Using aerial platforms in predicting water quality parameters from hyperspectral imaging data with deep neural networks

Taina Hakala*, Ilkka Pölönen*, Eija Honkavaara†, Roope Näsi†, Teemu Hakala†and Antti Lindfors‡

April 3, 2019

Abstract

In near future it is assumable that automated unmanned aerial platforms are coming more common. There are visions that transportation of different goods would be done with large planes, which can handle over 1000 kg payloads. While these planes are used for transportation they could similarly be used for remote sensing applications by adding sensors to the planes. Hyperspectral imagers are one this kind of sensor types. There is need for the efficient methods to interpret hyperspectral data to the wanted water quality parameters. In this work we survey the performance of neural networks in the prediction of water quality parameters from remotely sensed hyperspectral data in freshwater basins. The hyperspectral data consists of 36 bands in the wavelength range of 508 - 878 nm and the water quality parameters to be predicted are temperature, conductivity, turbidity, Secchi depth, blue-green algae, chlorophyll-a, total phosphorus, acidity and dissolved oxygen. The objective of this investigation was to study the behaviour of different types of neural networks with this kind of data. Study is a survey of the operation of neural networks on this problem, which can be used as a basis for the design of a more comprehensive study. The neural network types examined were multilayer perceptron and 1-, 2- and 3-dimensional convolutional neural networks with the effect of scaling the hyperspectral data with standard or min-max -scaler recorded. We also investigated investigated how the prediction of individual water quality parameter depends on whether the neural network model is done solely with respect to this one parameter or with several parameters predicted simultaneously with the same model.The results of the correspondence between the predicted and measured water quality parameters were presented with normalized root mean square error, Pearson correlation coefficient and coefficient of determination. The best models were obtained the 2-dimensional convolutional neural networks with standard scaling made separately for each parameter. The parameters showing good predictability were conductivity, turbidity, Secchi-depth, blue-green algae, chlorophyll-a and total phosphorus, for which the coefficient of determination was at least 0.96 (apart from Secchi-depth even 0.98).

Keywords: remote sensing; hyperspectral; water quality; convolutional neural networks

1 Introduction

In near future it is assumable that automated unmanned aerial platforms are coming more common. There are visions that transportation of different goods would be done with large planes, which can handle over 1000 kg payloads. While these planes are used for transportation they could similarly be used for remote sensing applications. Novel multi- and hyperspectral imagers has quite small weight. Thus, it would be efficient to combine remote sensing missions, where temporal monitoring is needed, to the transport missions. Because amount of gathered data might be relative large in such missions, there is need for the efficient model to interpret desired parameters. In this work we survey the performance of neural networks in the prediction

*Faculty of Information Technology, University of Jyväskylä, Jyväskylä, Finland; taina.v.hakala@gmail.com, ilkka.polonen@jyu.fi

†Finnish Geospatial Research Institute FGI, Kirkkonummi, Finland; (eija.honkavaara, roope.nasi, teemu.hakala)@nls.fi

‡Luode Consulting Ltd, Espoo, Finland; antti.lindfors@luode.net

of water quality parameters from remotely sensed hyperspectral data in freshwater basins.The hyperspectral data consists of 36 bands in the wavelength range of 508 - 878 nm and the water quality parameters to be predicted are temperature, conductivity, turbidity, Secchi depth, blue-green algae, chlorophyll-a, total phosphorus, acidity and dissolved oxygen.

Remotely sensed multi- or hyperspectral data has been used in monitoring the water quality especially in oceanic waters. Probably the simplest method to predict values of a water quality parameter from hyperspectral data is to first find the spectral band that has the best correlation with the given parameter and then make a predictive model between the band and the parameter values using linear regression. More explicit correlations have been found by replacing the simple spectral bands by spectral indices, which are combinations of different spectral bands by arithmetic operations such as quotient, difference or the combination of these (eg. NDVI). However in freshwater basins these methods are not as effective, as the bands or indices correlating with the water quality parameters differ greatly due to varying conditions of the water basins. In an earlier study using the lake Hiidenvesi data [Hakala et al., 2018] it was observed that it was impossible to determine the spectral indices that could be applicable in different parts of the lake. To utilize this method it would be necessary to survey which properties of the lake basin determine the most useful spectral indices.

In the current work the same data from lake Hiidenvesi is used, but instead of studying the correlation of the single spectral bands or few-band indices with the water quality parameters, the whole set of bands is used in the same model by letting a neural network resolve how the water quality parameters depend on them. The benefit of neural networks compared to traditional modelling methods is their ability to solve very complicated relations between the independent and dependent variables. Based on universal approximation theory wide feed forward neural networks are able to approximate any enough smooth functions [Cybenko, 1989].

Even thought basic principle of feed-forward networks have been known from 1940's [McCulloch and Pitts, 1943], it is until now when they have merged as state-of-the-art pattern recognition and machine learning methods. There has been series of incremental steps, which had led to this dramatic evolution. First, back-propagation algorithm was introduced in 1986 [Rumelhart et al., 1986], which made possible to train deeper networks. After 1990's AI winter period, increased computational power in CPU and in GPU machines made training of deep networks with large volume of data. Different regularisation methods were also introduced to overcome issue with overfitting of networks such as Dropout-method [Srivastava et al., 2014]. Increased computational power made possible to adjust also features. One of the most efficient feature extraction algorithms for computer vision currently seems to be convolutional neural networks (CNN).

Real breakthrough of neural networks in a pattern recognition took place 2012 when AlexNet [Krizhevsky et al., 2012] architecture outperformed other conventional approaches in ImageNet classification contest. After that convolutional neural networks have been pushing accuracy limits of detection further. Even thought CNN's are efficiently used for object detection from images, they can also be used to extract features from hyperspectral images and spectra itself. In this work we compare different kind of convolutional neural networks in regression.

In this work we concerned with the basic multilayer perceptron (MLP) and 1-, 2- and 3-dimensional convolutional neural networks (CNN1D, CNN2D and CNN3D). In some cases data preparation has significant effects on the results. This work also investigated the effect of scaling the hyperspectral input-data with either standard or min-max scaling method for each network type. In the standard scaling method (std) the input data is scaled in such a way that for each feature (here the spectral bands) the average and the standard deviation over the training data are 0 and 1 respectively. In the min-max scaling method (mm) the scaling is done such that the minimum and maximum of each band of the training data are 0 and 1 respectively.

When there are multiple water quality parameters to predict, there is the possibility to make a separate model for each parameter or to make a combined model for predicting several parameters at the same time. In this work we have made the separate models for each parameter, a combined model predicting all the parameters and also combined models for two different 6 parameter subsets. The idea behind using differently selected parameter subsets for the combined models is to figure out the effect of the chosen

parameter combination in the prediction of individual parameters.

In an earlier study concerning neural networks and water quality parameters [Chebud et al., 2012] the multispectral (7 band) data from the spaceborne Landsat TM sensor was used to predict the water quality parameters chlorophyll-a, turbidity and total phosphorus in Kissimmee-river (Florida, USA). The neural network used here was a simple 2-layer MLP-type network combined for all the water quality parameters for which the number of neurons in the hidden layer was optimized.

2 Materials and Methods

2.1 Research data

The research material consists of the hyperspectral data and measured water quality parameters from August 2015 in the lake Hiidenvesi in southern Finland. The ecological state of Hiidenvesi is average and its type is eutrophic and naturally turbid with mud. According to paleolimnologic research, Hiidenvesi has been relatively eutrophic even 300 years ago, but in the last 50 years the eutrophication has escalated mainly because of human impact. The area of Hiidenvesi is ca. 30 km 2. [Ranta et al., 2016] Figure 1 shows the shape and location of the lake Hiidenvesi along with the route of the reference measuring boat.

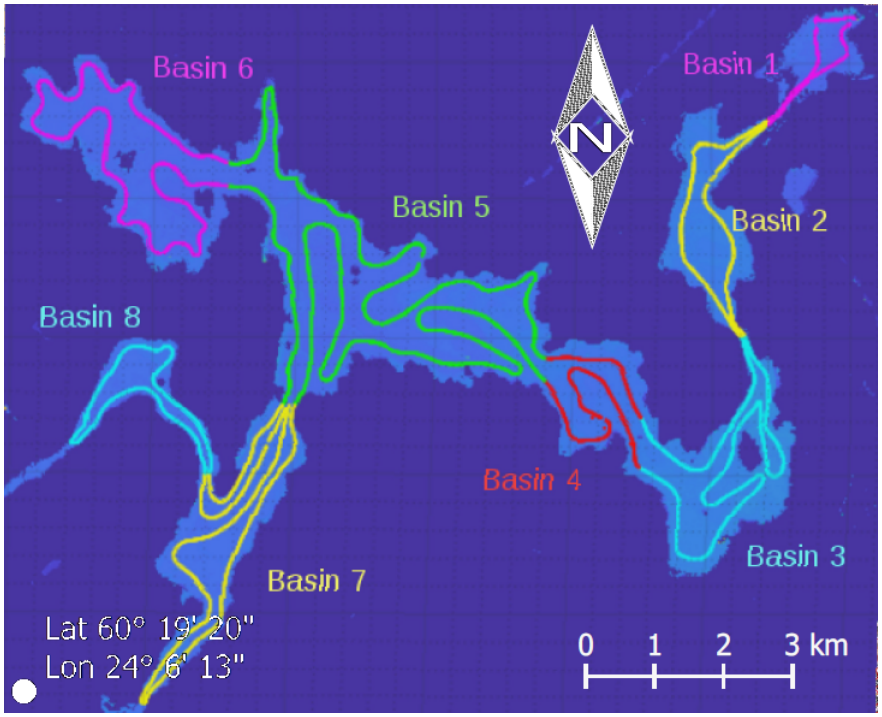


Figure 1: The shape and location of the lake Hiidenvesi. The curve shows the route of the reference measuring boat and the colors indicate the division of the lake into basins used in [Hakala et al., 2018]

Remote sensing datasets were captured using a hyperspectral camera based on Fabry-Perot interferometer (FPI). This imaging technology captures whole frames from one to two separate wavebands at once and provides the hyperspectral spectrum by scanning the spectral range within 2 s time, providing thus the possibility to combine photogrammetric and hyperspectral analysis of data.The 36 hyperspectral bands used here were appoximately equally spaced in the wavelength area 508...878 nm [Hakala, 2018]. Images were captured from a manned single engine aircraft Cessna 172 Reims Rocket aeroplane with an FPI camera using a flight height of about 2025 m above the mean sea level, providing a ground sample distance

(GSD) of 2 m. Image cubes provided by the FPI camera were first laboratory calibrated [Saari et al., 2013]. Image orientations were solved with self-calibrating bundle block adjustment using the VisualFM software [Markelin et al., 2014].

Spectrometric image mosaics were generated using software by Honkavaara [Honkavaara et al., 2013]. Open access aerial laser scanning data by NLS were used as digital surface model in orthomosaic generation. Mosaic was calibrated to the reflectance by least square fitting using field spectrometer, which measured white calibration target and lake water during data collection.

The water quality parameters studied here are (abbreviation, units)

- temperature (Temp, celsius-degree)
- conductivity (Cond, μs/cm)
- turbidity (Turbid, NTU)
- Secchi depth (S-depth, m)
- blue-green algae (BGA, mg/l)
- chlorophyll-a (CHL, μg/l)
- total phosphorus (Ptot, μg/l)
- acidity (pH, pH-scale)
- optically sensed dissolved oxygen (ODO, mg/l)

These parameters were measured simultaneously from a boat using YSI-Multiparameter sonde and S::CAN UV-VIS spectrometer. Measurements were calibrated based on laboratory measurements of water samples.

Each individual series of water quality parameter measurements (ca. 14000 pcs.) has been associated with the series of hyperspectral reflectances determined at the corresponding locations. The outliers of the reflectance data have been removed by hand from each spectral band (these seemed to correspond to a boat, bridge or shore being located in the same pixel of the hyperspectral image).

2.2 Neural networks

Neural networks belong to supervised machine learning, which means that we need to separate the training data to build the model with and the validation data to check how the model works outside the training data [Goodfellow et al., 2016]. Because the validation data can in principle impact the choosing of the best model (ie. the choosing is done in such a way that the results are best for the validation data), it is common to separate yet another set, called the test set, to make the final evaluation of the performance of the chosen model. Here the division of the data into train, validation and test parts is a random split into three equally sized sets (and these sets are the same for all models).

Neural network can be thought of as a function composition with a layered structure. The output of this function is the set of predictions (for the water quality parameter(s)) and the input has two different sets of variables. First set of variables is the set of features on which the prediction relies (the spectral band reflectances) and the second is the set of learnable parameters which are adjusted in such a way that the average error between the the output (predictions obtained for a given set of features) and the target values over the training data is as small as possible. Only after the optimizing the learnable parameters the network is ready to predict outputs for new inputs (validation data) [Goodfellow et al., 2016].

The essential factor of the network performance is the structure, number and order of its layers, which correspond to the components of the function composition. In this work the most relevant layer types are dense and convolution. Dense layer is the most basic layer type and it handles all the features in the same way. Concisely described, dense layer is an affine function between the spaces spanned by the neurons of

two consecutive layers, followed by a nonlinear activation function. The learnable parameters from a dense layer are the entries of the matrix of the affine function and their number is essentially the product of the neuron numbers in the corresponding consecutive layers, which can be quite large.

The idea of a convolution layer is to preserve the structure of the feature space, such as the adjacent band reflectances or the neighboring pixels of an image. For instance, with 2-dimensional image data the image is not converted to a vector (as is the case for the affine map of a dense layer), but is instead kept as a 2D-array and image-like attributes are searched using filters. Filter is a small 2D-array which scans through the image area (possible color channels regarded as overlapping layers) and by calculating inner products at its stopping points yields a new (often smaller) 2D-array. A single convolution layer usually holds several filters of the same size, each of which looks for a certain attribute in the image. The output of the convolution layer is the set of 2D-arrays from all filters, stacked on top of each other to be handled as the color channels in the next layer. Here the learnable parameters are the entries of the filters and they are considerably fewer than in a dense layer of the same size.

Nevertheless, even for convolutional neural networks the total number of learnable parameters can easily rise to tens or hundreds of thousands (see Figure 1 below). This gives neural network models flexibility, which is superior compared to traditional models. To compare with the spectral index method, there are essentially only 3 optimizeable parameters: the choosing of the best correlating spectral index and the parameters (eg. slope a and y-intercept b) of the best fitting line in the scatter figure of the index vs water quality parameter, obtained by linear regression. In this case the predictive function would simply map a sequence of spectral band reflectances to a*(best index)+b, where 'best index' is the value of the predetermined best correlating spectral index at the given sequence of spectral band reflectances.

When a single data item is not an image but a sequence of numbers (as the series of 36 hyperspectral band reflectances), the most natural convolution type is 1-dimensional. In a 1D-convolution there is only one image dimension and also the filters are 1-dimensional. 3-dimensional convolutions are formed analogously, considering the 'image' and the filters as 3D-arrays.

The dimension of the convolution does not have to be the natural dimension of the independent variable, which is 1 for the band reflectance series and 2 for images. The input-data can in principle be shaped into any dimension, as long as the number of features (length of the band series or the pixel number in an image) is high enough to cover all the dimensions adequately. One of the aims of this work is to compare the results when intrinsically 1-dimensional spectral data is handled with not only the natural network types MLP and CNN1D, but also with CNN2D and CNN3D.

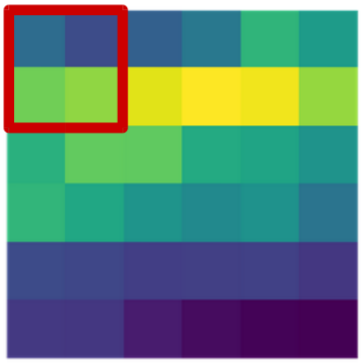
For the 2D convolution the 36 features (spectral bands) have been arranged to 2D array of shape 6×6 and similarly for the 3D convolution the shape $3 \times 3 \times 4$ is used. These shapes have been chosen here for simplicity and balance between the dimensions, but it should be noted that the lengths of the array sides are usually not unique (e.g. $36 = 2 \times 18 = 3 \times 12 = 4 \times 9$) and also the ordering of the features may be varied.

Figure 2 illustrates the shaping of the data features and the operation of convolution in different dimensions. Figure 2a shows the spectral part of one data item as a 1D sequence with the red highlight showing one position for a 2-sided filter. Figure 2b shows the spectral part of the same data item as a 2D array of shape 6×6 and again one position of a 2-sided filter is highlighted. In Figure 2c the same data point is shown as a 3D array of shape $3 \times 3 \times 4$ with different layers of the depth dimension shown in different plane figures. Also here one position of a 2-sided filter (consisting of 8 volume pixels) is highlighted with red.

Figure 2 also demonstrates one issue in comparing convolutional networks of different dimensions. First it is difficult to make the filter structure for different dimensions commensurate as for example the relative size of a 2-sided filter with respect to the feature array increases significantly going from 1 dimension to 3, which impacts the effectiveness of the convolution. Second point is the mutual relation of the features, as for 1D there are at most 2 neighbors to each feature, whereas for 3D there can be 26 neighbors. Another problem is the question of how many convolutional layers can be fitted in a network in each dimension. A filter with sides greater than 1 causes the sides of the image array to reduce in the next layer, so for an input array with shorter sides (such as the 3d array of size 3*3*4) the image degenerates with too many layers. The reducing of the CNN3D network used here is explained in detail in section 2.3 and Figure 1 below.

The challenging part of neural networks is the difficulty to optimize the network for each problem.

(a) Spectrum in 1 dimension



(b) Spectrum in 2 dimensions

(c) Spectrum in 3 dimensions (with third dimension presented as distinct plane figures)

Figure 2: The reflectance spectrum of a single data point arranged in (a) 1d-, (b) 2d- and (c) 3d-arrays for the utilization of convolutional neural networks of the corresponding dimensions. The red highlights indicate the shape and size of 2-sided filters in various cases.

Optimizing means adjusting the networks numerous hyperparameters, which include the number, type and ordering of the layers, activation functions, regularization, weight initialization, learning rate etc. in such a way that the performance of the network for the current task is as good as possible. Here the role of hyperparameters is taken by the network type (MLP, CNN1D, CNN2D, CNN3D) and its variation with respect to normalization (none, std, mm) and the set of water quality parameters modeled at the same time (single parameter models, allparams, 6params, 6params1). However thorough hyperparameter optimization was not carried out for the individual models, but instead the default hyperparameters of Keras were used. This implies that the performances of different types of networks are not fully comparable. Although the best results here are given by the CNN2D-type networks, it is possible for example that the MLP-type network could be optimized to a performance level higher than what can be achieved for CNN2D.

Thus the aim of this work is not to find the best possible neural network for the problem of predicting water quality parameters from hyperspectral data. Instead it is a survey of the operation of neural networks on this problem, which can be used as a basis for the design of a more comprehensive study.

2.3 Network types

The representatives of the network types multilayer perceptron and 1-, 2- and 3-dimensional convolutional neural networks used here are specified in Figure 1. The design of the networks is partly based on trial and error to get rather simple models with reasonably good performance. The structure of different network types is basically similar with the effective layers (dense for MLP and convolutions of given dimension for the CNNs) at the same positions. For the CNN-networks some MaxPooling and Dropout layers have been added and the last layers are the same for all networks.

In the CNN3D-network the dimensions of the input data (3,3,4) were so small that it was necessary to remove one MaxPooling and convolution layer in order to keep the sides of the neuron layer box positive. It could have been possible to add a convolution layer with filter size (1,1,1), which does not decrease the sides of the neuron layer box, but it was noticed that the results were clearly worse with this addition than by simply omitting the layer.

Table 1: Details of the network types used in this study. For the 2- and 3-dimensional convolutional networks also the shape of the input data is mentioned, i.e. CNN2D66 and CNN3D334 mean that the input shapes of the networks are 6*6 and 3*3*4 respectively. The table text is essentially Keras code with some simplifications. For example Dense(64, ReLU) means dense layer with 64 neurons and rectified linear unit as the activation function. Similarly for example Conv2D(32,(2,2) ReLU) means 2D convolutional neural network with 32 filters (the output depth or 'channels') of spatial dimension 2*2 and the activation as before.

MLP	**CNN1D**	**CNN2D66**	**CNN3D334**
Dense(64,ReLU)	Conv1D(32,(2),ReLU)	Conv2D(32,(2,2),ReLU)	Conv3D(32,(2,2,2),ReLU)
Dense(64,ReLU)	Conv1D(32,(2),ReLU)	Conv2D(32,(2,2),ReLU)	Conv3D(32,(2,2,2),ReLU)
	MaxPooling1D((2))	MaxPooling2D((2,2))	
	Dropout(0.25)	Dropout(0.25)	Dropout(0.25)
Dense(64,ReLU)	Conv1D(32,(2),ReLU)	Conv2D(32,(2,2),ReLU)	
	Dropout(0.25)	Dropout(0.25)	
Flatten()	Flatten()	Flatten()	Flatten()
Dense(32)	Dense(32)	Dense(32)	Dense(32)
Dense(16)	Dense(16)	Dense(16)	Dense(16)
Dropout(0.25)	Dropout(0.25)	Dropout(0.25)	Dropout(0.25)
Dense(x)	Dense(x)	Dense(x)	Dense(x)
Params 83000	Params 39000	Params 15000	Params 11000

In the last layer Dense(x) of each network type the figure x specifies the number of final outputs of the network, which is determined by the number of water quality parameters being modeled simultaneously. Thus for the single parameter models x=1, for all parameter model x=9 and for the two different 6 parameter models x=6. The last row of each column indicates the approximate number of learnable parameters for each network type (which varies slightly with the number x of the water quality parameters being modeled). It can be seen that the number of learnable parameters decreases significantly when going from MLP to CNN1D and further to CNN2D.

The compilation of the model also requires the specification of the loss-function and the optimizer. The loss-function is the measure of the predictive error at the training stage of the model and here it is defined as the mean squared error (mse). Optimizer is an algorithm used to enhance the finding of the optimal learnable parameters to minimize the loss-function. Here the optimizer Adam is used throughout.

The software used in this work to handle the neural networks are Python (version 3.3.6), Keras (version 2.2.0) and Tensorflow (version 1.9.0).

2.4 Scaling methods

The performance of neural networks can often be improved by scaling the independent variables (features). In this work along with the unscaled models, two different scaling methods were tested. In the standard normalization method (std) the spectral part of the train-data is centered and scaled so that for each band the mean and the standard deviation are 0 and 1 respectively. The it is calculated so that

$$\mathbf{x}_{std} = \frac{\mathbf{x} - \mu}{\sigma},$$

where $\mathbf{x}$ is a recorded spectrum, μ is average and σ is standard deviation of the training spectra. The spectral parts of the validation and test data are scaled using the scaling parameters from the train data, so that the corresponding mean and standard deviation for each band are close to 0 and 1 (but not exactly, as the details of the data parts differ a little).

Analogously in the min-max normalization method (mm) the spectral part of the train-data is moved and scaled so that for each band the minimum and the maximum values are 0 and 1 respectively and again the validation and test data are scaled using the scaling parameters from the train set. The min-max normalization is calculated following

$$\mathbf{x}_{mm} = \frac{\mathbf{x} - \mathbf{x}_{\min}}{\mathbf{x}_{\max} - \mathbf{x}_{\min}},$$

where $\mathbf{x}_{\min}$ and $\mathbf{x}_{\max}$ are minimum and maximum vectors over the spectral bands of training data set.

The effect of scaling the spectral data is demonstrated in Figure 3, where the color expresses the strength of the reflectance, the vertical axis of each sub-figure shows the 36 hyperspectral bands and the horizontal axes show the corresponding measuring points. (Thus the data corresponding to the spectrum (or 'image') of one measuring point is shown in one column of a sub-figure.) Figure 3a shows the whole data unshuffled and unscaled. It can be noted that the reflectance values of the bands 8...10 are stronger throughout the data, with especially high values around the points 1000...3000 and 12500..14000 along the measuring path (these parts actually correspond to the same areas in the lake as the route of the measuring boat was a closed loop). Figure 3b shows the unscaled spectral part of the randomly chosen train data. This figure is rather homogeneous along the horizontal axis (as the points are randomly mixed), but the strength of the bands 8...10 is still showing. In figures 3c and 3d the train data is shown with the standard and min-max scaling respectively. In these last two figures the differences in the brightness of the bands longer stand out. The two scaling methods naturally result in different ranges of the values, but it can also be noted that in the min-max-method the colors differ more extensively, which means that a larger part of the data is located close the endpoints of the scale.

2.5 Configuration of the modelled water quality parameters

As there are several water quality parameters to be predicted, the models can be built either separately for each parameter or as a combined model for a subset of the parameters. The combined models have been built for the whole 9 parameter set (abbreviated allparams), for the 6 parameter set {Turbid, S-depth, BGA, CHL, Ptot, pH} (6params) and for a slightly different 6 parameter set {Cond, Turbid, S-depth,BGA, CHL, Ptot} (6params1).

The idea behind using two different 6 parameter sets was to examine how the performance of the combined model is affected by the constituent parameters performance in the single parameters models. The parameter set 6params1 was assembled from those parameters that had the best results in the single parameter models, whereas the set 6params included the parameter pH which was not very well approximated by the single parameter model.

2.6 Metrics

The comparison of the final models trained with the training data is carried out by using each model to predict the water quality parameter values from the spectra in the validation set and comparing these to the actual measured values of the water quality parameters. The comparison is done with the following three metrics, each of which expresses the model performance from a different point of view:

- NRMSE = normalized root mean square error
- pearson = pearson correlation coefficient
- r2 = coefficient of determination

Here the metric NRMSE is defined as the square root of the mean of the squared errors of the predicted vs correct values, divided by the range of the correct values (i.e. the root mean square error divided by the target range). NRMSE is an error metric, which means that a smaller number corresponds to a better model. The reason for normalizing the error is to reduce the effect of different scales of distinct water quality parameters to obtain more congruent error estimation.

The metric pearson is a number between 0 and 1 describing the amount of linear dependence between the predicted and the measured values of the water quality parameters. Pearson is a score metric which means that in principle a greater value is better. However, in this context a high pearson correlation does not necessarily mean a good model, since it only implies that the points of the scatter plot of predicted vs measured values are close to some straight line. If this line is other than the main diagonal, the predictions are somehow distorted from the target values and the model is not good (see Figure 4 below).

The coefficient of determination r2 has a few different definitions. The one used here is the same as in Scikit-learn and is descibed as providing a measure of how well future samples are likely to be predicted by the model (http://scikit-learn.org/stable/modules/model_evaluation.html#r2-score). Also r2 is a score metric whose highest and best value is 1. R2 can also have negative values which indicate that the prediction is worse than what would be obtained by using the expected value (mean) of the target variables. The coefficient of determination is difficult to describe, but it seems to be closely related to the points of the predicted vs measured -scatter figure being close to the main diagonal.

2.7 Learning and adjusting the models

The optimizable number of epochs was adjusted for each model. Epochs is a number that tells how many times the whole train data has been used in adjusting the learnable parameters of the neural network. When training the network with Keras, the loss value (the error between the predicted and correct values) of both the train and validation data is recorded for each epoch-value. Usually both of these losses are large for the first epochs but decrease quickly to a level determined by the performance of the model.

When the model is unstable, increasing the epochs number causes the validation-loss to begin to rise at some point while the train-loss continues to decrease. This is caused by overfitting, which means that the model starts to memorize the training data, but no longer works so well with the slightly different validation data. In this case the best possible model for the validation data is obtained when the learning is stopped at the epoch-value where the validation-loss obtains its minimum. The starting point of the adjusting of the epochs-value was 2000 for all the models. If there was a clear minimum of the validation loss, the model was trained again with the epoch-value corresponding to this minimum. If no minimum was found (i.e. the validation-loss curve continued to decrease) the 2000 epochs model was accepted as the final model, or sometimes if the decreasing at epoch's to 2000 still seemed significant the model was trained again with even larger epoch value.

The unstable models for which the training was cut before epoch number 2000 were mainly the 1D-convolutional single parameter models made with unscaled spectral data (CNN1D_none_x -series). Figure 4 shows the behavior of the unscaled CNN1D-model of the water quality parameter CHL made with epoch numbers 2000 and 250. Figure 4a shows the train and validation losses at epoch values 0...2000 and it shows that the minimum of validation loss is achieved around epoch values 250...500. Figure 4c shows the comparison of the predicted and measured values of CHL obtained with this non-optimal model. Figure 4b and 4d show the same model with the training stopped at the epoch value 250 before the model starts overfitting. Comparing the two predicted vs measured -scatter figures 4c and 4d it can be seen that in the latter case of epochs=250 the points are closer to the main diagonal going from lower left to upper right corner, which means a better correspondence of the predictions with the actual values.

In the scatter Figures 4c and 4d also the performance metrics of the models are shown. We see that the metrics 'NRMSE' and 'r2' are better for the model with epochs=250, but the metric 'pearson' is slightly worse. This is actually typical behavior: when the model is overfitted (trained for too long), the pearson metric usually gets better, although the performance of the model (i.e. the location of the scatter points close to the diagonal and the metrics NRMSE and r2 describing this) gets worse. Also the predicted vs measured -scatter for any overfitted model resembles Figure 4c in that the line which is close to the scatter points is too high at the end meaning that predictions are systematically smaller than the actual values.

In reality the loss-curves of the unstable models vary greatly from one execution to another, probably due to different initializations of the network. Sometimes the loss curves for the model CNN1D_none_chl were greater than 40 for all the epochs and also the performance metrics were naturally worse. For the comparison in figure 4 the executions were chosen in such a way that the minimum value for the loss was

approximately at the same level (around the value 20) to see more clearly the effect of overfitting on the predicted vs measured -scatters.

The other models tested were clearly more stable, which means that the shape of the validation loss curve is similar to shape of reciprocal function. If the negative slope seemed to be very close to zero around the epoch-value 2000, this was directly made as the final model. If on the other hand the decreasing of the loss at epoch=2000 still seemed reasonably fast, the same model was executed with a larger epoch-value usually causing the performance metrics to improve a little. Epochs-values greater than 4000 were not tested as increasing the epochs-value directly increases the time required for the training and also the improvement of the performance of the models seemed to be very subtle.

In contrast to the single parameter models, the loss-curve of a combined model is shared for all the parameters involved (the sets allparams, 6params or 6params1). For these models the behavior of the loss curve seemed very stable, so it was reasonable to make the epochs-value as large as possible (i.e. 4000). In practice the increasing of the epochs-value often resulted the performance of the model to improve for some parameters and deteriorate for others. However the changes were so small that they may well be caused by randomness. Proper hyperparameter optimization was not carried out for the individual models, because it seemed that results were good even without it.

3 Results

The results of comparing the models using the validation data have been presented in a table for each of the metrics NRMSE, pearson and r2 separately, see Figures 5, 6 and 7. The naming of the models in the tables is done with the formula

$$\textbf{(network type)\_(scaling method)\_(configuration of parameters)} \tag{1}$$

where

- network type $\in$ {MLP, CNN1D, CNN2D66, CNN3D334}
- scaling method $\in$ {none, std, mm}
- configuration of parameters $\in$ {x, allparams, 6params, 6params1}

In the place of the configuration of parameters the symbol x (the first 12 rows of each table) means single parameter model, where the parameter is given by the column. In this part of the table not only the rows but also the columns depict distinct models, as a different model is trained for each water quality parameter. In the rest of the table (configurations allparams, 6params, 6params1) the same model is used in every column of a row, since the model gives results for every parameter included in the configuration.

Note that NRMSE is an error metric (smaller is better), while pearson and r2 are score metrics (larger is better), but the coloring in each table is chosen in such a way that lighter cells indicate better model performance.

From the tables 5, 6 and 7 it can be seen that although the general appearances of the tables for different metrics are rather similar, there are some cases where the metrics disagree. For example the parameters Temp and Cond in the allparams-model group: according to NRMSE Temp performs better with the none-scaled models, while according to pearson or r2 the results for Cond are clearly better. This is why the determination of a single best model is difficult.

The first observation may be that the water quality parameters can be categorized according to the general functionality of hyperspectral methods in the predictions. Good performance results are obtained for the parameters Turbid, S-depth, BGA, CHL and Ptot, and partly also for the parameter Cond. For the parameters Temp, pH and ODO the results are clearly worse. This categorization is probably based on the parameters nature; some are more easily detected by optical methods than others.

The tables show that generally the best performing models are the single parameter models (the first 12 rows whose name end with x). However there are some exceptions as for the parameters S-depth and

Ptot the performance of the combined models seems to be just as good or even slightly better than of the single parameter models. On the other hand for the parameters Temp, Cond, pH and ODO the results of the combined models are significantly worse than those of the single parameter models.

Scaling (especially std) improves the results most clearly for the CNN-type models but has no significant effect with the MLP-type. In many cases the MLP gives better results than the none-scaled CNNs, but after scaling the results of CNNs are usually better.

If each parameter configuration (single parameters (x), allparams, 6params, 6params1) is inspected separately it can be seen that generally (considering all parameters simultaneously) the best model is CNN2D_std. Also the models CNN2D_mm, CNN3D_std and CNN3D_mm perform almost as well. Again for individual parameters there are exceptions: for BGA in the combined models (allparams, 6params, 6params1) the best models are usually MLP_std or MLP_mm.

When comparing the differently configurated combined models (allparams, 6params ja 6params1) for the applicable parameters, no obvious differences can be found. This means that the hypothesis of the parameters performing well in the single parameter models Cond, Turbid, S-depth, BGA, CHL and Ptot (i.e. the set 6params1) would make a better combined model has no evidence. However this is a very constricted survey on this topic and further studies are needed to make reliable conclusions.

The difference in the performance between the single parameter models and the combined models can be visualized with scatter figures 8 and 9, where for each parameter the relation between the predictions and the correct (measured) values are shown. For the combined models only the allparams-configuration is presented as the corresponding figures for applicable parameters of the 6params- and 6params1-configurations are rather similar. For both the single parameter models and the allparams-model the model variant showing on average the best performance is CNN2D66_std, so these are the models used in this comparison.

In the single parameter models depicted in Figure 8 the scatter points are generally closer to the diagonal indicating a better correspondence between the predicted and measured values. There is some parameter-dependent variation on the correlation (the closeness of the points to the diagonal), but in all cases the points are located rather symmetrically around the diagonal. In contrast for the allparams-model shown in Figure 9 there is generally much more asymmetry and deviation from the diagonal. Some parameters (eg. S-depth, CHL, Ptot) are still predicted quite well, but others (especially Temp, Cond, pH and ODO) show irregular patterns that are not near the diagonal. Moreover for a fixed water quality parameter these irregular patterns are somewhat similar regardless of the network type, scaling or the parameter configuration of the combined model.

Determining a single best model (hyperparameter combination) is difficult because several metrics were recorded and their results do not always coincide. Also the results of a single model change a little from execution to another, so that the tables 5, 6 and 7 are not completely fixed. However when considering all the water quality parameters at once, one could say that on average the best performing model is the set of CNN2D single parameter models with standard scaling (CNN2D66_std_x).

In the table of Figure 2 the results of the model CNN2D66_std_x are shown evaluated with the test-data (the 'clean' dataset, which was not used in training nor in the choosing of the best model from the metrics tables). The hypothesis when evaluating the best model with the test-data is that the results are a little worse than the corresponding results for validation-data, since in principle the choosing of the best model involved the characteristics of the validation-data. Comparing the results of Figure 2 to the rows CNN2D66_std_x of the metrics tables it can be seen that the results indeed are mostly the same or a little worse, with the exceptions for NRMSE to S-depth and pearson to Cond, where the results are slightly better (likely due to randomness).

4 Discussion

This study revealed/verified the different potentials of various water quality parameters to be predicted with hyperspectral data. It seems natural that the 'visible' parameter Turbid and the ones possessing a good correlation with it (S-depth (negative correlation), BGA, CHL and Ptot) are easier to predict with optical methods. On the other hand the parameters Temp, pH and ODO with no optical properties were

Table 2: Evaluation of the best model CNN2D66_std_x on the test-data

	Temp	**Cond**	**Turbid**	**S-depth**	**BGA**	**CHL**	**PTot**	**pH**	**ODO**
NMRSE	0.06	0.03	0.04	0.05	0.03	0.03	0.04	0.06	0.06
Pearson corr.	0.94	1.00	0.99	0.99	0.99	0.99	0.99	0.95	0.94
$\mathbf{r^2}$	0.82	0.99	0.98	0.96	0.99	0.98	0.98	0.86	0.85

much harder to predict. The non-visible parameter Cond showed rather contradictory behavior, since it had generally good results with the single parameter models but poor results with the combined models.

On average the best models turned out to be the single parameter models of type CNN2D with standard scaling. Especially compared with the basic model MLP of neural networks, the CNN2D had not only better performance, but also significantly less learnable parameters, which is an advantage. Somewhat special is that the input-data used here is not exactly of the kind that 2d-convolutional neural networks are usually recommended to use with.

2d-convolutional networks are most often used when the input-data is in image form (e.g. image recognition). The basic property of image-data, besides 2-dimensionality, is that the objects of interest are certain image elements, which may be found in different parts of the image array (e.g. the shape of a cats ear may be located in any part of the image). The 2d-convolutional neural networks are thought to be suited especially well for this kind of task, since as the filters scan through the image, the image elements are found equally well in any parts of the feature space ('translational invariance'), at the same time keeping the number of learnable parameters reasonable. Spectral data however does not have this kind of 'image elements', but instead (in the unscaled data) the high values are found mostly at the same bands (corresponding to the same element being at the same location in most of the images). Another difference between image- and spectral data is that the 'meaning'/significance of a spectral series remains the same when the bands are consistently arranged to a different order, whereas for image data the interpretation becomes impossible after rearranging the pixels.

Thus, even though spectral data does not have the properties typically addressed with the use of convolutional neural networks, and MLP-type could be generally thought of as a 'stronger' network type, the convolution seems to perform better. This demonstrates the ability of convolutional neural networks to adapt to different problems and could courage their usage in situations for which they are not usually recommended.

In the work reported by [Chebud et al., 2012] the r2-score between the predicted and the measured values to be greater than 0.95. In the current study the r2-scores between the predicted and the correct values varied greatly with the water quality parameter and the model. Here the best r2-results for the corresponding parameters Turbid, CHL and Ptot given by the single parameter models of type CNN2D with the standard scaling were 0.98. In the combined models (which here included more parameters than in the mentioned previous research) the results were 0.92-0.93 for the parameter Turbid and 0.96-0.97 for the parameters CHL and Ptot.

The results of this study are not directly comparable to those of [Chebud et al., 2012], as the number of spectral bands and water quality parameters is greater. Also the focus of this work is in comparing the behaviour of different types of neural networks and their variations, leaving a minor role to the optimization of the network performance. This work demonstrates the flexibility of convolutional neural networks to adapt to problems that are not in the traditional scope of the topic. Despite of the spectral reflectance series (used as features) being intrinsically 1-dimensional and not fulfilling the 'translational invariance' -property usually associated to convolutions, the best performing models turned out to be 2- and 3-dimensional convolutional neural networks.

The remote sensing data capture was made using a 2D frame format hyperspectral camera weighing less than 700 g. The sensor was operated onboard a small single engine manned aircraft in order to cover the large area. The same technology can be operated onboard unmanned aerial vehicles (UAV) as well. For example, in previous studies it has been used in agricultural [Honkavaara et al., 2013], forestry [Nevalainen et al., 2017,

Näsi et al., 2016] and water quality studies [Pölönen et al., 2014]. It is expected that in the near future the legislation and UAV technology will develop and enable operation beyond visual line of sight. This will enable cost-efficient and flexible data capture over larger water areas, which will be highly interesting in water studies.

It is also interesting that among the convolutional neural networks of different dimensions the most natural 1d-type was not the best one, but instead the artificially constructed 2d-type performed better. Also the performance of the 3d-type was almost as good as that of the 2d-type. In this context the functionality of the 3d-type may have been reduced by the number of features (36) being too small, forcing the 3d-CNN to be shallower than the 2d-version. The better performance of the higher dimensional convolutions may be related to the features being more closely connected (number of neighbors of a feature is 2 for 1d, 8 for 2d and 26 for 3d).

It was noted that the best performing models were usually the single parameter models. One could presume this to be related to having a larger number of features (bands) per parameter to be predicted. On the other hand this would not be consistent with the observation that reducing the number of parameters in the configuration of a combined model (from allparams to 6params or 6params1) did not improve the results for the applicable parameters. It is also peculiar that for some parameters (Temp, Ph, ODO and especially Cond) the results for the combined models were significantly worse that those of the single parameter models, whereas for other parameters (S-depth, Ptot) there was practically no difference. By comparing the results for the parameter configurations 6params and 6params1, no evidence was found for the assumption that the performance of the combined models individual parameters performance in the single parameter models would have an effect in the performance of the combined model (the set 6params1 consisted solely of the parameters showing good performance in the single parameter models, whereas in the set 6params there was also other parameters). However this one example is not sufficient for proper conclusions.

It turned out that for the parameters Temp, Cond, Ph and ODO showing poor performance in the combined models, the predicted vs measured -scatter points formed patterns which are distinctive for different parameters. These patterns remained essentially the same regardless of the model type or the parameter configuration of the combined model. It seems that in the combined models predicting several parameters at once there is some internal transaction between the parameters and to solve this transaction further studies are needed.

Direction for further studies could be to examine the effect on the performance of a combined model of the parameter configurations size and the internal predictability (the performance of the single parameter models) or the mutual correlation of the parameters. This could be done by first studying various 2 parameter configurations and then increasing the configuration size gradually.

5 Conclusion

Monitoring of water quality parameters is essential in the environmental sciences and community planning. These tasks can be executed by the hyperspectral remote sensing. In this study we demonstrated how aerially collected remote sensing data can be transferred to the quality parameters. These methods can be used while aerial vehicles are transporting goods. We have compared four different types of deep neural networks predicting water quality parameters from hyperspectral imagery. Our results show that there are relevant choices to make when deep neural networks are used for this. Both data normalization and neural network's structure has impact to the results. In the network's structure it is shown that more complex feature learning scheme and model, which learn one parameter at the time gives better results. Despite of the good results it would be good to have more data sets from from fresh water areas, when data is gathered from a different lakes and different time of the year. This would made evaluation of the models even more reliable.

Acknowledge This research has been co-funded by Finnish Funding Agency for Innovation Tekes (grants 2208/31/2013 and 1711/31/2016).

References

[Chebud et al., 2012] Chebud, Y., Naja, G. M., Rivero, R. G., and Melesse, A. M. (2012). Water quality monitoring using remote sensing and an artificial neural network. *Water, Air, & Soil Pollution*, 223(8):4875–4887.

[Cybenko, 1989] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.

[Hakala et al., 2018] Hakala, T., Pölönen, I., Honkavaara, E., Näsi, R., Hakala, T., and Lindfors, A. (2018). Variability of remote sensing spectral indices in boreal lake basins. In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume 42. International Society for Photogrammetry and Remote Sensing.

[Hakala, 2018] Hakala, T. V. (2018). Supplementary figures for the article 'variability of remote sensing spectral indices in boreal lake basins'.

[Honkavaara et al., 2013] Honkavaara, E., Saari, H., Kaivosoja, J., Pölönen, I., Hakala, T., Litkey, P., Mäkynen, J., and Pesonen, L. (2013). Processing and assessment of spectrometric, stereoscopic imagery collected using a lightweight uav spectral camera for precision agriculture. *Remote Sensing*, 5(10):5006–5039.

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

[Markelin et al., 2014] Markelin, L., Honkavaara, E., Näsi, R., Nurminen, K., and Hakala, T. (2014). Geometric processing workflow for vertical and oblique hyperspectral frame images collected using uav. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*.

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

[Näsi et al., 2016] Näsi, R., Honkavaara, E., Tuominen, S., Saari, H., Pölönen, I., Hakala, T., Viljanen, N., Soukkamäki, J., Näkki, I., Ojanen, H., et al. (2016). Uas based tree species identification using the novel fpi based hyperspectral cameras in visible, nir and swir spectral ranges. *International archives of the photogrammetry, remote sensing and spatial information sciences; Volume XLI-B1*.

[Nevalainen et al., 2017] Nevalainen, O., Honkavaara, E., Tuominen, S., Viljanen, N., Hakala, T., Yu, X., Hyyppä, J., Saari, H., Pölönen, I., Imai, N. N., et al. (2017). Individual tree detection and classification with uav-based photogrammetric point clouds and hyperspectral imaging. *Remote Sensing*, 9(3):185.

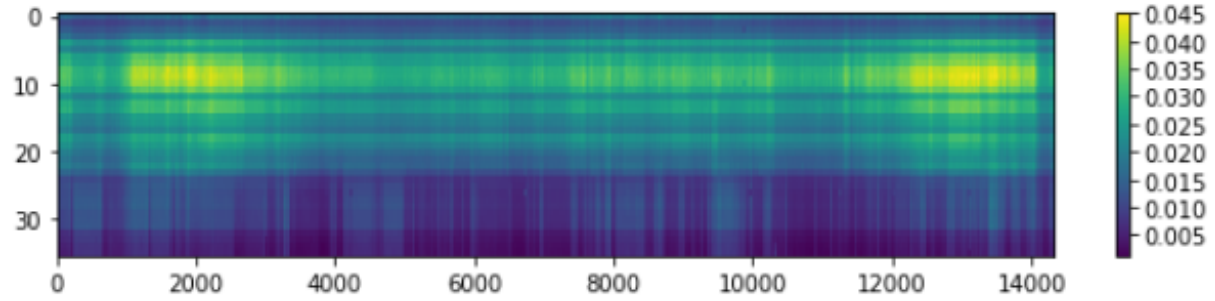
[Pölönen et al., 2014] Pölönen, I., Puupponen, H.-H., Honkavaara, E., Lindfors, A., Saari, H., Markelin, L., Hakala, T., and Nurminen, K. (2014). Uav-based hyperspectral monitoring of small freshwater area. In *Remote Sensing for Agriculture, Ecosystems, and Hydrology XVI*, volume 9239, page 923912. International Society for Optics and Photonics.

[Ranta et al., 2016] Ranta, E., Valtonen, M., and Ikonen, E. (2016). Hiidenveden alueen yhteistarkkailun yhteenveto vuodelta 2015. *Länsi-Uudenmaan vesi ja ympäristö ry. Käsikirjoitus*.

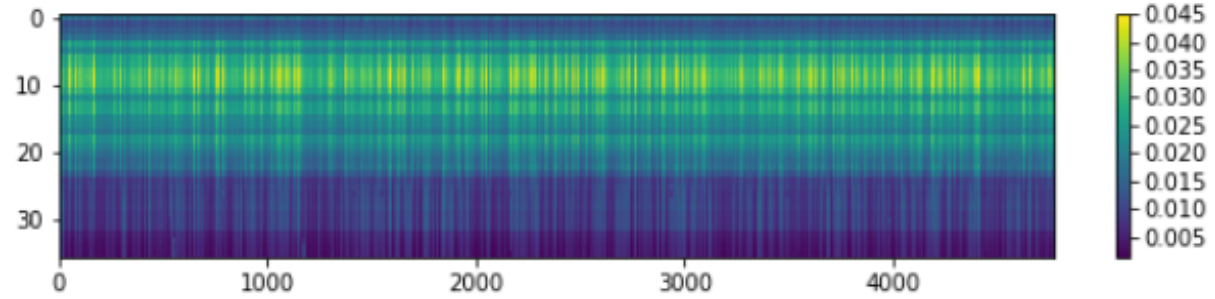
[Rumelhart et al., 1986] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533.

[Saari et al., 2013] Saari, H., Pölönen, I., Salo, H., Honkavaara, E., Hakala, T., Holmlund, C., Mäkynen, J., Mannila, R., Antila, T., and Akujärvi, A. (2013). Miniaturized hyperspectral imager calibration and uav flight campaigns. In *Sensors, Systems, and Next-Generation Satellites XVII*, volume 8889, page 88891O. International Society for Optics and Photonics.

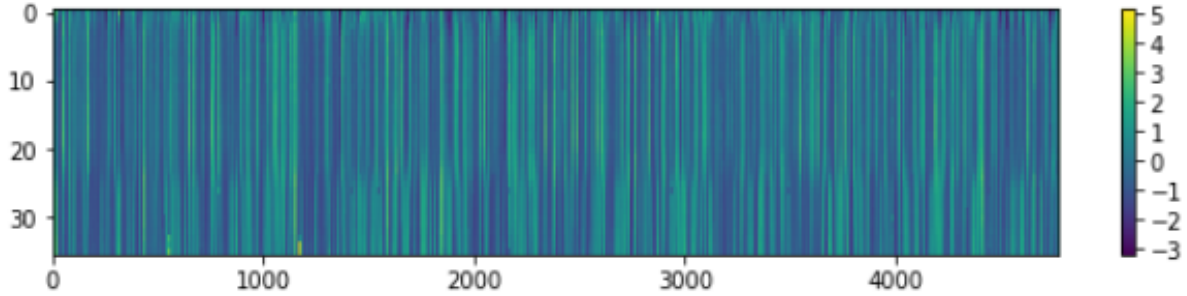
[Srivastava et al., 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.



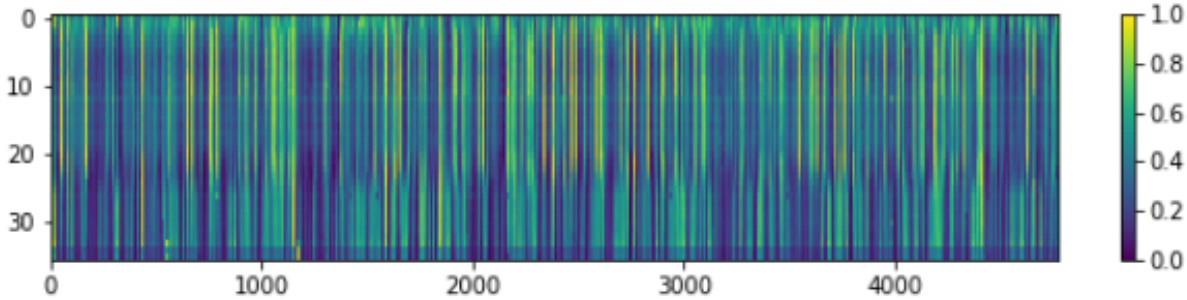
(a) Spectra of whole data



(b) Spectra of unscaled train data

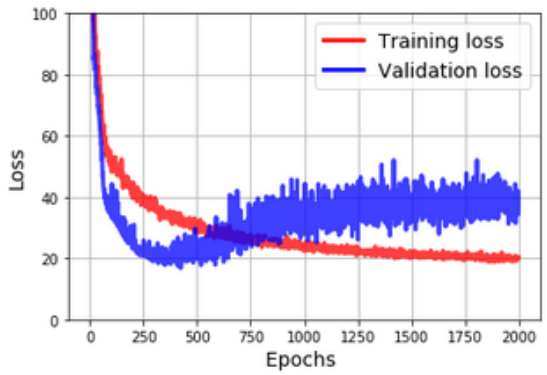


(c) Spectra of standard-scaled train data

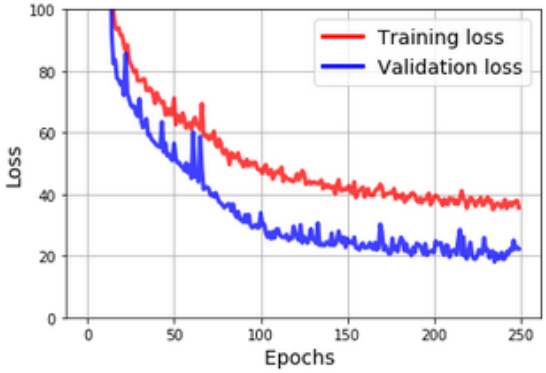


(d) Spectra of min-max -scaled train data

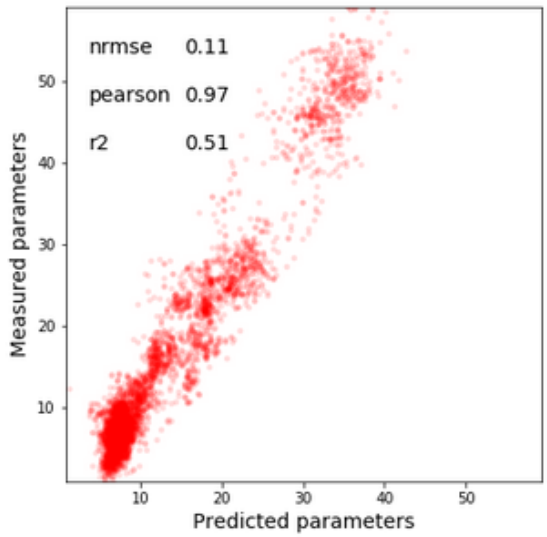
Figure 3: Demonstration of the effects of scaling the spectral part of the data. In (a) the whole unshuffled and unscaled data is shown. In the rest of the subfigures the randomly selected and shuffled train-data is shown (b) with no scaling, (c) with the standard scaling and (d) with the min-max scaling.



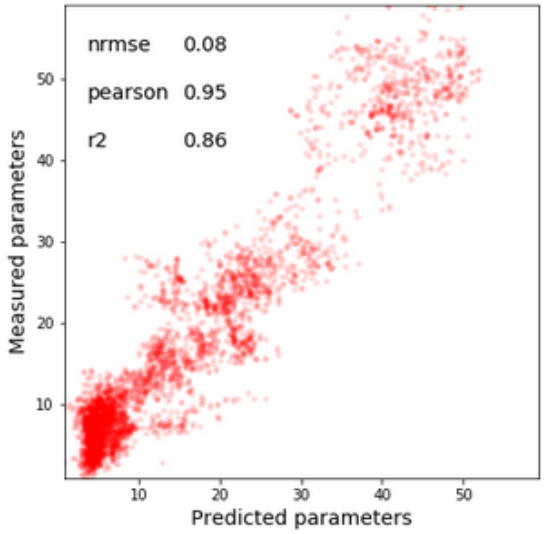
(a) Loss for epochs=2000



(b) Loss for epochs=250



(c) Scatter for epochs=2000



(d) Scatter for epochs=250

Figure 4: The effect of changing the epochs value in the training of the none-scaled single parameter CNN1D-network for CHL. (a) and (b) show the loss-curves for the models trained with 2000 or 250 epochs respectively. (c) and (d) depict the predicted vs measured -scatter figures for the corresponding models evaluated with the validation data, along with the performance metrics values. In the optimal epochs=250 case the performance of the model is generally better (points of the scatter figure closer to the diagonal), although the metric pearson is slightly worse.

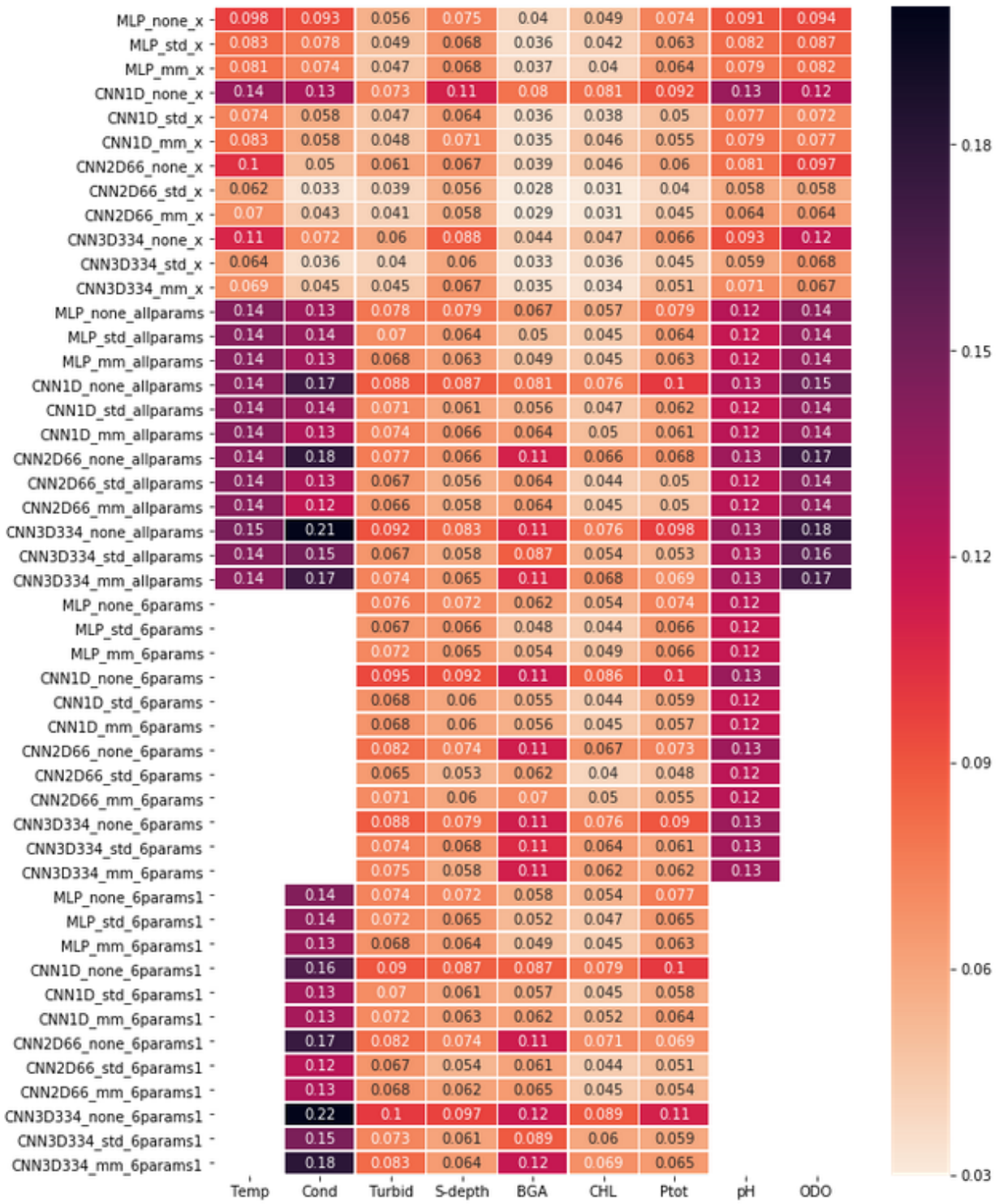


Figure 5: NRMSE:s between predicted and measured parameter values for different models

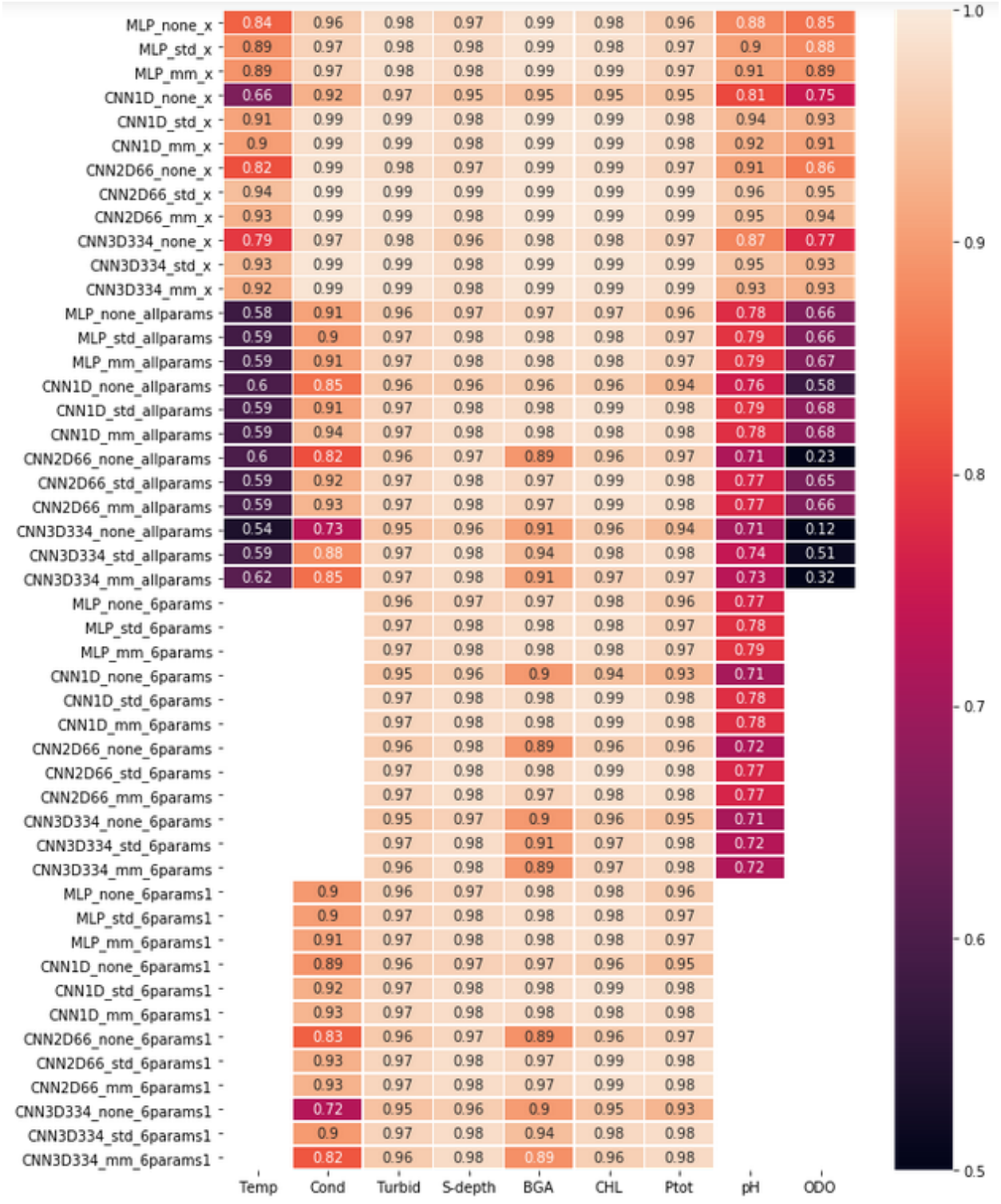


Figure 6: Pearson correlations between predicted and measured parameter values for different models

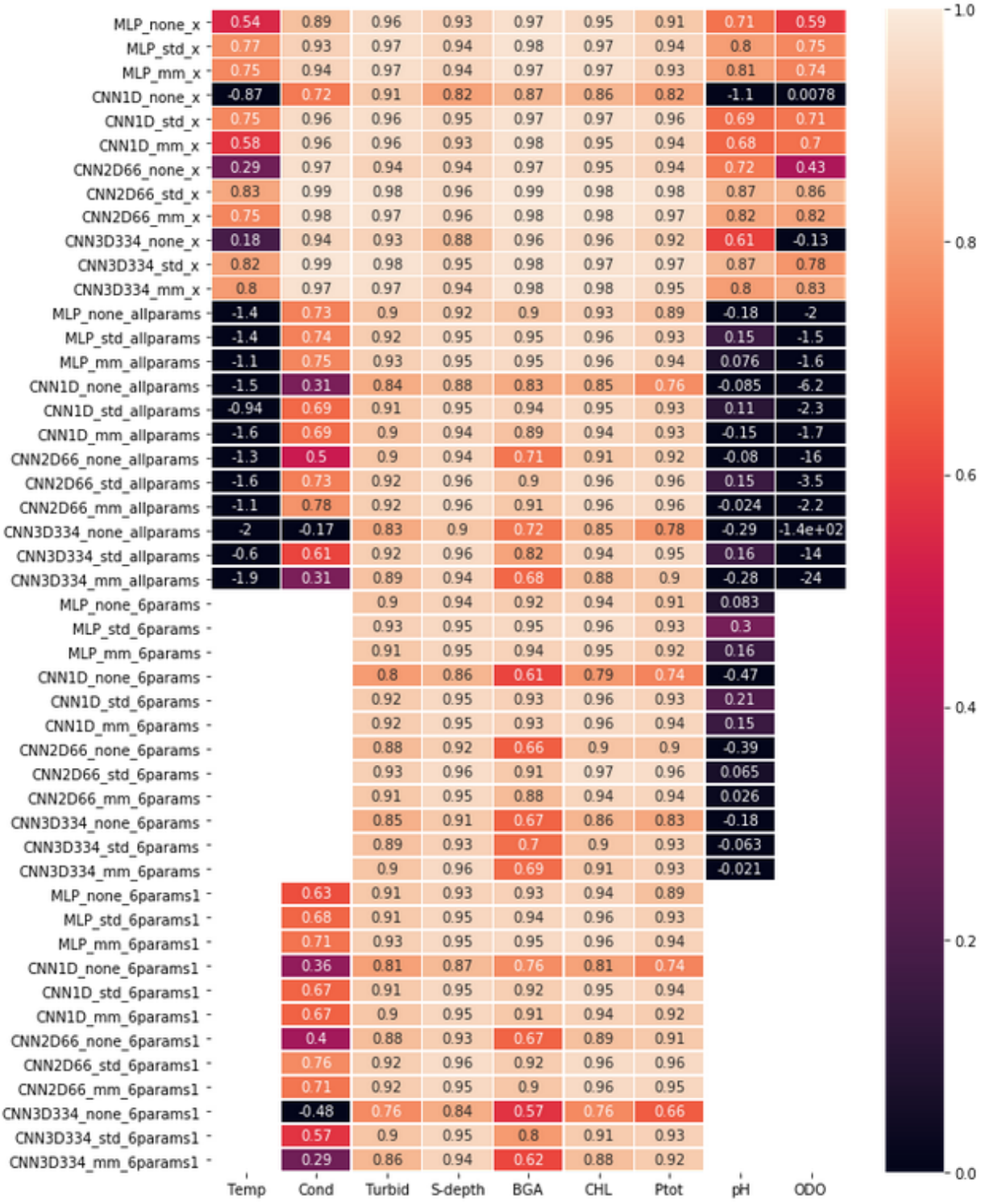


Figure 7: R2-scores between predicted and measured parameter values for different models

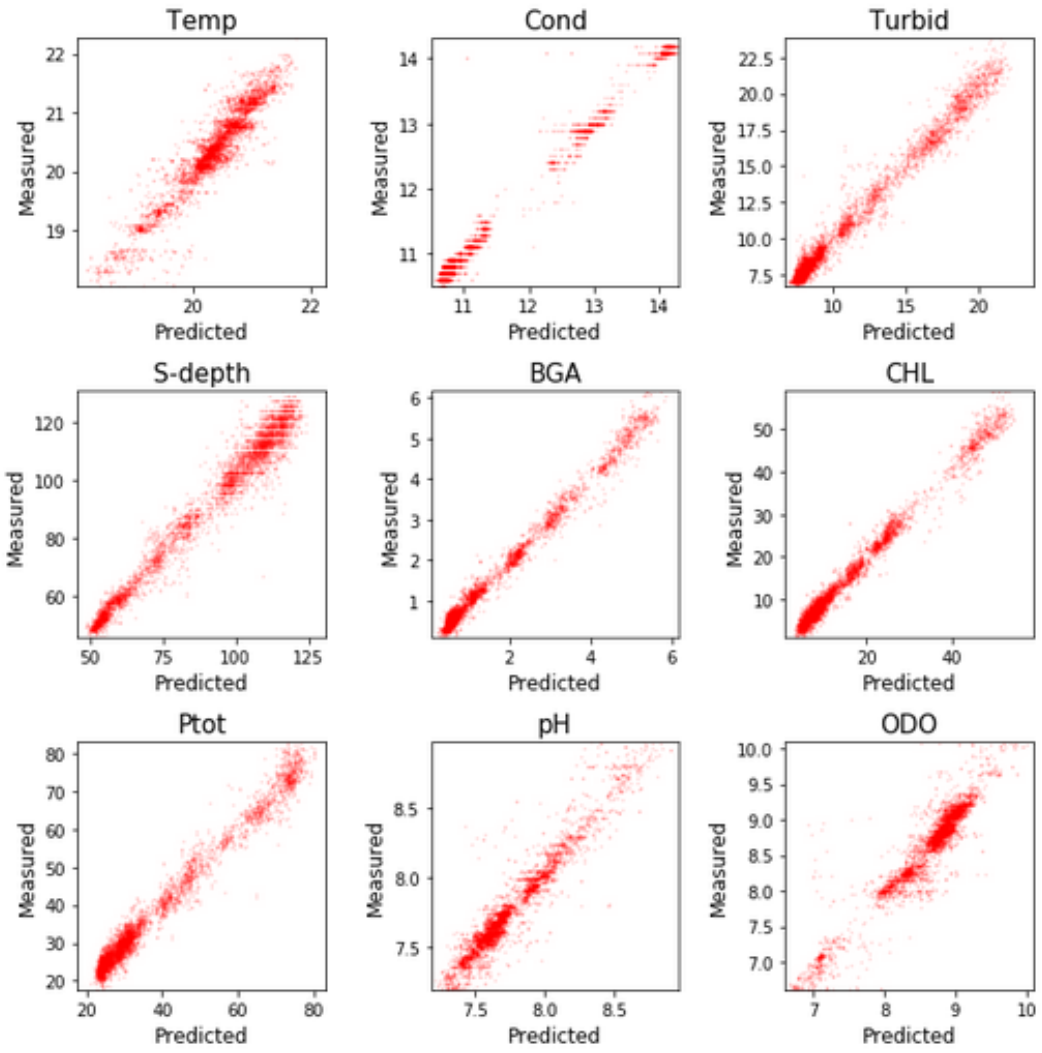


Figure 8: Predictions vs measurements -scatter figures for the single parameter models CNN2D66_std_x. Here the scatter points for all the water quality parameters are located fairly symmetrically around the main diagonal.

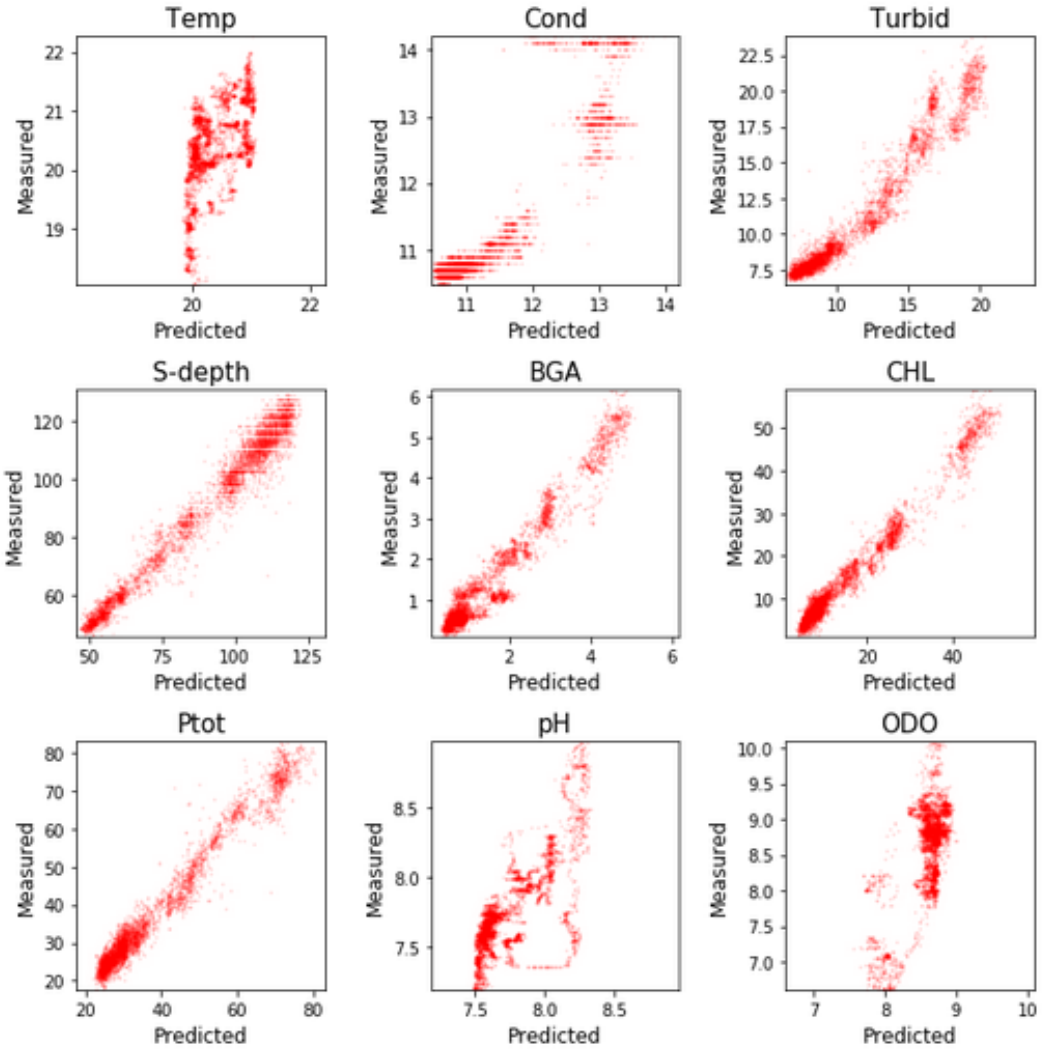


Figure 9: Predictions vs measurements -scatter figures for the combined allparams model CNN2D66_std_allparams. Here the scatter points for especially the parameters Temp, Cond, pH and ODO deviate greatly from the diagonal (compared to Figure 8). The patterns formed by the scatter points seem to be characteristic to the water quality parameters and show only slight variation with network type, scaling or the parameter configuration of the combined model.