back" direction: from the outputs of the ANNs to the inputs. The loss function is the difference between the obtained and desired values. The loss function is usually minimized by the gradient descent method. The choice of the loss function (and minimization method) depends on the problem being solved using a neural network.



(a)



(b)

Figure 3. Example of: (a) Real neurons; (b) Artificial neurons. [18]

3.2 Multilayer Perceptron

For the regression task the most common and simple type of ANNs is Multilayer Perceptron. MLP is a simple ANN with N neurons in the input layer (here N is a number of X dimension), one neuron at the output layer with the linear activation function. Hidden layers could be modified for the task.

Olanrewaju et. al. [19] introduce the comparison between MLP and multiple linear regression for assessing performance of project selection. Referring to work of Amir Heydari et al [20] the authors used the fact that an upper limit for the number of neurons in the hidden layer should be smaller than 2N+1, where N is the number of input neurons (neurons from previous layer) in order to insure that neural networks are able to approximate any continuous function. As the metrics coefficient of determination R^2 and Mean Abso-

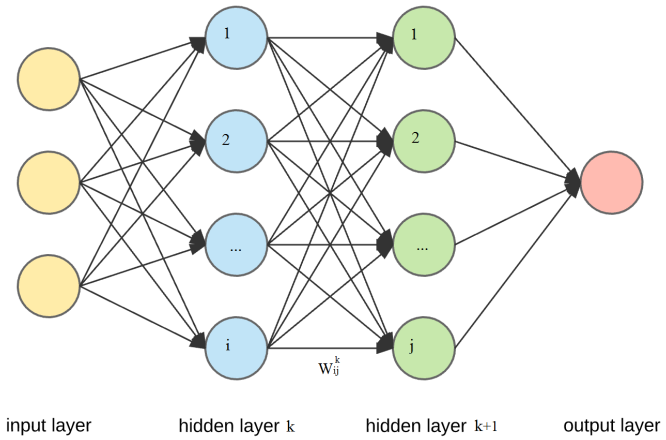


Figure 4. Artificial Neural Network scheme

lute Percentage Error (MAPE) were chosen in order to illustrate and compare the results. Multiple linear regression had coefficient of determination R^2 of 0.5136 with MAPE of 0.4, whereas neural network had MAPE of 0.237 and coefficient of determination R^2 of 0.755.

In [21] the authors implement the LR and ANN algorithms to illustrate the effect of the blade profile geometry parameter (e.g. hinged blade angle, blade length, and blade thickness, number of blade and foil shape) for a hinged blade cross axis turbine. As a result, the accuracy of Neural Network is higher by a small margin compare to the Linear Regression. There is partial difference of value at around an average of 11% between the two methods.

3.3 Generalized Regression Neural Network

At 1991 Donald Specht presented a new type of Neural Networks for regression problems [22]. GRNN is a memory based neural network that provides estimations of continuous variables and builds the sought function surface in a nonparametric fashion through the available data set. It has a highly parallel structure and uses only one iteration for the learning process. According to the author, this algorithmic form can be used for any regression problem in which an assumption of linearity is not justified.

GRNN has shown to be a useful method for regression problem and various modern implementations can prove it. For example, at [23] the GRNN models were reviewed and implemented as a part of more complex evaluation models for evaluating atmospheric

CO_2 effects.

In order to find an approximation, the general regression method use the next formula:

$$\hat{Y}(X) = \frac{\sum_{i=1}^{N}(Y_i \exp(D_i^2/2\sigma^2)}{\sum_{i=1}^{N}(\exp(D_i^2/2\sigma^2)} \tag{1}$$

where

$$D_i^2 = (X - X_i)^T(X - X_i) \tag{2}$$

is the distance between the training sample and the point of prediction. Here X_i is a training sample, X is a test sample, $\hat{Y}$ is the response value, Y_i is a train value. It is used as a measure of how well the each training sample can represent the position of prediction X. σ is the standard deviation or the smoothness parameter. So, here σ is the aim to a search. A bigger smoothness parameter allows to represent the prediction point, evaluated by the training sample, with a wider range of X. And for a small value of the smoothness parameter the representation of the evaluated point is limited to a narrow range of X, respectively. This equation allows to predict behavior of systems based on few training samples, approximate smooth multidimensional curves, and interpolate the function between the training points. Fig. 5 illustrates the differences of choosing different smoothness parameter. The example was taken from [24].

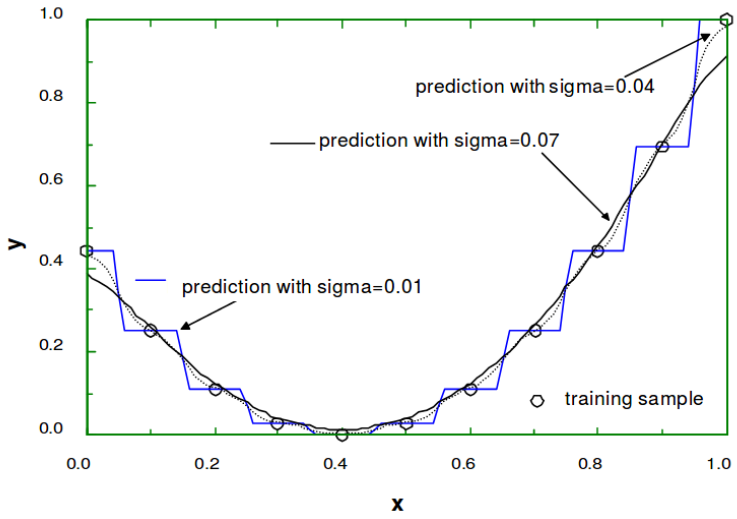


Figure 5. Comparison of different smoothness parameters [24]

Fig. 6 illustrates the structure and the architecture of the neural network. This NN has the next layers:

1. Input layer - one neuron for each variable X_i.
2. Hidden layer 1 - pattern units. The activations performed in each pattern neuron are $\exp(-D_i^2/2\sigma^2)$.
3. Hidden layer 2 - summation neurons. It consist of 2 neurons: Denominator and Numerator. The Denominator neuron collect the weighted (with the corresponding values of the training samples) signals of the pattern neurons. The weights of the signals going into the Numerator Neuron are equal to one.
4. Output layer calculates the output value. The value estimates by division of Numerator output signal by Denominator output signal.

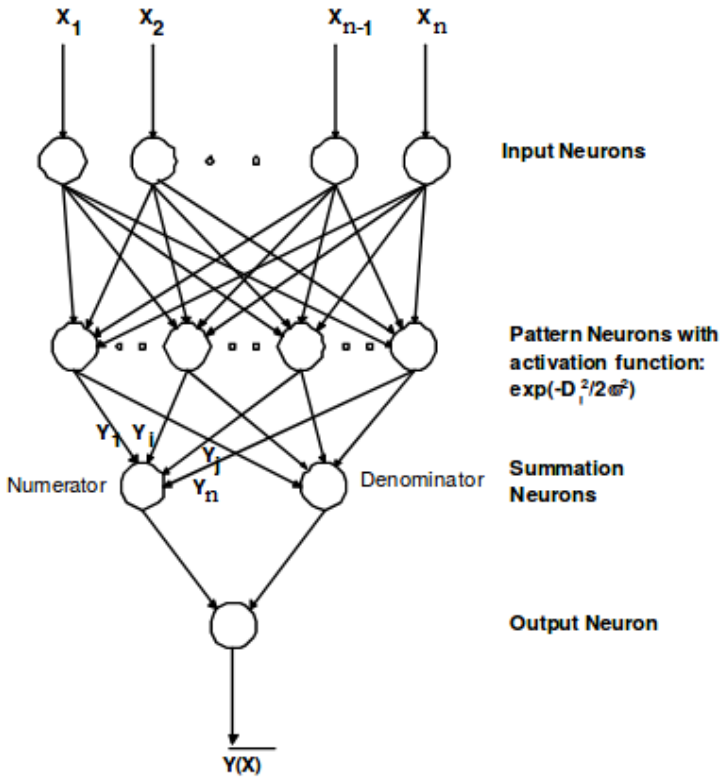


Figure 6. Generalized Regression Neural Network scheme [24]

Buliali et.al. [25] illustrated an application of the GRNN for traffic flow predictions and compares it with other predicting methods like ARIMA, Single Exponential Smoothing, and Moving Average. In this study, the authors used the Mean Absolute Percentage Error as the evaluation criterion. Comparing to other methods, GRNN showed two times better results.

In [26], the authors estimated aboveground biomass of Dacrydium pierrei and compared the results of using Support Vector Machine(SVM), ANN, GRNN and other algorithms. The modelling process was based on climate data. At this work the best results were shown by ANN and GRNN models.

The article of Gunawan et. al. [27] illustrates the estimation of food calories using the several features obtained from its digital images as an input like brightness, color, complexity of product, size, etc. This work shows that the GRNN models could have problems, when the input data does not represent the output values very well. Such kind of tasks requires a more complex data preprocessing or for example deep learning models.

4 PROPOSED METHODS

This chapter describes the list of methods that were implemented in this study. Linear regression was chosen for illustrating the baseline of existing methods. In this case, if another method shows worse results, it will be marked as unsuccessful for this task. Subchapter 4.2 describes the structure of ANN algorithms and subchapter 4.3 describes the structure of GRNN. Subchapter 4.4 describes the Principal Component Analysis (PCA) algorithm, that was implemented for feature selection.

This thesis is aimed to solve and analyze the regression problem. Regression analysis includes many methods of modeling and analysis of several variables when the goal is to find the relationship between the dependent variable Y and one or more independent variables X. Regression dependence can be determined as follows:

$$y(x_1, x_2, \dots x_n) = E(Y|X_1 = x_1, X_2 = x_2, \dots, X_n = x_n), \tag{3}$$

where $Y, X_1, X_2, \dots, X_n$ - set of variables, $x_1, x_2, \dots, x_n$ - set of values (samples), n - number of variables (size of dimension). Then the function $y(x_1, x_2, \dots, x_n)$ is called the regression function of the variable Y over the variables $X_1, X_2, \dots, X_n$, and its graph is a regression line of Y for the set $X_1, X_2, \dots, X_n$.

4.1 Linear Regression

Linear regression is the most common method for the regression task. In this method, the regression line is sought as a linear function

$$Y = b_0 + b_1 X_1 + b_2 X_2 + \dots + b_n X_n, \tag{4}$$

that best approximates the desired curve. Here $b_0, \dots, b_n$ are the regression parameters (coefficients). Usually, the method of least squares is implemented for solving this equation:

$$\sum_{k=1}^{M} (Y_k - \hat{Y_k})^2 \to min, \tag{5}$$

where M is the number of samples.

In this thesis, the linear regression model was built by using the Scikit-learn library for Python. For estimating the solution, the Algorithm 1 was proposed:

Algorithm 1 Linear Regression forecasting

Input: LiDAR measurements and digital aerial photograph attributes X, volume values Y

Output: Predicted values $\hat{Y}$, calculated errors.

1. Data preprocessing.
2. Start a cycle
3. Divide the datasets into two parts: training and testing (xtrain, xtest, ytrain, ytest)
4. Build and train a linear regression model with the training dataset
5. Calculate the predictions for the test dataset and compare it with test Y values. Estimate the error.
6. End of the cycle.
7. Repeat steps 2 - 6 for 100 times.

At this and further algorithms, the learning and estimation process is stored inside the loop. This loop is used because the division for train and test is realized with a random choice. To avoid the mistake when the error depends on the chosen samples, the statistical set of error estimations was chosen for a better assessment. The train set contains 90% of samples and the test set keeps 10% of samples. The number of loops is equal to 100 for all the algorithms. The output accuracy is calculated as the average of estimations for all 100 values for each metric.

4.2 MLP

Multilayer Perceptron was chosen as the first ANN model. This model requires the preliminary definition of architecture and hyperparameters such as the number of hidden layers, number of neurons on hidden layers, activation functions, optimization algorithm, batch size, etc. Besides that, MLP requires a number of epochs. To fit the weights of neurons, the model comes through the set several times. Here, Algorithm 2 describes the learning process of MLP:

Algorithm 2 MLP Learning Process

Input: independent variables X, dependent variables Y

Output: Predicted values $\hat{Y}$, calculated errors, prediction model.

1. Initialization of the MLP model
2. Divide the datasets on train set and validation set
3. Data sample X_i goes through the MLP and gives the predicted result of $\hat{Y}_i$.
4. Estimation of the ith error, sum up the $batchsize$ numbers of errors.
5. Change the weights of the NN by backpropagation algorithm to minimize the calculated error
6. One epoch: repeat the steps 2 - 5 for the whole dataset
7. Calculate the errors on the validation subset. This number is the current size of errors based on the unknown for NN data.
8. Repeat the steps 2 - 7 for $Number\_of\_epochs$ times.
9. The last validation and train errors will be indicators of the current NN prediction error.

In general, the model was implemented following the next Algorithm 3:

Algorithm 3 Neural network forecasting

Input: LiDAR measurements and digital aerial photograph attributes X, volume values Y

Output: Predicted values $\hat{Y}$, calculated errors, prediction model.

1. Data preprocessing.
2. Choose the architecture and fix the hyperparameters of MLP.
3. Start a cycle
4. Divide the datasets into two parts: training and testing (xtrain, xtest, ytrain, ytest)
5. Build and train an MLP regression model with the training dataset
6. Calculate the predictions for the test dataset and compare it with test Y values. Estimate the error.
7. End of the cycle.
8. Repeat steps 2 - 7 for 100 times.

This neural network model was built by using Keras library for Python.

4.3 GRNN

The next model is GRNN. This model requires only one cycle for learning process and has only one hyperparameter - smoothness parameter. The full Algorithm 4 for the model is mostly the same as for previous, with difference in choice of hyperparameters:

Algorithm 4 Neural network forecasting

Input: LiDAR measurements and digital aerial photograph attributes X, volume values Y

Output: Predicted values $\hat{Y}$, calculated errors, prediction model.

1. Data preprocessing.
2. Estimate the smoothness parameter for a GRNN model.
3. Start a cycle
4. Divide the datasets into two parts: training and testing (xtrain, xtest, ytrain, ytest)
5. Build and train a GRNN regression model with the training dataset
6. Calculate the predictions for test X set and compare it with test Y values. Estimate the error.
7. End of the cycle
8. Repeat steps 2 - 7 for 100 times.

The choice of this parameter, in fact, contains the enumeration of its values end estimation of the error. The smallest error will correspond to the required parameter.

This neural network model was built by using the Neupy library for Python.

4.4 Principal Component Analysis

In section 5.1 the data set is presented and described. Presented graphs in the fig. 7 and 8 illustrate the high correlation between variables. And it seems that in order to decrease inaccuracy in the prediction model, the number of input variables could be decreased. For this purpose, the Principal Component Analysis algorithm was used.

PCA is one of the most classical algorithms that allows to approximate the original model by the lower dimension model. The aim of this algorithm is to find the most informative low dimension projections of the original observations X. It constructs a representation of the data by finding a linear basis of reduced dimensionality in which the variance is maximal.

This linear dimensionality reduction algorithm uses Singular Value Decomposition of the data to project it to a lower dimensional space.

PCA seeks for a linear mapping M which maximises the cost function trace ($M^T \Sigma M$) where

$$\Sigma_{ij} = cov(x_i, x_j) = E((x_i - \mu_i)(x_j - \mu_j)) \tag{6}$$

and x_i and x_j are ith and jth elements of X.

The principal components are the eigenvectors of the covariance matrix Σ . They are found by solving

$$\Sigma M = \lambda M, \tag{7}$$

and the new representation $\hat{X} = XM$. Scaling of the eigenvectors to unit length produces uncorrelated principal components whose variance is equal to the corresponding eigenvalue. To reduce the dimensionality, only the first L eigenvectors (in the decreasing order of eigenvalue, that is, variance) are selected.

5 EXPERIMENTS

5.1 Data

In this work, two different sites were used: Juuka (Dataset 1) and Karttula (Dataset 2). These datasets are the same as at the work of Junttila et.al. [15]. Each dataset consists of 38 variables obtained from LiDAR measurements, 2 feature variables obtained from the aerial photographs and 4 target variables of forest stand parameters.

The set of variables derived from LiDAR measurements consists of percentile points and cumulative percentile parts of the first and last pulse heights of non ground hits (height > 2 m), percentile intensities of first and last pulse intensities of non ground hits, mean of first pulse heights > 5m, standard deviation(SD) of first pulse height, and the number of measurements < 2m of first and last pulse heights divided by the total number of the same measurements of each plot.

The attributes derived from digital aerial photographs represent the percentage of all pixels in a photograph of a plot that are classified as hardwood (Hwd) and coniferous trees (Cnf). The classification was carried out by a human interpreter manually. Variables take values in the range from 0 to 100% and follow the following rule: Hwd + Cnf + hits to ground = 100%.

The forest stand parameter dataset, as the target variables, consists of Vt - the total volume and species-specific volumes of V1 - Scots pine, V2 - Norway spruce, and V3 - for hardwoods treated as a group, but mostly comprised of birch.

Thus, each dataset consist of two matrices: X_{ni}, Y_{ni} , where $X_{ni} = X_{n_i,lidar}; X_{n_i,aerial}, Y_{n_i} = V_{n_i,total}; V_{n_i,pine}; V_{n_i,spruce}; V_{n_i,birch}$; n_i - size of i dataset, i=1,2. X_{n_i} has a size $40N_i$, Y_{n_i} has a size $4 * N_i$.

Tables 1 and 2 illustrates the mean values and standard deviation for the dataset dependent values under consideration.

Table 1. Values of Mean and Standard Deviation for Volume parameters for Dataset 1.

Variable	V_{total}	V_1	V_2	V_3
Mean value	145.46	87.96	41.90	15.60
Standard deviation	81.19	68.81	73.27	28.92

Table 2. Values of Mean and Standard Deviation for Volume parameters for Dataset 2.

Variable	V_{total}	V_1	V_2	V_3
Mean value	205.91	52.71	109.55	43.65
Standard deviation	123.73	80.63	127.90	64.03

Separation of the dataset into training and test sets involves a random selection of the specified number of samples for verification. In this regard, there are variations in the training and assessment of the model because of the different separation of the data set. The Fig. 7 and 8 show a histograms of the distribution of quantities of the target values. It may be noticed that some ranges of values contain a small number of target values. It was decided not to reduce the number of samples. In this regard, in order to ensure the purity of the experiment, each model undergoes multiple repetitions of training with different divisions into the training and test sets. Further, to illustrate the results of the model, a mean value is taken for each characteristic. Further, in order to use necessary algorithms, the independent set of variables X for both datasets was scaled to $[0; 1]$ by using MinMaxScaler from the Scikit-learn library for Python.

Further analysis shows that the set of independent variables contains highly correlated data. Fig. 9 and 10 illustrate the plots of correlation matrix. Also, the PCA algorithm from the Scikit-learn library allows to illustrate the percentage of variance explained by each of the selected components. To show the percentage part of the influence of each variable, the entire dataset was passed through the algorithm without reducing the dimension. Tables 3 and 4 illustrates this information. In these tables, the values are sorted in descending order of the explained variance ratio meanings.

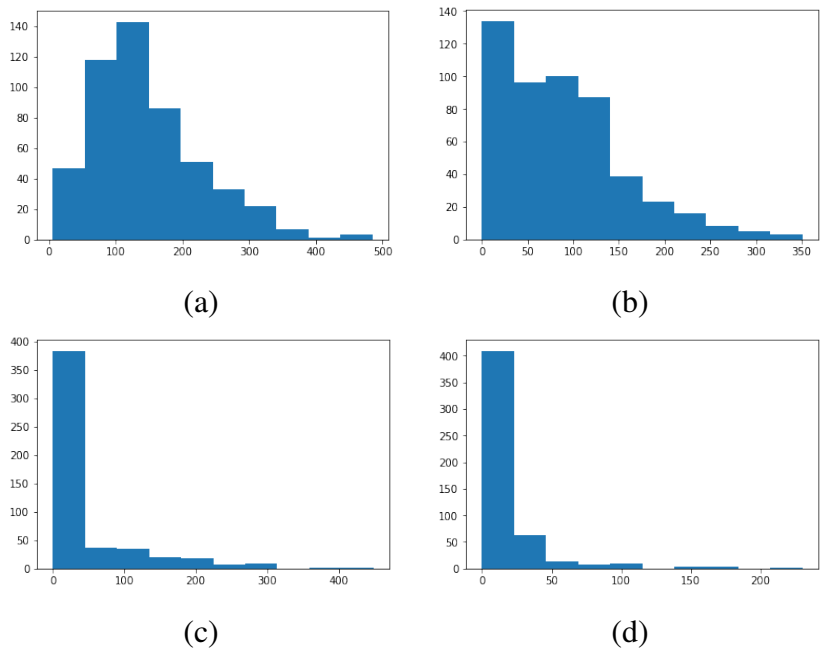


Figure 7. Histograms of distribution of volume at Dataset 1 for: (a) Total value; (b) Scots pine; (c) Norway spruce; (d) Birch.

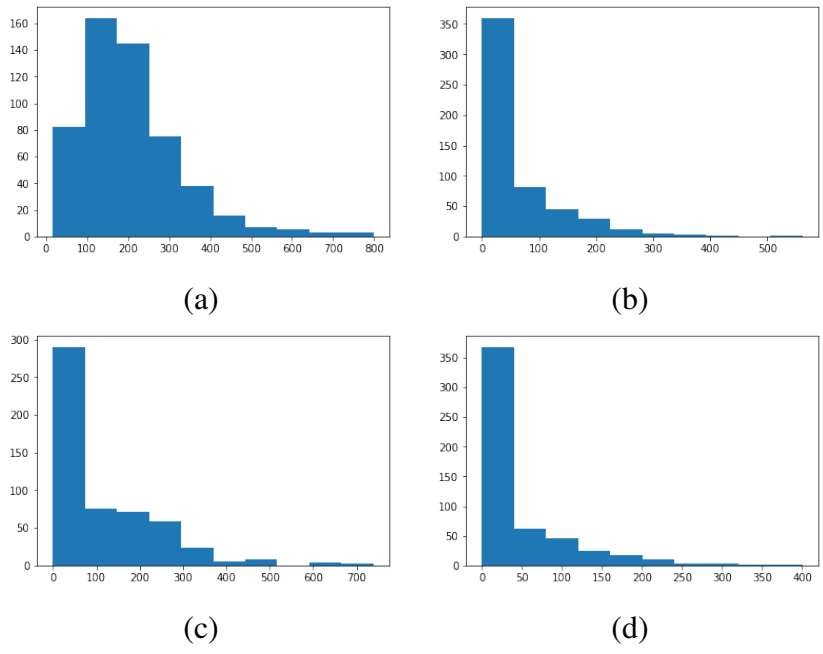


Figure 8. Histograms of distribution of volume at Dataset 2 for: (a) Total value; (b) Scots pine; (c) Norway spruce; (d) Birch.

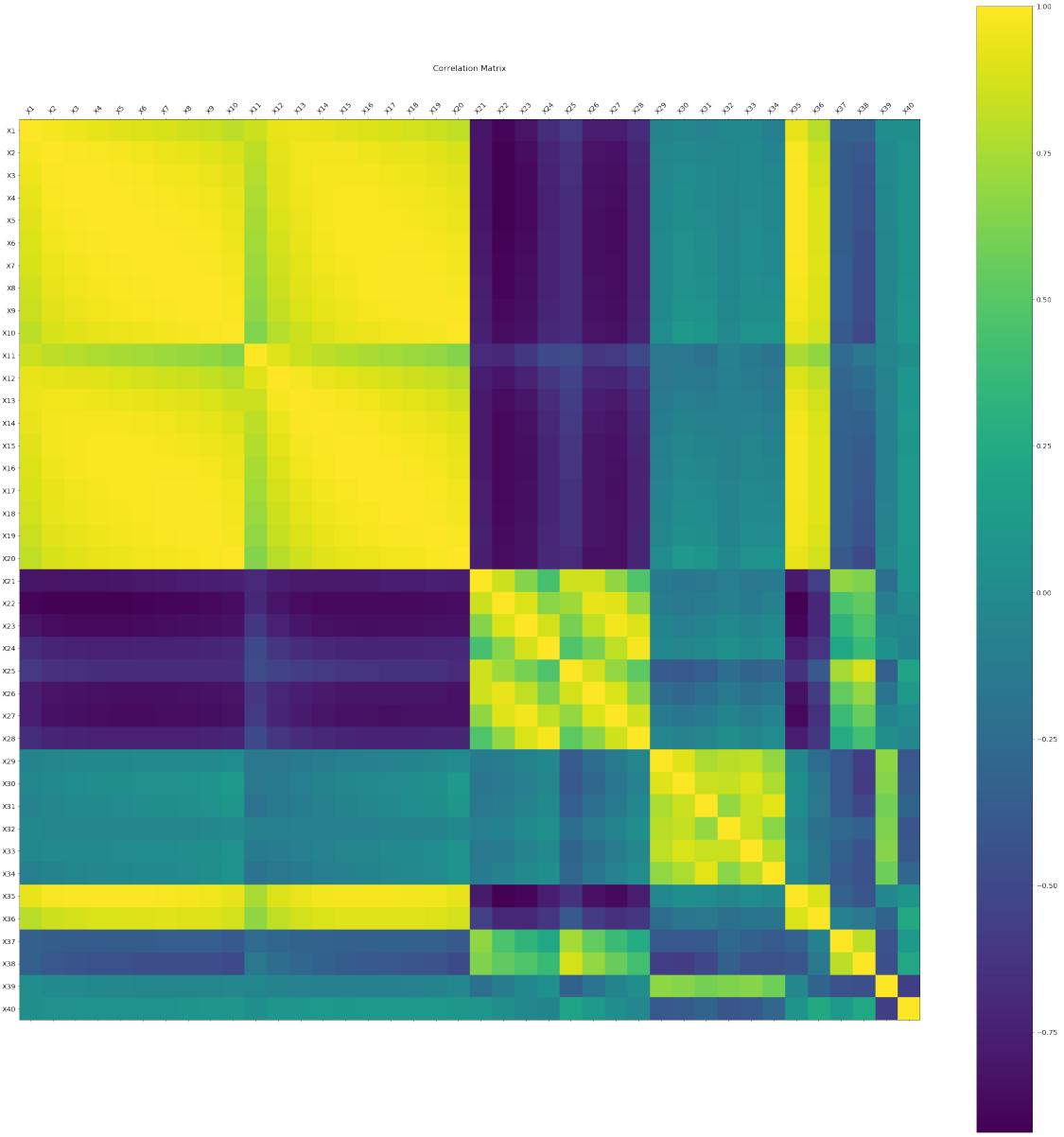


Figure 9. Correlation plot for independent values for Dataset 1.

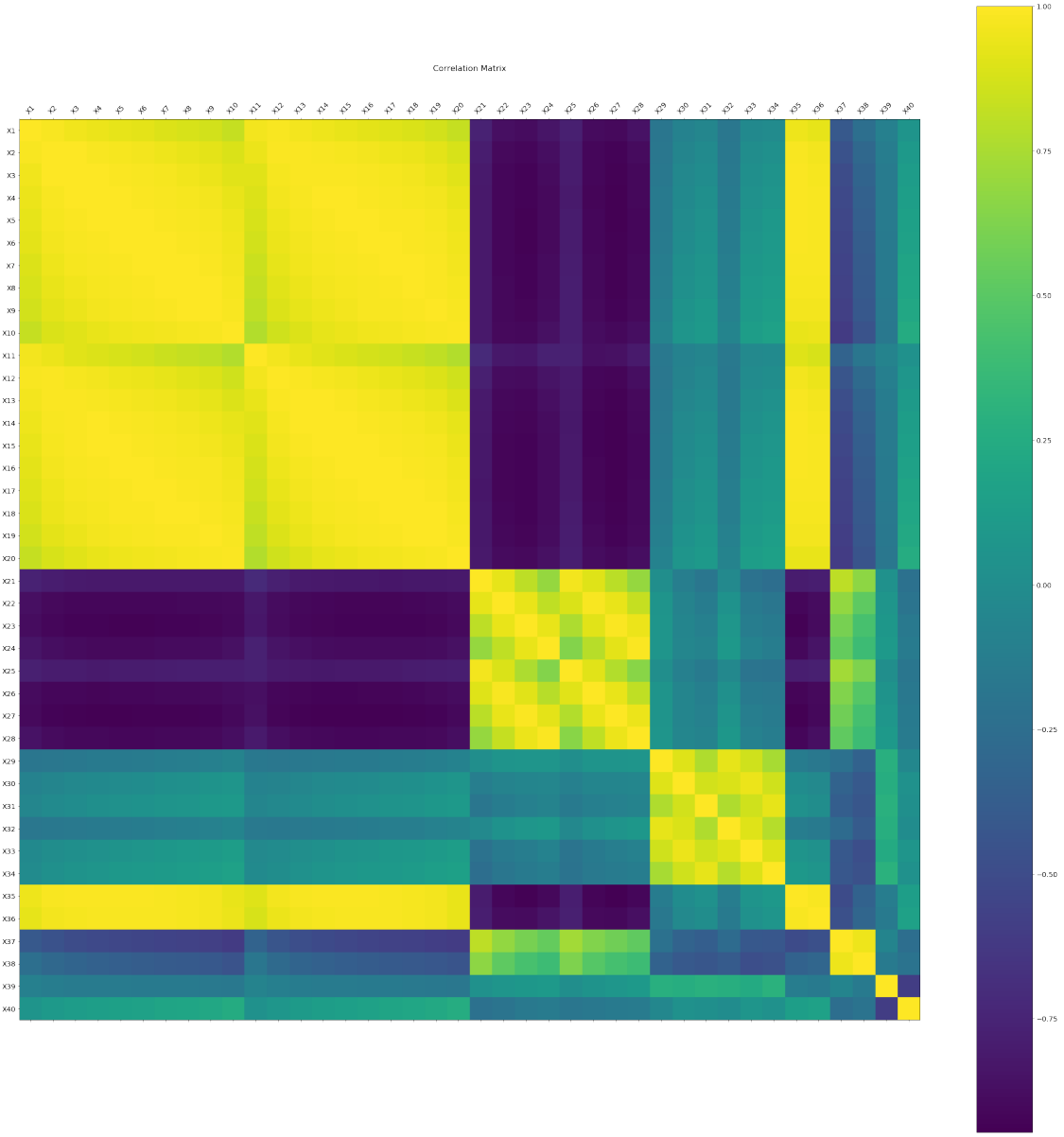


Figure 10. Correlation plot for independent values for Dataset 2.

Table 3. Explained variance ratio for Dataset 1.

Variable	X_1	X_2	X_3	X_4
Ratio	7.18923220e-01	1.28519137e-01	3.50104136e-02	2.64782965e-02
Variable	X_5	X_6	X_7	X_8
Ratio	2.32910636e-02	1.42609412e-02	1.04675259e-02	7.99610886e-03
Variable	X_9	X_{10}	X_{11}	X_{12}
Ratio	6.20139205e-03	4.56361871e-03	4.24610710e-03	3.35021174e-03
Variable	X_{13}	X_{14}	X_{15}	X_{16}
Ratio	2.61225345e-03	1.93844906e-03	1.71475858e-03	1.44734350e-03
Variable	X_{17}	X_{18}	X_{19}	X_{20}
Ratio	1.30949698e-03	1.11004656e-03	9.07677400e-04	8.50248723e-04
Variable	X_{21}	X_{22}	X_{23}	X_{24}
Ratio	7.00921540e-04	6.55914504e-04	5.30405063e-04	4.30660715e-04
Variable	X_{25}	X_{26}	X_{27}	X_{28}
Ratio	4.25728261e-04	2.80367640e-04	2.48741421e-04	2.37884900e-04
Variable	X_{29}	X_{30}	X_{31}	X_{32}
Ratio	1.90087249e-04	1.74625806e-04	1.68500231e-04	1.25097248e-04
Variable	X_{33}	X_{34}	X_{35}	X_{36}
Ratio	1.19727924e-04	1.04624229e-04	9.10472362e-05	8.01763878e-05
Variable	X_{37}	X_{38}	X_{39}	X_{40}
Ratio	7.51201411e-05	6.06928374e-05	5.84016592e-05	4.29641772e-05

Table 4. Explained variance ratio for Dataset 2.

Variable	X_1	X_2	X_3	X_4
Ratio	7.22439852e-01	1.00013613e-01	7.40403346e-02	3.26881198e-02
Variable	X_5	X_6	X_7	X_8
Ratio	1.88195082e-02	1.41592076e-02	1.18628716e-02	6.82522627e-03
Variable	X_9	X_{10}	X_{11}	X_{12}
Ratio	4.65258319e-03	2.48694869e-03	2.03346814e-03	1.49670831e-03
Variable	X_{13}	X_{14}	X_{15}	X_{16}
Ratio	1.44169616e-03	1.32711492e-03	7.69469674e-04	7.21916828e-04
Variable	X_{17}	X_{18}	X_{19}	X_{20}
Ratio	6.31053735e-04	5.59360885e-04	4.74428957e-04	3.73454593e-04
Variable	X_{21}	X_{22}	X_{23}	X_{24}
Ratio	3.25054251e-04	2.88836280e-04	2.23737221e-04	1.87516661e-04
Variable	X_{25}	X_{26}	X_{27}	X_{28}
Ratio	1.72464431e-04	1.46779679e-04	1.21084922e-04	1.00556307e-04
Variable	X_{29}	X_{30}	X_{31}	X_{32}
Ratio	9.44791581e-05	8.58115534e-05	7.63676770e-05	6.71689430e-05
Variable	X_{33}	X_{34}	X_{35}	X_{36}
Ratio	6.11191853e-05	5.58617560e-05	4.59460580e-05	3.46370398e-05
Variable	X_{37}	X_{38}	X_{39}	X_{40}
Ratio	2.82799689e-05	2.48694879e-05	2.31652045e-05	1.93262683e-05

5.2 Evaluation criteria

To evaluate the performance of the method and to compare the methods, the following performance measures were used. The performance of prediction was measured using Mean Square Error (MSE):

$$MSE = \frac{1}{M}\sum_{k=1}^{M}(Y_k - \hat{Y_k})^2, \tag{8}$$

where M is the number of samples, Y_k - target value, $\hat{Y_k}$ - predicted value.

For a better understanding and clarity, the following MSE conversions are used: Root

Mean Squared Error as a standard deviation of residuals and RMSE% .

$$RMSE = \sqrt{MSE}, \tag{9}$$

and

$$RMSE\% = RMSE/\overline{Y}, \tag{10}$$

where $\overline{Y}$ is the mean value of Y.

To illustrate how well the observed outcomes are replicated by the model, based on the proportion of total variation of outcomes explained by the model, the coefficient of determination R^2 was used:

$$R^2 = 1 - \frac{\sum_{k=1}^{M}(Y_k - \hat{Y}_k)^2}{\sum_{k=1}^{M}(Y_k - \overline{Y})^2}. \tag{11}$$

In addition to this, during the learning process of ANN, the Mean Absolute Error (MAE) is minimized:

$$MAE = \frac{1}{M}\sum_{k=1}^{M}|Y_k - \hat{Y}_k|, \tag{12}$$

And to estimate the bias of predictions, the next formula is used:

$$bias = \frac{1}{M}\sum_{k=1}^{M}(Y_k - \hat{Y}_k). \tag{13}$$

5.3 Description of experiments

To investigate the most efficient technique to perform the task of forest volume prediction three methods were compared: Linear Regression, Multilayer Perceptron, and Generalized Regression Neural Network. The experimental part of this thesis consists of building prediction models, selection of hyperparameters for these models and experiments with data processing. The results of predictions are illustrated in Tables 5–11.

The datasets were divided into two parts: train and test sets. The test set is made up of randomly selected samples. It includes 10% of all data. To avoid the errors associated with a random “successful” or “unsuccessful” selection of test samples, the process of dividing and fitting the model was included in the cycle with $RepeatNumber$ = 100.

After that, the actual error for the model estimates as the mean of errors for all test sets.

The first model is a Linear Regression model. It was chosen as the basis for determining the quality of neural network models. The model was built in Jupyter Notebook, by using Scikit-Learn library on Python v.3.6.9. The table 5 contain the results of the estimation.

Table 5. Estimated errors for predictions for Linear Regression model with Dataset 1.

Metrics	MAE	RMSE	RMSE%	Bias	R^2
V_{total}	19.61	26.96	18.34	-0.35	0.88
V_1	31.17	41.62	48.07	0.45	0.62
V_2	31.14	43.74	106	-0.84	0.60
V_3	10.27	15.43	95.59	0.04	0.66

The next model is the Multilayer Perceptron. Since the construction of the architecture of the neural network includes the selection of a large number of hyperparameters, it was decided to include several models of different complexity. Hyperparameters selection was provided manually. The MLP models were built in Jupyter Notebook, by using Keras library on Python v.3.6.9. It should be noted that, in contrast to the classical description of neural network models, in Keras, the input layer has an arbitrary dimension which is independent of the dimension of the input data. Whereas in the classical description the input layer of a neural network has the same dimension as the input variable. Instead of that, in the input layer of the NN in Keras, each neuron has the same dimension as the input variable.

In order to select the number of epochs for all MLP models, there were provided several numbers of experiments with a validation set. The validation set is a set of randomly chosen samples during the learning process. The model does not use these samples for training but they are used to check the results of predictions for every epoche. When the estimated forecast error on the validation set stops falling, it means that it is necessary to stop the learning process to avoid the overlearning process. After the number of epochs was set for each MLP model, the validation set was eliminated to give the neural network more samples for training. Fig. 11 illustrates an example of a change in the error on the training and validation sets depending on the epochs for the MLP2 model. For the first learning process in order to estimate the number of epochs, the validation split was 10%, and the number of epochs was 500. Observing the learning process, it was possible to

notice that the decrease in the error value on the validation data ceases and stabilizes at about 90 - 110 epochs.

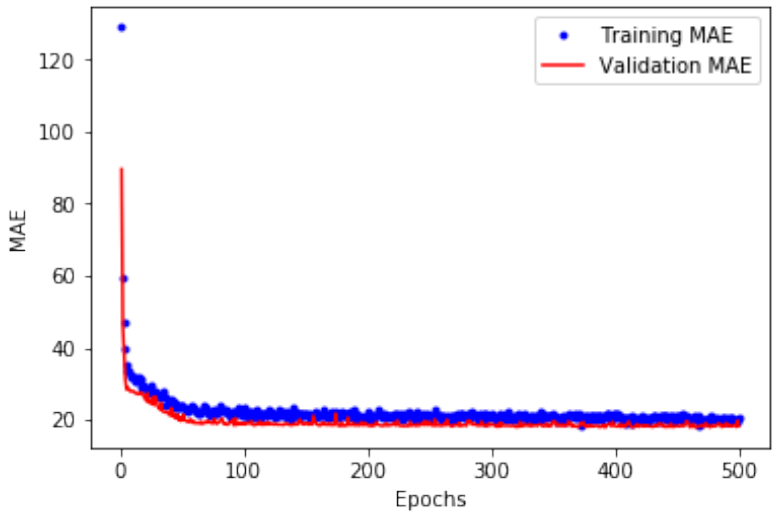


Figure 11. Graph of MAE changes for validation and train sets

The first model MLP1 consists of 40 neurons at the input layers with linear activation function and 1 neuron in the output layer with the linear activation function. In fact, this model describes the linear regression model in terms of neural networks (Fig. 12).

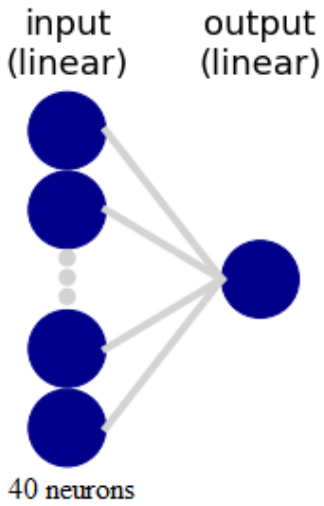


Figure 12. Architecture of the MLP1 model

Table 6. Estimated errors for predictions for MLP1 model with Dataset 1.

Metrics	MAE	RMSE	RMSE%	Bias	R^2
V_{total}	23.20	31.80	21.67	1.11	0.84
V_1	31.52	43.17	49.79	0.74	0.60
V_2	32.98	46.13	111.15	0.58	0.57
V_3	10.03	15.16	93.67	0.04	0.68

As it seen in Fig. 13, the second model MLP2 consist of 128 neurons at input layer with ReLU activation function. It has only one hidden layer with 32 neurons and the same activation function. At the output layer, there is only one neuron with a linear activation function, which is a prerequisite for solving the regression problem using ANN. The size of the batch set was chosen as 5. The "Adam" was chosen as the optimizer algorithm instead of the classical SGD. The loss function was estimated in terms of MSE.

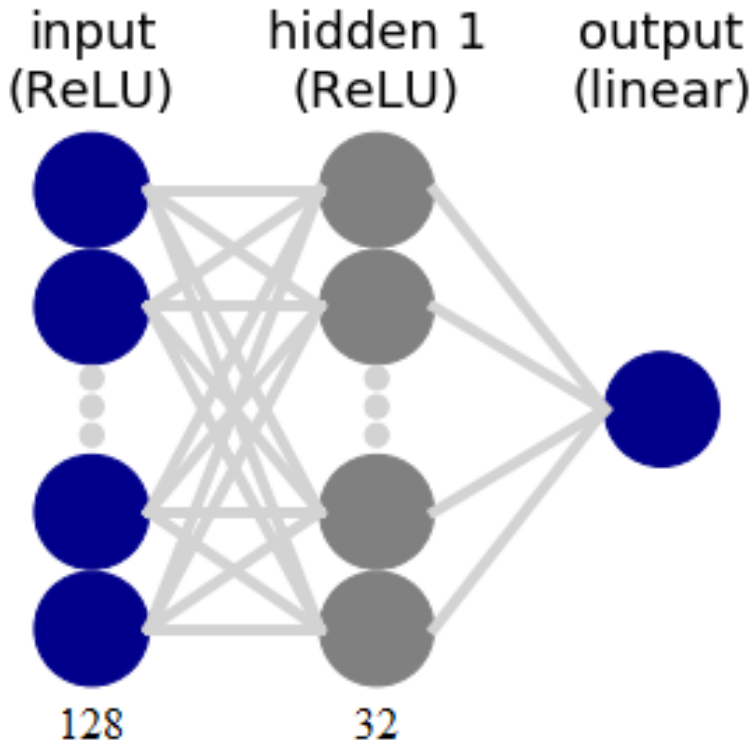


Figure 13. Architecture of the MLP2 model

Table 7. Estimated errors for predictions for MLP2 model with Dataset 1.

Metrics	MAE	RMSE	RMSE%	Bias	R^2
V_{total}	19.13	26.55	18.06	1.15	0.89
V_1	25.32	35.93	41.41	1.07	0.72
V_2	22.05	40.05	95.92	1.06	0.67
V_3	8.36	13.7	84.88	0.32	0.73

One of the most common problems for ANN is overlearning. Overlearning is the process when the NN remembered some features that are specific for the training set but do not match the attribute in general. To avoid this problem there are several methods. One of them is Dropout.

Dropout is a regularization method for artificial neural networks, designed to prevent network retraining. The essence of the method is that in the learning process, a layer is selected from which a certain number of neurons (for example, 30%) are randomly thrown, which are turned off from further calculations. This technique improves the effectiveness of training and the quality of the result. More trained neurons gain more weight on the network. Fig. 14 illustrates this process.

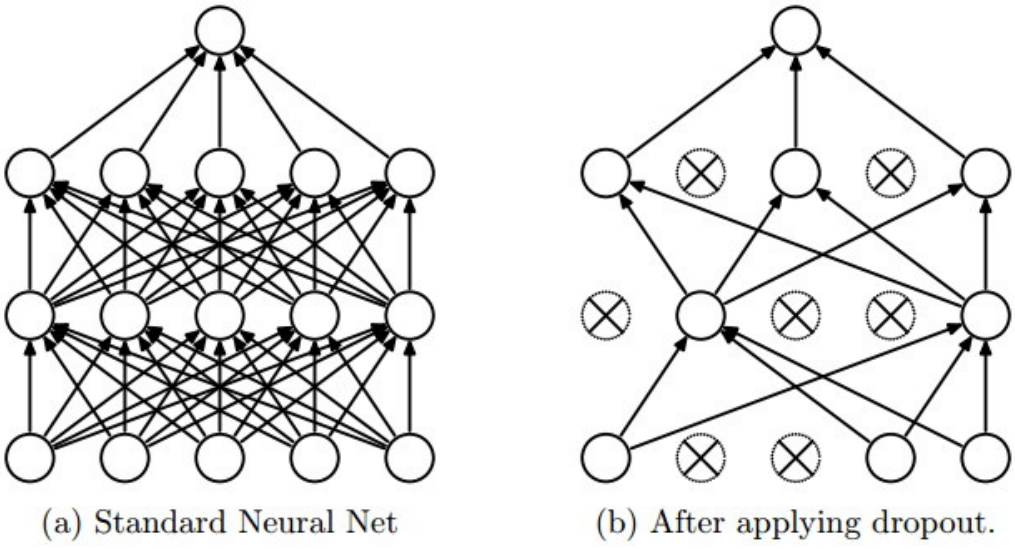


Figure 14. Dropout application.

For the next model MLP3, a more complicated architecture was used. The model consists of the input layer with 800 neurons and ReLU as an activation function. The hidden part includes two layers. The first hidden layer, as an input layer, includes 800 neurons with ReLU as an activation function. The second hidden layer includes 128 neurons. The output layer consists of one neuron with a linear activation function. Fig. 15 illustrates this architecture. As the next step, Dropout layers were added between the layers of this model. The Table 8 illustrates the results of the final model.

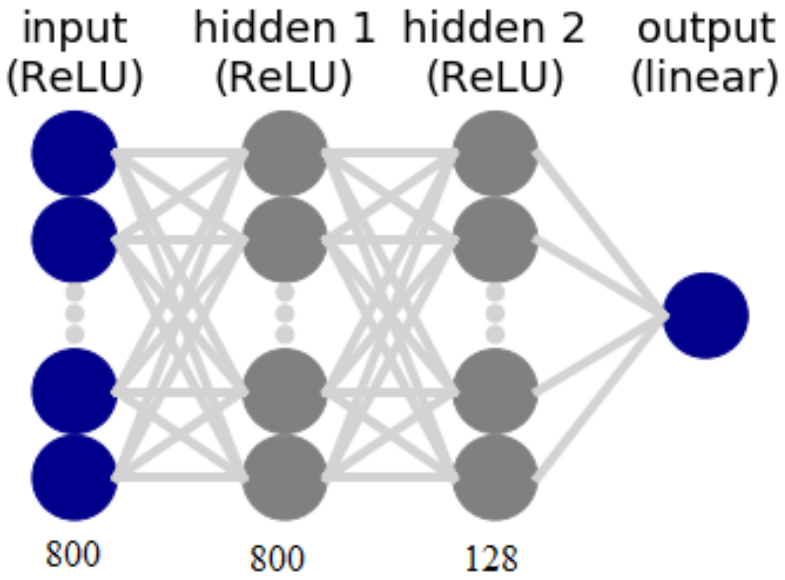


Figure 15. Architecture of the MLP3 model

Table 8. Estimated errors for predictions for MLP3 model with Dataset 1.

Metrics	MAE	RMSE	RMSE%	Bias	R^2
V_{total}	20.15	27.78	18.90	-0.49	0.88
V_1	25.23	35.98	41.45	1.90	0.71
V_2	20.93	38.81	93.13	-1.78	0.68
V_3	8.53	14.41	89.05	0.04	0.70

The MLP4 model is more simple than the previous one. But, to increase the nonlinearity, the sigmoidal functions were chosen as activation functions. Fig. 16 illustrates its architecture. At the input layer, there are 800 neurons with a sigmoid activation function. The

hidden part of MLP consists of two layers. At the first hidden layer, there are 32 neurons with the ReLU activation function. The second hidden layer includes 8 neurons also with the ReLU activation function. And one neuron at the output layer with a linear activation function. Besides that, this model also contains Dropout layers between all layers of NN. The results are presented in table 9.

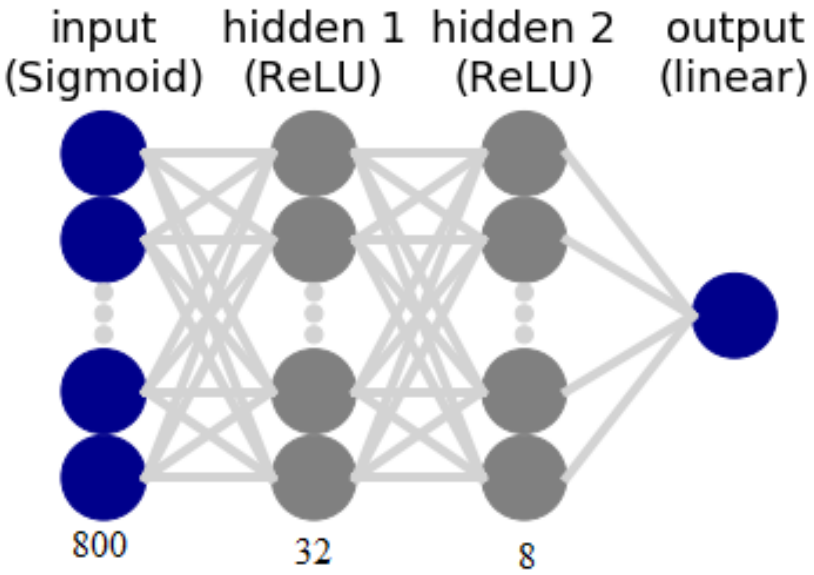


Figure 16. Architecture of the MLP4 model

Table 9. Estimated errors for predictions for MLP4 model with Dataset 1.

Metrics	MAE	RMSE	RMSE%	Bias	R^2
V_{total}	19.60	27.10	18.43	3.94	0.88
V_1	28.77	39.55	45.65	0.44	0.66
V_2	24.78	42.47	101.73	1.94	0.63
V_3	8.68	14.18	87.71	0.45	0.71

The next model is Generalized Regression Neural Network. The model was built in Jupyter Notebook, by using Neupy library on Python v.3.6.9.

At first, this model required to determine the smoothness parameter. There are several methods to estimate the smoothness parameter. In this work, it was made by finding the

smallest error of the prediction model. Here the model takes the whole dataset (without dividing for train and test sets) and trying to approximate it. The correct smoothness parameter is taken from the network with the smallest error. After that, the smoothness parameter is applied to the new GRNN model. As it was described in CHAPTER 4.3 and as it is illustrated in Fig. 17, the GRNN model consists of four layers. The first and second layers include 40 neurons. The activation function on the second layer is a Gaussian kernel. The third layer consists of two summation neurons. And output layer as it was previously, consists of one neuron. Tha GRNN uses lazy learning which means that the network does not need iterative training. It stores the parameters and uses them to make predictions.

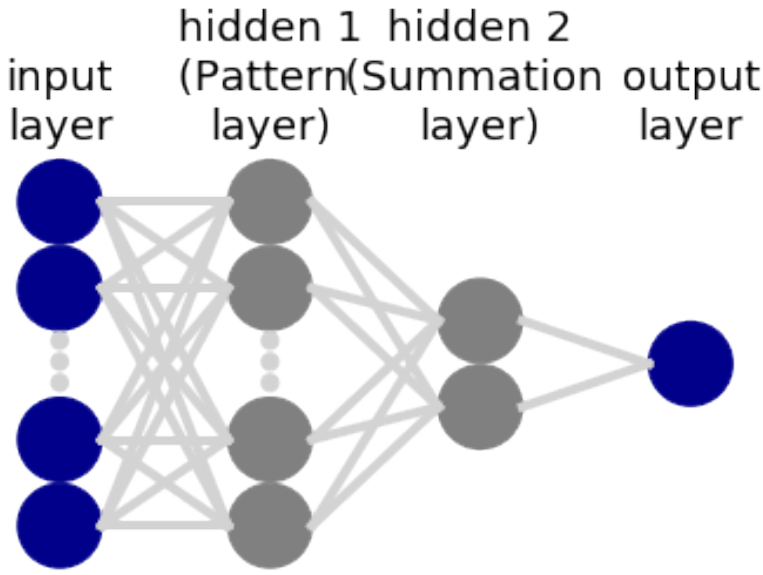


Figure 17. Architecture of the GRNN model

Table 10. Estimated errors for predictions for GRNN model with Dataset 1.

Metrics	MAE	RMSE	RMSE%	Bias	R^2
V_{total}	23.66	31.59	21.49	-0.28	0.85
V_1	32.43	44.04	50.84	-1.68	0.58
V_2	29.54	45.47	108.51	-0.65	0.59
V_3	9.87	15.90	97.14	2.05	0.66

The last experimental model was provided similar to the MLP3 model with 3 neurons and the SoftMax activation function on the output layer. This model was trying to approximate only species-specific variables.

A SoftMax activation function is usually used for classification tasks when there are more than two classes. Its result can be interpreted as the probability of belonging to the class. But in fact, this is a function that takes as input a vector of K real numbers and normalizes it into a probability distribution consisting of K probabilities proportional to the exponentials of the input numbers. After applying softmax, each component will be in the interval (0, 1) and the components will sum up to 1.

So, if the species-specific measurements can be summarized to the known total variable, then, we can use the NN with a softmax activation function to predict what part of the total volume of wood does a particular species occupy. For this purpose, the output variable Y was determined as:

$$Ynew_{i,sp} = \frac{Y_{i,sp}}{Y_{i,total}} \tag{14}$$

where i is the number of sample, sp- species.

In this case, the target output matrix $Ynew$ would consist of the proportions of the volumes of each species to the total volume. And in this case, the SoftMax activation function could be implemented.

The architecture for this neural network was taken the same as for the third MLP model (Fig. 18). It consists of 800 neurons at the input layer and the first hidden layer with the ReLU activation function and the second hidden layer with 128 neurons and the same activation function. And three neurons at the output layer with a SoftMax activation function. In order to estimate the error at the same ranges as for other models, there is a need to multiply predicted $\hat{Ynew}$ variables to the total value. And it keeps the problem. For future use, this model will need prior knowledge about the total values of the variable. For example, it could use the predicted total volumes from previous models, but then the value will be affected by the error of total predictions and the results keep a really huge error. For example, average RMSE% for this model is about 150%. But, in order to demonstrate the capabilities of such a model, the estimations with a prior knowledge about total volume were provided at the table 11.

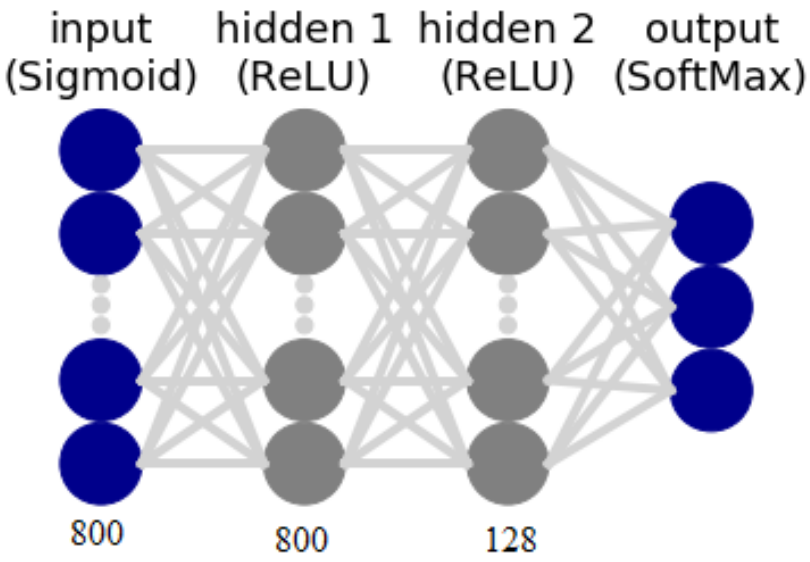


Figure 18. Architecture of the MLP5 model

Table 11. Estimated errors for predictions for MLP5 model with Dataset 1.

Metrics	MAE	RMSE	RMSE%	Bias	R^2
With real total values					
V_1	18.45	33.57	38.16	-0.20	0.74
V_2	18.21	35.01	83.55	0.06	0.73
V_3	8.71	15.36	98.45	-0.26	0.65

5.4 Results

The table 12 illustrates the comparison of the models results. The results are illustrated with RMSE%, R^2, and bias.

Table 12. Comparison of the results for Dataset 1.

RMSE%				
Model	V_{total}	V_1	V_2	V_3
LR	18.34	48.07	106.02	95.59
MLP1	21.67	49.79	111.15	93.66
MLP2	18.06	41.41	95.92	84.88
MLP3	18.90	41.34	93.13	89.05
MLP4	18.43	45.65	101.73	87.71
MLP5	-	38.16	83.56	98.45
GRNN	20.72	44.89	100.89	95.51
R^2				
Model	V_{total}	V_1	V_2	V_3
LR	0.88	0.62	0.60	0.66
MLP1	0.84	0.60	0.57	0.68
MLP2	0.89	0.72	0.67	0.73
MLP3	0.88	0.72	0.68	0.70
MLP4	0.88	0.66	0.63	0.71
MLP5	-	0.74	0.73	0.65
GRNN	0.86	0.67	0.64	0.68
Bias				
Model	V_{total}	V_1	V_2	V_3
LR	-0.35	0.46	-0.84	0.04
MLP1	1.11	0.74	0.58	0.05
MLP2	1.15	1.07	1.06	0.32
MLP3	-0.49	1.88	-1.78	0.04
MLP4	3.94	0.44	1.94	0.45
MLP5	-	0.20	0.06	-0.26
GRNN	-1.53	-0.27	-2.38	1.79

As it seems, the model MLP2 showed the best results for most of the target variables. Of course, for species-specific estimation, the model with a SoftMax activation function is better, but without prior knowledge about total volume values, it is impossible to achieve such results.

In order to improve the results and to make the models easier and clearer, all the exper-

iments above were also modified with a data preprocessing. The Original dataset was modified with a PCA algorithm. As described at subchapter 5.1, the necessary information could be described with a less number of variables. In that case, for the first experiment, it was decided to keep the variables that explain 99% of the variance. It allowed reducing the dimensionality from 40 to 16 variables. In order to keep the information obtained from aerial photographs which are important for species-specific classification, variables X_{39} and X_{40} were also added, after the PCA algorithm implementation. Table 13 illustrates the evaluations of this predictions.

Table 13. Comparison of the results for Dataset 1 with using PCA (keeping 99% of variance information)

RMSE%				
Model	V_{total}	V_1	V_2	V_3
LR	18.25	48.68	108.03	94.61
MLP1	18.25	48.68	108.06	94.66
MLP2	18.23	39.84	86.41	89.33
MLP3	21.25	43.89	89.27	91.26
MLP4	18.73	45.14	98.41	89.86
MLP5	-	34.77	74.80	103.05
GRNN	21.01	45.93	103.81	96.48
R^2				
Model	V_{total}	V_1	V_2	V_3
LR	0.88	0.62	0.59	0.67
MLP1	0.88	0.61	0.59	0.67
MLP2	0.89	0.72	0.67	0.73
MLP3	0.88	0.72	0.68	0.70
MLP4	0.88	0.66	0.63	0.71
MLP5	-	0.78	0.79	0.61
GRNN	0.86	0.67	0.64	0.68
Bias				
Model	V_{total}	V_1	V_2	V_3
LR	-0.35	0.49	-0.74	-0.06
MLP1	-0.43	-0.61	-0.41	-0.04
MLP2	1.15	1.07	1.06	0.32
MLP3	-0.49	1.88	-1.78	0.04
MLP4	3.94	0.44	1.94	0.45
MLP5	-	1.09	-0.94	-0.15
GRNN	-1.53	-0.27	-2.38	1.79

But this process has its own limits. For example, the table 14 illustrates the estimations for the dataset that keeps 90% of the information. In fact it contains only variables: X_1...X_4 (and additional X_{39} and X_{40}) and it affects to the results. As it seems in the table below, the quality of predictions decreased.

Table 14. Comparison of the results for Dataset 1 with using PCA (keeping 90% of variance information).

RMSE%				
Model	V_{total}	V_1	V_2	V_3
LR	18.44	56.58	124.94	98.09
MLP1	18.61	56.51	125.66	97.91
MLP2	18.10	46.67	105.79	85.83
MLP3	19.59	50.56	118.54	89.59
MLP4	19.10	52.38	121.67	93.09
MLP5	-	53.39	117.04	104.69
GRNN	20.62	48.19	107.43	93.39
R^2				
Model	V_{total}	V_1	V_2	V_3
LR	0.88	0.48	0.46	0.65
MLP1	0.88	0.48	0.45	0.65
MLP2	0.89	0.64	0.60	0.72
MLP3	0.87	0.58	0.49	0.70
MLP4	0.87	0.55	0.48	0.67
MLP5	-	0.51	0.48	0.60
GRNN	0.86	0.62	0.61	0.72
Bias				
Model	V_{total}	V_1	V_2	V_3
LR	-0.30	0.49	-0.74	-0.06
MLP1	0.04	0.36	0.02	-0.07
MLP2	0.13	1.17	0.95	0.17
MLP3	1.15	1.86	0.87	0.46
MLP4	3.26	2.68	2.78	1.78
MLP5	-	0.07	0.19	-0.26
GRNN	-0.89	0.12	-1.82	1.13

As a result, for predictions of the total volume, six components are enough to keep the prediction accuracy with the increasing of the errors less than 1%. But for species-specific predictions, the errors are increased rapidly.

6 DISCUSSION

6.1 Current study

From the obtained results, it can be seen that ANN models do a good job of estimating the model. Comparing to linear regression, it can be noticed that for the Vtotal parameter the difference in the errors is small, and some models are even inferior to the LR model. Given the simplicity of the LR model, more complex models, such as ANNs, could be neglected. But to characterize species-specific parameters that have a more non-linear relationship with the input data, ANN models showed a much more superior result. In addition, the set of output data contains errors in the measurements, which does not give the models a more accurate approximation. However, the characteristic R^2 shows a measure of the dependence of the output and predicted data for each constructed model. A higher score for ANNs also indicates that these models better explain data behavior. Besides that, the ANN models have the smallest biases of predictions.

In general, the MLP2 model shows the best result in all respects. Other models with more neural connections, in general, show worse results, sometimes surpassing the MLP2 model in some characteristics. The GRNN model shows the worst result compared to other ANN models. This can be explained by the distribution of the quantities of the target parameters. In addition, for more complex dependences of species-specific parameters, the GRNN model shows enough good results.

The PCA method also helped to improve results for all models. One of the features of neural networks is automatically adjusted weights, which determine the significance of the contribution of a variable. Despite this, the use of a lower-dimensional input dataset helped to improve the results. And this is understandable because for the NN of the same architecture a simpler model is supplied. After this, there is no need to take into account the contribution of the components that have the least impact, and this would reduce the error. However, the reduction in dimension should be justified, and the dataset should retain important information in the fullest possible way. While maintaining 90% of the information that the input data describes, the error is already starting to grow. Of course, it should be mentioned that with keeping 90% of the explained variance ratio, the error did not grow very much. It means, that the stored variables describe the biggest part of an aimed function.

6.2 Future work

At the moment, the development of machine learning algorithms is gaining more and more popularity. The use of ANNs covers an ever wider range of tasks; deep learning methods show very promising results. Every year new models appear and show better and better results. Therefore, a further solution to this problem can progress using other ANN models. Many types of neural networks are able to work with images and point clouds directly. In further works, to determine the necessary features, laser scans can be processed, for example, by using convolutional neural networks. Such examples already exist, for example, the work [28], where the authors classified objects in images obtained from airborne laser scanning using convolutional neural networks.

Also, despite its widespread popularity, the interpretation of ANN models is still difficult. The selection of architecture and hyperparameters often has to be done manually, sometimes one has to guess. But modern implementations of machine learning libraries can automate this process as well. For further development of this topic, the grid-search module from the Scikit-Learn library can be used, as well as the recently appeared Keras-tuner module for the Keras library. Although the search for the necessary parameters is mostly carried out by enumeration, these modules help to automate the process and find the best model characteristics.

7 CONCLUSION

This work was aimed at studying and analyzing models of neural networks to predict forest stand parameters, namely, the volume of wood. The data obtained from LiDAR scans, airborne photographs and measurements of forests were used as datasets. The data correspond to territories of study sites in Juuka and Karttula. As models of neural networks, several MLP and GRNN architectures were considered. A comparison is made with the linear regression method. Also, in order to study the dimensionality reduction influence, the principal component method was used. As can be seen from the presented results, ANNs have proven themselves to solve this regression problem. Of the minuses for MLP, it can be specified a longer training time than for other models. Also, the bigger the dataset is used, the more accurate the result will be. For GRNN, the restrictions on the dataset volume are much smaller, and the execution time is shorter. However, despite a good recommendation in the literature, this type of neural network is poorly suited to solve this problem. The percentage of errors is higher than for MLP.

The method of principal component analysis also improved prediction results. Despite the fact that ANNs are able to automatically adjust weights and distribute the strength of the influence of certain variables, a certain screening of unnecessary variables allowed to increase the accuracy. However, it should be noted that there is a sufficiently high threshold, after which the quality of the predicted values begins to decrease sharply.

REFERENCES

[1] Erik Naesset. Determination of mean tree height of forest stands using airborne laser scanner data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 52(2):49–56, 1997.

[2] Erik Næsset. Predicting forest stand characteristics with airborne scanning laser using a practical two-stage procedure and field data. *Remote sensing of environment*, 80(1):88–99, 2002.

[3] M Maltamo, J Malinen, P Packalén, A Suvanto, and J Kangas. Nonparametric estimation of stem volume using airborne laser scanning, aerial photography, and stand-register data. *Canadian Journal of Forest Research*, 36(2):426–436, 2006.

[4] Virpi Junttila, Matti Maltamo, and Tuomo Kauranne. Sparse bayesian estimation of forest stand characteristics from airborne laser scanning. *Forest Science*, 54(5):543–552, 2008.

[5] Ilkka Korpela, Hans Ole Ørka, Matti Maltamo, Timo Tokola, Juha Hyyppä, et al. Tree species classification using airborne lidar–effects of stand and tree parameters, downsizing of training set, intensity normalization, and sensor type. *Silva Fennica*, 44(2):319–339, 2010.

[6] Øystein Rudjord and Øivind Due Trier. Tree species classification with hyperspectral imaging and lidar. In *2016 8th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4. IEEE, 2016.

[7] Xinhuai Zou, Ming Cheng, Cheng Wang, Yan Xia, and Jonathan Li. Tree classification in complex forest point clouds based on deep learning. *IEEE Geoscience and Remote Sensing Letters*, 14(12):2360–2364, 2017.

[8] Arne Nothdurft, Joachim Saborowski, and Johannes Breidenbach. Spatial prediction of forest stand variables. *European Journal of Forest Research*, 128(3):241–251, 2009.

[9] Carlos Silva, Carine Klauberg, Andrew Hudak, Lee Vierling, Wan Jaafar, Midhun Mohan, Mariano Garcia, António Ferraz, Adrián Cardil, and Sassan Saatchi. Predicting stem total and assortment volumes in an industrial pinus taeda l. forest plantation using airborne laser scanning data and random forest. *Forests*, 8(7):254, 2017.

[10] Petri Varvia, Timo Lähivaara, Matti Maltamo, Petteri Packalen, and Aku Seppänen. Gaussian process regression for forest attribute estimation from airborne laser scanning data. *IEEE Transactions on Geoscience and Remote Sensing*, 2018.

[11] Colin J Gleason and Jungho Im. Forest biomass estimation from airborne lidar data using machine learning approaches. *Remote Sensing of Environment*, 125:80–91, 2012.

[12] Colin Gleason and Jungho Im. A fusion approach for tree crown delineation from lidar data. *Photogrammetric Engineering & Remote Sensing*, 78:679–692, 07 2012.

[13] Harri Niska, Jukka-Pekka Skon, Petteri Packalen, Timo Tokola, Matti Maltamo, and Mikko Kolehmainen. Neural networks for the prediction of species-specific plot volumes using airborne laser scanning and aerial photographs. *IEEE Transactions on Geoscience and Remote Sensing*, 48(3):1076–1085, 2009.

[14] Jorge García-Gutiérrez, Francisco Martínez-Álvarez, A Troncoso, and José C Riquelme. A comparison of machine learning regression techniques for lidar-derived estimation of forest variables. *Neurocomputing*, 167:24–31, 2015.

[15] Virpi Junttila, Tuomo Kauranne, and Vesa Leppänen. Estimation of forest stand parameters from airborne laser scanning using calibrated plot databases. *Forest Science*, 56(3):257–270, 2010.

[16] Virpi Junttila, Tuomo Kauranne, Andrew O Finley, and John B Bradford. Linear models for airborne-laser-scanning-based operational forest inventory with small field sample size and highly correlated lidar data. *IEEE transactions on geoscience and remote sensing*, 53(10):5600–5612, 2015.

[17] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[18] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[19] Oludolapo A Olanrewaju, Adisa A Jimoh, and Pulek A Kholopane. Comparison between regression analysis and artificial neural network in project selection. In *2011 IEEE International Conference on Industrial Engineering and Engineering Management*, pages 738–741. IEEE, 2011.

[20] Amir Heydari, Keivan Shayesteh, and Ladan Kamalzadeh. Prediction of hydrate formation temperature for natural gas using artificial neural network. *Oil and Gas Business*, (2), 2006.

[21] Arvin H Fernando, Isidro Antonio V Marfori, and Archie B Maglaya. A comparative study between artificial neural network and linear regression for optimizing a hinged blade cross axis turbine. In *2015 International Conference on Humanoid,*

Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), pages 1–4. IEEE, 2015.

[22] Donald F Specht. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[23] Shaoqing Liu, Qianlai Zhuang, Yujie He, Asko Noormets, Jiquan Chen, and Lianhong Gu. Evaluating atmospheric co2 effects on gross primary productivity and net ecosystem exchanges of terrestrial ecosystems in the conterminous united states using the ameriflux data and an artificial neural network approach. *Agricultural and forest meteorology*, 220:38–49, 2016.

[24] Matthias M Bauer. General regression neural network for technical use. 1995.

[25] Joko Lianto Buliali, Victor Hariadi, Ahmad Saikhu, and Saprina Mamase. Generalized regression neural network for predicting traffic flow. In *2016 International Conference on Information & Communication Technology and Systems (ICTS)*, pages 199–202. IEEE, 2016.

[26] Chunyan Wu, Yongfu Chen, Changhui Peng, Zhaochen Li, and Xiaojiang Hong. Modeling and estimating aboveground biomass of dacrydium pierrei in china using machine learning with climate change. *Journal of Environmental Management*, 234:167 – 179, 2019.

[27] First Teddy Surya Gunawan, Mira Kartiwi, Noreha Abd Malik, and Nanang Ismail. Food intake calorie prediction using generalized regression neural network. In *2018 IEEE 5th International Conference on Smart Instrumentation, Measurement and Application (ICSIMA)*, pages 1–4. IEEE, 2018.

[28] Aldino Rizaldy, Claudio Persello, Caroline Gevaert, Sander Oude Elberink, and George Vosselman. Ground and multi-class classification of airborne laser scanner point clouds using fully convolutional networks. *Remote sensing*, 10(11):1723, 2018.

Appendix 1. Tables of results for Dataset 2.

Tables of results for Dataset 2.

Table A1.1. Comparison of the results for Dataset 2.

RMSE%				
Model	V_{total}	V_1	V_2	V_3
LR	27.28	134.97	76.46	111.37
MLP1	28.39	130.25	81.95	107.87
MLP2	24.52	119.63	66.75	92.11
MLP3	26.32	124.09	57.13	100.07
MLP4	25.44	121.75	67.94	96.21
MLP5	-	116.53	50.91	106.57
GRNN	28.28	128.08	67.32	104.81
R^2				
Model	V_{total}	V_1	V_2	V_3
LR	0.77	0.19	0.56	0.39
MLP1	0.76	0.27	0.51	0.42
MLP2	0.82	0.37	0.68	0.57
MLP3	0.79	0.32	0.74	0.48
MLP4	0.81	0.31	0.64	0.53
MLP5	-	0.41	0.79	0.39
GRNN	0.77	0.29	0.66	0.48
Bias				
Model	V_{total}	V_1	V_2	V_3
LR	-3.87	-12.65	-33.58	42.31
MLP1	3.58	3.38	-2.09	0.45
MLP2	6.45	8.54	-0.32	3.11
MLP3	4.22	5.70	2.04	2.50
MLP4	3.47	13.00	10.36	6.27
MLP5	-	5.81	-2.101	-3.71
GRNN	-3.85	0.69	-10.83	5.54

(continues)

Appendix 1. (continued)

Table A1.2. Comparison of the results for Dataset 2 with using PCA (keeping 99% of variance information)

RMSE%				
Model	V_{total}	V_1	V_2	V_3
LR	26.85	129.10	75.73	106.01
MLP1	27.06	129.80	77.43	106.29
MLP2	25.18	114.46	59.69	87.46
MLP3	27.21	125.31	61.25	95.77
MLP4	25.53	121.81	70.85	97.56
MLP5	-	118.62	54.12	97.96
GRNN	28.06	128.27	64.13	101.05
R^2				
Model	V_{total}	V_1	V_2	V_3
LR	0.78	0.28	0.58	0.44
MLP1	0.78	0.27	0.56	0.44
MLP2	0.81	0.42	0.74	0.62
MLP3	0.77	0.29	0.71	0.52
MLP4	0.81	0.35	0.61	0.53
MLP5	-	0.39	0.74	0.52
GRNN	0.77	0.28	0.69	0.51
Bias				
Model	V_{total}	V_1	V_2	V_3
LR	1.32	3.51	-2.75	0.57
MLP1	1.55	3.21	-2.12	0.29
MLP2	4.04	3.04	1.98	1.38
MLP3	4.39	5.01	2.03	3.53
MLP4	4.14	9.12	11.20	4.46
MLP5	-	1.09	-0.94	-2.26
GRNN	7.52	-5.26	-9.84	4.64

(continues)

Appendix 1. (continued)

Table A1.3. Comparison of the results for Dataset 2 with using PCA (keeping 90% of variance information).

RMSE%				
Model	V_{total}	V_1	V_2	V_3
LR	29.99	135.21	83.21	108.32
MLP1	30.20	134.71	84.22	108.74
MLP2	25.84	126.25	70.02	92.85
MLP3	26.38	136.61	68.93	95.34
MLP4	26.75	134.01	72.60	103.74
MLP5	-	144.21	66.18	117.68
GRNN	28.19	132.46	67.36	103.28
R^2				
Model	V_{total}	V_1	V_2	V_3
LR	0.74	0.22	0.49	0.42
MLP1	0.73	0.22	0.48	0.41
MLP2	0.80	0.31	0.63	0.57
MLP3	0.79	0.19	0.65	0.54
MLP4	0.79	0.22	0.61	0.46
MLP5	-	0.13	0.64	0.22
GRNN	0.77	0.23	0.66	0.49
Bias				
Model	V_{total}	V_1	V_2	V_3
LR	1.77	3.50	-2.44	0.71
MLP1	2.25	3.75	-2.10	0.67
MLP2	3.69	4.96	0.52	0.54
MLP3	6.61	4.67	2.77	3.87
MLP4	4.72	7.81	1.43	3.39
MLP5	-	7.08	-4.58	-2.49
GRNN	-3.27	-0.93	-6.76	4.41

(continues)