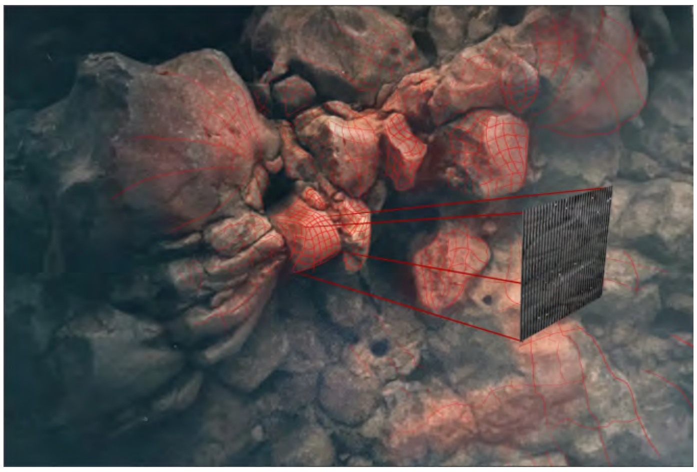
Professur für Hydrologie

der Albert-Ludwigs-Universität Freiburg i. Br.

Nadja Veigel

Using Machine Learning to Quantify the Strength of Weathering at Carbonate Rock Landscapes



Masterarbeit unter Leitung von

Prof. Dr. Andreas Hartmann

Freiburg i. Br., Juli 2020

Professur für Hydrologie

der Albert-Ludwigs-Universität Freiburg i. Br.

Nadja Veigel

Using Machine Learning to Quantify the Strength of Weathering at Carbonate Rock Landscapes

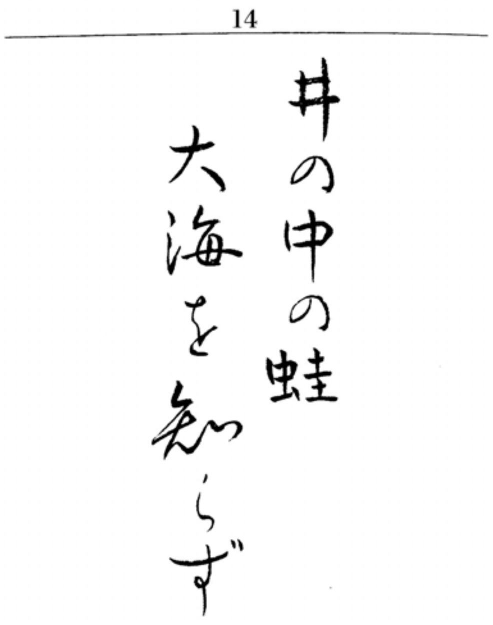
Referent: Prof. Dr. Andreas Hartmann

Korreferent: Dr. Michael Stölzle

Freiburg i. Br., Juli 2020

Preface

Thanks to Dr. Benjamin Mewes for the support and supervision.



14

井の中の蛙
大海を知らず

Der Frosch im Brunnen ahnt nichts von der Weite des Meeres

Summary

The topographic variability in karst regions is thought to be different than in non-carbonate regions. Deep learning approaches were used on digital elevation data to distinguish whether karst occurs in a certain area or not. Quantifying the strength of weathering in karst regions was the second objective of this work. With this information regions with unreliable carbonate rock estimates could be mapped more reliably. A digital elevation model with a 1 arcsec resolution covering the majority of Europe was used to train a convolutional neural network, as well as a Quantile Regression forest, along with the karst regions set in the world karst aquifer map. Similarly a convolutional neural network and a quantile regression forest were trained on a smaller scale Digital Terrain model covering Slovenia, which is a highly karstified country. Simulations and predictions of the convolutional neural network were insufficient. A sensitivity analysis was conducted increasing the topographic parameters by 5, 10 and 20%. The sensitivity analysis showed that even a 5% increase of terrain parameters raised training accuracy from 0.64 to 0.79. With a 20% increase the convolutional neural network predicted the occurrence of karst in 98% of cases correctly. It was possible to classify topographically distinct regions as karstified with a quantile regression forest, although accuracy measures over the whole dataset didn't surpass the random assignment of karst regions given the proportion of their extent. Considering the high resolution data for training the convolutional neural network, the accuracy achieved was higher. A maximum testing accuracy of 0.96 and 0.92 along with maximum training accuracy of 0.82 and 0.80 for different loss functions was found. At visual inspection of the predictions of these models, the high accuracy was not representable for the model's ability to predict karst in a suficcient way, caused by large class imbalances that not even a more robust loss function could overcome. Finally the quantile regression forest trained with the high resolution data showed similar results to the convolutional neural network. Reported accuracy was high, but the predictions didn't outperform a random assignment significantly. For the data with higher resolution the convolutional neural network was able to identify features and relate them to the ocurrence of karst. Possible reasons of failure to make reliable predictions are (1) very limited computing power (2) choice of preprocessing (3) high abstraction level of the data.

Keywords:
Deep Learning, Machine Learning, Convolutional Neural Network, Quantile Regression Forest, Karstification, Karst, Carbonate rock, Digital Elevation Model

Zusammenfassung

Es wird angenommen, dass die topographische Variabilität in Karstgebieten anders ist als in Nicht-Karbonatgebieten. In dieser Arbeit wurden Deep-Learning-Ansätze für digitale Höhendaten verwendet, um zu unterscheiden, ob in einem bestimmten Gebiet Karst vorkommt oder nicht. Die Quantifizierung der Stärke der Verwitterung in Karstregionen war das Ziel des zweiten Schritts dieser Arbeit. Mit diesen Informationen konnten Regionen mit unzuverlässigen Karbonatgesteinsschätzungen zuverlässiger kartiert werden. Ein digitales Höhenmodell mit einer Auflösung von 1 arcsec, das den größten Teil Europas abdeckt, wurde verwendet, um ein Convolutional Neural Network sowie einen Quantile Regression Forest zusammen mit den Karstregionen der weltweiten Karstaquiferkarte zu trainieren. In ähnlicher Weise wurden ein Convolutional Neural Network und ein Quantile Regression Forest auf ein digitales Geländemodell in kleinerem Maßstab trainiert, welches das stark verkarstete Land Slovenien abdeckt. Die Simulationen und Vorhersagen des Convolutional Neural Network waren unzureichend. Es wurde eine Sensitivitätsanalyse durchgeführt, bei der die topographischen Parameter um 5, 10 und 20% erhöht wurden. Die Sensitivitätsanalyse zeigte, dass eine Erhöhung der Parameter um 5% zu einer Verbesserung der Trainingsgenauigkeit von 0,64 auf 0,79 führte. Bei einer Erhöhung um 20% sagte das Convolutional Neural Network das Auftreten von Karst in 98% der Fälle korrekt vorher. Es war möglich mit einem Quantile Regression Forest verkarstete Regionen zu klassifizieren, obwohl die Genauigkeitsmaße über den gesamten Datensatz die zufällige Zuordnung von Karstregionen angesichts des Anteils ihrer Ausdehnung, nicht übertraf. In Anbetracht der hochauflösenden Daten für das Training des konvolutionären neuronalen Netzes war die erreichte Genauigkeit höher. Es konnte eine maximale Testgenauigkeit von 0,96 und 0,92 zusammen mit einer maximalen Trainingsgenauigkeit von 0,82 und 0,80 für verschiedene Loss Function erreicht werden. Eine visuelle Analyse der Modellvorhersagen zeigte, dass die hohe Genauigkeit der Modelle nicht ihre Fähigkeit Karst in ausreichender Weise vorherzusagen repräsentiert. Dem zugrunde liegen große Klassenungleichgewichte, welche nicht einmal eine robustere Loss-Function überwinden konnte. Schließlich zeigte der mit den hochauflösenden Daten trainierte Quantile Regression Forest ähnliche Ergebnisse wie das Convolutional Neural Network. Die berichtete Genauigkeit war hoch, aber die Vorhersagen übertrafen eine Zufallszuweisung nicht signifikant. Bei den Daten mit höherer Auflösung war das Convolutional Neural Network in der Lage, Merkmale zu identifizieren und sie mit dem Auftreten von Karst in Beziehung zu setzen. Mögliche Gründe für das Scheitern zuverlässiger Vorhersagen sind (1) sehr begrenzte Rechenleistung (2) falsche Wahl der Vorverarbeitung (3) hoher Abstraktionsgrad der Daten.

Stichworte:
Deep Learning, Machine Learning, Convolutional Neural Network, Quantile Regression Forest, Verkarstung, Karst, Karbonatgestein, Digitales Geländemodell

Contents

List of Figures

List of Tables

List of Figures in the Appendix

Acronyms

CNN Convolutional Neural Network

DEM Digital elevation model

D Dice coefficient loss function

DL Deep Learning

DTM Digital terrain model

FCN Fully Convolutional Network

g Activation function

GPU Graphics processing unit

H Crossentropy loss

Lidar light detection and ranging

ML Machine Learning

MLP Deep multilayer perceptron

QRF Quantile regression forest

ReLU Rectified Linear Units activation function

RGB Red, green, blue ; referring to the bands of multispectral images

SRTM Shuttle Radar Topography Mission

TPI Topographic Position Index

TRI Terrain Ruggedness Index

TWI Topographic Wetness Index

w weight, trainable parameter in a neural network

w_0 bias, trainable parameter in a neural network

$\mathbf{x}$	input to a neuron, observation
$\mathbf{x'}$	rescaled observation
$\mathbf{y}$	output of a neuron
y_p	predicted output of the model
y_t	expected output of the model

1. Introduction

The first scientific descriptions of karst landscapes and the term "karst" originated from inhabitants of the Dinaric karst (Bonacci, 1987). The historical importance of karst, according to Parise and Sammarco (2015), is manifested in the circumstance that inhabitants of karstified areas were forced to develop deeper understanding of the underlying geologic structures and develop techniques to distribute water. This factor was leading to a main driving force in the development of many karstified regions. Nowadays karst is of great economic and hydrologic importance since approximately 15 % of global drinking water is supplied by karst aquifers (Parise and Sammarco, 2015). The entanglement of karst aquifers in combination with its high relevance for water supply remain some of the lesser understood and addressed topics in hydrology (Bonacci, 1987).

The Evolution of karst, more precisely the physical and chemical processes relevant in the evolution of karst, is called karstification. Karst is prone to being highly impacted by weathering and erosion. Karst rocks (evaporites: gypsum, rock salt, anhydrite; limestone: marble, dolomite) share the property of being soluble in water, therefore erosion due to dissolving is their major type of weathering. The close relationship between karst and hydrology becomes immediately clear (Gunn, 2004). One of the main characteristics of karst landscapes is the absence of surface waters. With an increasing degree of karstification the application of standard hydrologic models and concepts cannot be used as an accurate description of hydrologic processes in affected regions (Bonacci, 1987). This issue has been addressed by researchers who developed catchment scale models to simulate hydrological processes in karst areas (Parise et al., 2018). Thus the disregard of karst occurrence may result in a large amount of error in prediction of river discharge or spring discharge as well as other estimates based on models of any scale.

Machine learning (ML) is an emerging field in applied various disciplines including Hydrology (Shen et al., 2018). Rosenblatt (1958) was trying to understand and recreate biological systems and neurological information processing. He developed the Perceptron, the key element of neural networks. Since his discovery Rumelhart et al. (1986) introduced backpropagation as an imitation of the natural learning processes for networks of perceptrons and in 1995 the first neural network specialized on computer vision problems was proposed (Bengio et al., 1995). Despite the early discovery of these relatively simple model structures it is only today that their full potential can be realized. Besides new ML techniques the main reason for the recent success of neural networks is the availability of Big Data (Barnes, 2013) and increasing computing power. In fact, Krizhevsky et al. (2012) claimed that their results on image recognition tasks are solely limited by the availability of memory on their two graphics processing units (GPU's) and the size of training datasets. Even tough this statement is likely to contain some truth, Xie et al. (2019) for example showed that ML advances don't depend exclusively on technical advances. On one hand their research was conducted with more advanced processors. On the other hand the newly introduced self-learning technique contributed majorly to the advance in performance of their model on the ImageNet task.

> "Deep learning (DL) is a suite of tools centered on artfully designed large-size artificial neural networks." Shen et al. (2018)

The depth of a neural network is connected with its amount of layers. Deep learning is a branch of artificial intelligence that shows the greatest capacity to extract relevant information from large datasets (Shen et al., 2018). The deep structure makes the prediction of non-linear relationships between input images and labels given to each pixel possible. The task of labeling a pixel of each image with a certain class is called semantic segmentation in Data science. It can be applied successfully to remote sensing images (Zhu et al., 2017). The main focus of this work are supervised deep learning networks, namely convolutional neural networks (CNN), which have proven to be very successful in image segmentation tasks (Long et al., 2015).

Inferring karst areas and quantifying the weathering of carbonate rock landscapes is crucial to improve hydrologic models on a large (possibly global) scale and identify false model behavior in smaller scale models due to karstification. Simultaneously by training a machine learning algorithm to identify karst regions reverse conclusion on which topological factors are the most influential to define karst areas, is possible. This thesis represents an intersection between deep learning, geomorphometry and remote sensing with focus on applying the acquired knowledge and data to hydrologic models and understanding of hydrologic behavior of karstified regions.

CNNs have been applied to geodata. A brief summary of such applications can be found in Zhu et al. (2017) and will be elaborated further in the following literature survey. There are several different related fields of application of deep learning on spatial data. Most of the projects are realized using satellite images. The remote sensing tasks that have been successfully automated using DL are:

- Identifying above-ground structures
- Digital soil mapping
- Identifying land cover and crop types
- Natural disasters (landslides and sinkholes)
- Karst rocky desertification

Using digital elevation models (DEM) and DL to identify above-ground structures with focus on determining trees and buildings in urban areas (Marmanis et al., 2015). In this study a deep multilayer perceptron algorithm (MLP) was chosen over the more complex and statistically promising CNN due to sufficient classification skills of the simple algorithm and the ability to quickly process large amounts of remotely sensed data. They also state that other DL architectures are likely to result in better predictions.

Another field of application for DL in remote sensing are land cover and soil mapping tasks (Heung et al., 2016). A study conducting digital soil mapping tasks using a CNN (Padarian et al., 2019) and a study to determine soil moisture distribution with an MLP (Song et al., 2016) have been executed. In both cases the DL Models outperformed conventional models in calculation time and accuracy. Padarian et al. (2019) used a 3 arcsec resolution DEM and derived slope and topographic wetness index along with annual temperature and rainfall data. All data was standardized to a 100m grid. The closest related study to the following project is an attempt to map superficial geology in northern Canada (Latifovic et al., 2018). The authors used RGB imagery from Landsat data and derived information about the landcover e.g. normalized difference vegetation index from it. Additionally they included air photos and a DEM with 8 m spatial resolution and superficial material categorized by experts to train the CNN. Other studies have shown that deep learning (a multi layer feed-forward artificial neural network) methods can outperform other methods (gaussian pyramid and random forest) results on digital soil mapping tasks with the input parameters being derived from DEM information prior to training (Behrens et al., 2018).

Wurm et al. (2019) demonstrated that a fully convolutional network (FCN) can be used to map slums in mega cities like Mumbai, India. The authors used satellite images from different sources and with different resolutions to map areas that are likely to be a part of a slum. With the highest resolution satellite images (QuickBird) Wurm et al. (2019) were able to achieve a 88.39% accuracy for detecting slum pixels.

A CNN and a MLP were compared to segment and predict crop types and land covers in Ukraine based on Satellite (Landsat-8 and Sentinel-1A RS) images (Kussul et al., 2017). The ensemble CNNs trained on the specific data outperformed the MLP and was able to identify major crop types with an accuracy of 85 %.

A CNN has been used in natural hazard management to detect landslides in Shenzen area, China (Ding et al., 2016). The authors suggest to use texture change and a

specially trained CNN to detect areas where landslides occurred. Similar applications of deep learning on hazards happening in carbonate rock landscapes, which are mainly collapses have been studied using a CNN to forecast sinkholes (Hoai et al., 2019) and Bayes-based machine learning algorithms to map current sinkholes (Taheri et al., 2019). Identification of underground cavities with ground-penetrating radar was successfully realized using a CNN in Seoul, South Korea (Kang et al., 2019).
Support vector machines have been used to identify karst related geomorphological features using extensive data like karst rocky desertification and multi-spectral satellite images (Xu et al., 2015).
Interestingly CNNs have been applied to identify and label pixels in magnetic resonance imaging pictures to detect brain tumors (Havaei et al., 2017). The kind of brain tumor the authors examined can appear at any location of the brain and have numerous shapes, but still the CNN was able to segment it with higher accuracy and in a shorter time than the models trained on the same problem before that. This study is of interest to this project because the magnetic resonance images are an abstraction of the brain as DEMs are an abstraction of the earths surface. While most of the described studies utilize satellite images in combination with other data that are as good as photos of the earths surface, Havaei et al. (2017) show the ability of CNNs to extract patterns from more abstract data.

The main focus of this survey is on applying a CNN to identify karst areas. In addition to this a quantile regression forest (QRF) was set up and evaluated as a promising method to identify karstified areas. Random forests were introduced by Breiman (2001) and they have been used successfully for various regression and classification tasks. A review of their application can be found in Tyralis et al. (2019). A further description of the application of QRFs in remote sensing and other classification tasks exceeds the capacity of this work and is not carried out at this point.

Artugyan and Urdea (2016) conducted a survey they claim to be the first description of a karstified mining area in South-West Romania. They found that the terrain parameters extracted from the DEM are related to the areas grade of karstification, which they related to the occurrence of dolines. Slope gradient, drainage depth, drainage density, topographic wetness index (TWI) and topographic position index (TPI) were calculated to quantify the stage of karstification. Low slope gradients indicate strong karstification due to slow drainage of the areas, which was validated by matching sinkhole occurence and slope gradient. Drainage depth represents the range of slopes in a certain area leading to a representation of plateaus. This index showed a positive correlation to karstification. Low Drainage density is considered one of the main characteristics for karst areas (Bonacci, 1987). Similarily Marjan Temovski and Ivica Milevski (2015) conducted a classical geomorphometric research on karst areas in Macedonia using a DEM of 15 m resolution to characterize karst areas. The authors found that the elevation was higher, slope was significantly higher and aspect was slightly more prone to being eastern and western exposed for karst areas, comparing the distributions of parameters to the rest of the countries geomorphometric properties.

2. Objective

It seems that using only DEMs to predict the strength of weathering in a way that is unrelated to a specific karst phenomenum, for example sinkholes, is a very novel approach that has not been done in this manner before. Geomorphometric analysis suggest that the approach is promising (Artugyan and Urdea, 2016; Marjan Temovski and Ivica Milevski, 2015). According to Artugyan and Urdea (2016) the information if karstification occurs in certain regions and to which extent it is present must be contained in a DEM and therefore the current CNNs should be able to "see" and structure the data in a way to make this information accessible (Long et al., 2015). The Objective of this work is to use a machine learning algorithm to segment DEM images into areas affected by karstification and non-karstified areas and quantify the grade of karstification. The following research questions are to be answered during this project:

1. Can a CNN or a QRF be trained to predict the boundaries of karst areas on the basis of elevation data with an adequate accuracy?
2. Is the strength of weathering derivable from this data as well?
3. Can deep learning be used to improve karst mapping globally?

Thus the Null Hypothesis of this thesis is:

Deep learning can improve the current state of mapping carbonate rock regions.

In which the current state of mapping refers to the extent of the carbonate rock regions as well as the strength of weathering present in those regions.

3. Methods

The steps taken in order to model the occurrence of karst are summarized in the flowchart in figure 3.1. A number of topographic parameters were derived from the original elevation data. The dataset was explored using traditional statistical methods and a binomial regression was performed to get a first idea of influential variables. Slope and elevation were classified and related to the occurrence of karst. Afterwards a CNN was set up and trained on the data and three augmented datasets with an artificially increased input signal for detection. Furthermore the CNN was trained using different objective functions. Another CNN with was trained using data with a higher resolution to examine the influence of resolution. Finally a quantile regression forest was trained as a second approach to predict karstified areas for both datasets. All models and datasets required individual preprocessing.

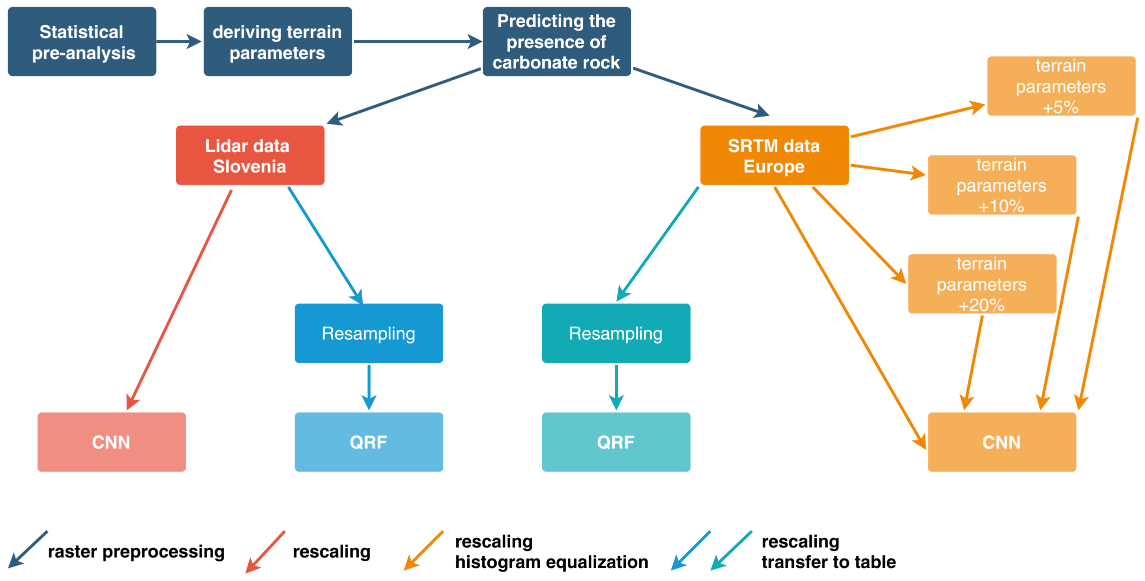


Figure 3.1: This flowchart shows the different ML models trained in this work and the datasets used. Additionally the links between the boxes represent the steps taken to process data or train models. The usage of different loss functions on the SRTM data was emitted for simplicity.

3.1 Data

Karst areas are mapped very reliably for countries in Europe and parts of North America by Chen et al. (2017) in the corresponding datasets. Those karst regions were assumed to be the ground-truth masks for the models. The DEM provided by the Space Shuttle Radar Topography Mission (SRTM) (Shuttle Radar Topography Mission, 2000) was used as the first predictor. On this mission the Earth's topography was covered with a $1arcsec$ resolution, which comes to approximately 30 meters for 80% of the Earth surface. In the second set of models a high resolution Digital Terrain Model (DTM) with a resolution of $1m^2$ of Slovenia was included (Lidar, 2010). The data was aquired using light detection and ranging (Lidar), which is a remote sensing technique that uses lasers of different wavelengths to measure the state of atmospheric and earth surface parameters. The DTM of Slovenia is freely available and can be downloaded from the government website (Lidar, 2010).
The SRTM data was provided as raster file and the world karst aquifer map (WOKAM) data was available as ESRI shapefile. The Lidar data was available as compressed point clouds in .laz format (Lidar, 2010).

3.2 Preprocessing

SRTM

Processing of the SRTM data was done in R (R Development Core Team, 2008), QGis (QGIS Development Team, 2009) and Python (Travis Oliphant, 2006–; GDAL/OGR contributors, 2019). The SRTM DGM was split up into tiles of 128 x 128 pixels (figure 3.2, left) resulting in images spanning an area of $14,75$ km^2. The tile size was approximated to match the extent of the pictures Wurm et al. (2019) used to successfully identify slums, although they used pictures with a higher resolution, as well as the images used to identify road scenes by Badrinarayanan et al. (2015) (100 x 100 pixels).

The dataset containing regions of karst areas were converted from shapes to raster files with the same extent and resolution as the SRTM data, and then cropped to the same extent as the DEM tiles in Python. Further processing and rescaling into binary images was implemented in R. Splitting of the European SRTM raster took several days to compute.

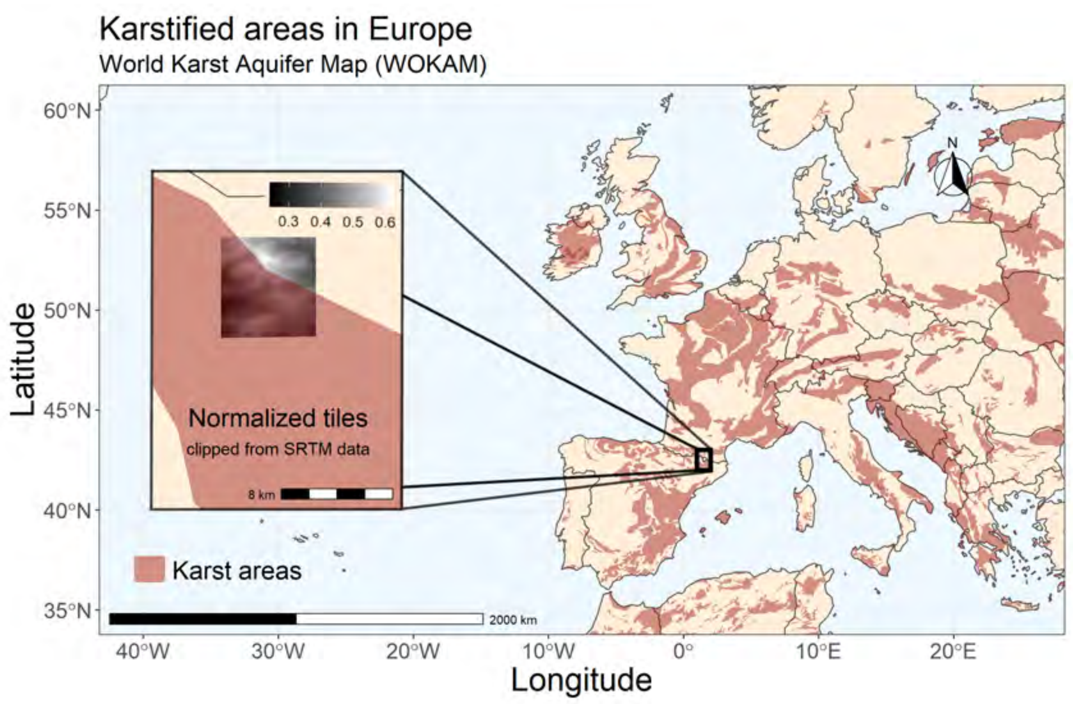


Figure 3.2: Map of karstified areas in Europe based on WOKAM data (right). Visualization of splitting the raster into tiles of 100x100 pixels (left)

According to Artugyan and Urdea (2016) and Marjan Temovski and Ivica Milevski (2015) the following parameters were derived from the elevation data, because it was possible to calculate them without including any further data. A selection of those parameters was used as additional predictors.

- slope, calculated according to Horn (1981) taking patches of 3x3 cells into account

- surface roughness, calculated as the difference of the maximum and minimum value of a patch of 3x3 cells
- aspect, also according to Horn (1981)
- TPI, which is the difference between a cell value and the mean value of its surrounding cells (Wilson et al., 2007)
- Terrain Ruggedness Index (TRI), for which the mean of the difference of a cell and each of its surrounding cells is calculated (Wilson et al., 2007)
- flow direction which indicates the direction of the largest descend in elevation or the minimum ascend if all surrounding cells are higher

These Parameters were calculated using the R package raster (Hijmans and van Etten, 2012).

Lidar

The Slovenian government provides lidar tiles with an extent of 1000 x 1000 pixels and a resolution of 1 m. Each tile spans an area of 1 km^2. Since most of Slovenia's surface is karstified (see figure 3.3) only the tiles containing bordering areas were used in this study. Downloading the files was automatized in R with a prior selection of relevant files to download using a grid shapefile containing all filenames of available tiles. Figure 3.3 shows the selected tiles that contained karst and non-karst areas from the WOKAM map. In the left window a zoomed tile is visible displaying the elevation values before rescaling the picture.

The compressed point clouds were extracted and processed in R. The points flagged with return level 2 and 8, which coresponds to water and earth surface reflection of laser were selected and then interpolated using a triangulated irregular network algorithm implemented in R, based on Zhu et al. (2008). To minimize computing time the corresponding karst masks were cropped from the WOKAM raster gener-

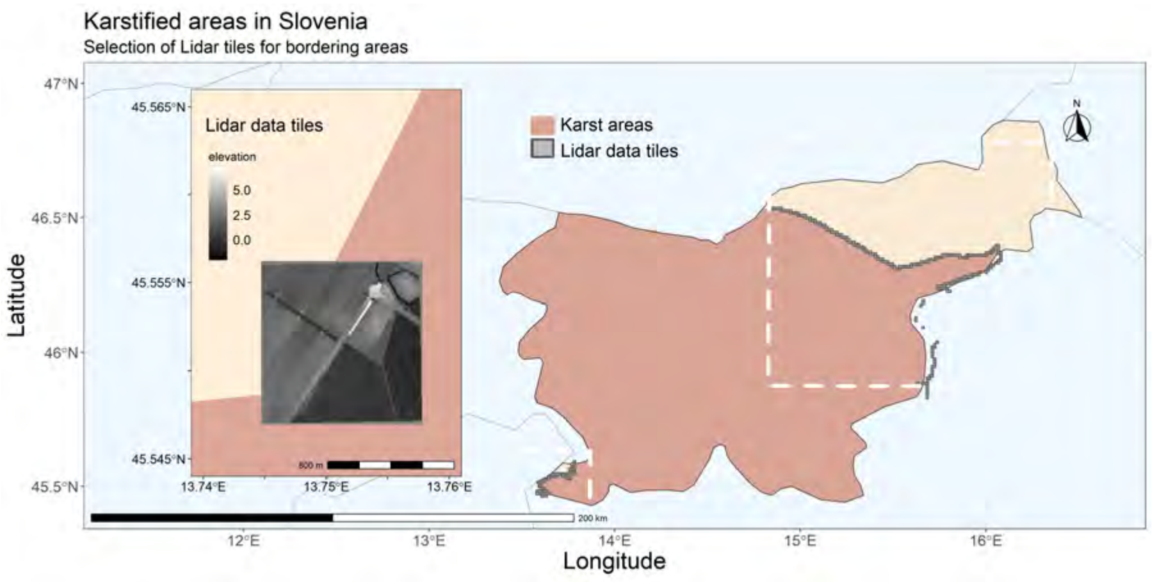


Figure 3.3: Map of karstified areas in Slovenia based on WOKAM data. Grey boxes are marking the position of Lidar point clouds that cover areas containing pixels classified as karst and non-karst, which were selected for the study. The inserted map of the tile gives an insight in the resolution and quality of the Lidar data.

ated for the SRTM data, then upsampled using a nearest neighbour algorithm and reclassification to binary information. Slope and flow direction were calculated for the Lidar data as described before for the SRTM data.

Statistical Pre-Analysis

Thorough statistical pre-analysis was performed only on the SRTM data. With regards to Ozturk et al. (2018), who related the occurrence of dolines to elevation, the occurrence of karst was linked to the slope or elevation of a pixel. The topographic parameters were categorized in a way that the same amount of pixels fall in every class. After that, karstified pixels and non-karstified pixels in every group were counted to see if a higher ratio of karstified pixels occur in a certain class of the given parameter.

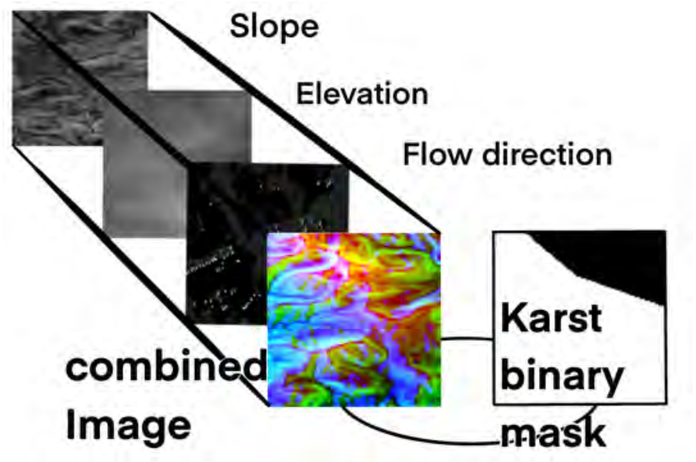


Figure 3.4: The preprocessing resulted in tiles of SRTM data and their coresponding binary masks. In order to choose to include slope, elevation and flow direction in the input data a binomial regression and cluster analysis was performed. This data was generated to be passed on to the CNN.

To get a first impression on the predictability of the karstified regions trough the elevation data of Europe (figure 3.2, left) two subsets of 10 000 and 20 000 regularly sampled pixels values were used in a binomial multiple regression to predict, if the corresponding area was karstified. In regular sampling of a raster the number of cells to be extracted is evenly distributed over the whole area with an equal distance between all sampling points. This sampling method ensured values from all over Europe to be included in the regression. The multiple regression was conducted assuming a binomial distribution for the output data with corresponding values of "1" for a karstified pixel and "0" for a karst-free pixel (Dormann, 2013). The most significant predictors for the karst areas were included in the data to be passed to the CNN afterwards. Traditional measures for the goodness of fit (e.g. R^2) are not suitable for binary data. Therefore the goodness of fit was accessed by comparing the null deviance with the residual deviance. The null deviance describes the performance of a model with only the intercept and the residual deviance represents how the model performs including the predictors. A small deviance indicates a higher model performance (Dormann, 2013). To avoid passing on highly correlated predictors a cluster analysis was performed (Hoeffding, 1948). If two predictors were significant, but highly correlated, one of them was excluded since they were not likely to contain extra information (Dormann, 2013).

3.3 Neural Network Architecture

In the following chapter the basics of CNNs and neural networks in general will be described based on Hastie et al. (2009) and Li (2019). A neural network is an algorithm freely inspired by the human brain. It consists of neurons, or perceptrons, that are connected non-linearly. It is therefore a form of complex regression. The neurons are stacked in layers and there are different types of layers depending on the tasks they are performing. In the most basic form of a neural network, there are input, hidden and output layers. The input layer has as many neurons as there are predictors in the model. These predictors are then passed on to the neurons of a hidden layer. The hidden layers consist of perceptrons and they are linked with weighted activation functions. The trainable parameters of the network are the weights and biases associated with the activation function. The number of hidden layers is chosen according to the complexity of the system and limited by computing power. After going trough several hidden layers, the Input is mapped into a probability output.
For computer vision, semantic segmentation or object detection tasks this simple architecture is not sufficient and a number of specified hidden layers have been introduced to successfully identify objects in pictures. The fundamentals of neural networks and their origins will coarsely be introduced before going into more detail explaining the features of a CNN and other terms and tools used in this project.

The Perceptron and the Basic Feed Forward Neuralnet

The perceptron is an artificial neuron located in the hidden layer and the basic compound of any neural network. It performs an operation called forward propagation. The input data (x_i) is passed on to the neuron and multiplied with a weight (w_i).

The resulting values are summed up and a bias (w_0, equivalent to the intercept of a linear regression) is applied to the sum. Adding the bias allows the activation function to be shifted. Consecutively non-linearity is added to the data by calculating the activation function g. Equation 3.1 represents the summing up of weighted input data, bias addition and calculation of the activation function, which is the core process inside a neuron.

$$\hat{y} = g(w_o \sum_{i=1}^{m} w_i \ x_i) \tag{3.1}$$

The output of the neuron (y) is passed on to the neurons in the next hidden layer as input. During training of the network the weights and biases are assigned to the neurons in order to minimize the loss function. The activation function (g) can be binary to turn a neuron on/off fully or gradual like a logistic sigmoid function.

Convolutional Neural Networks

CNNs are particularly suitable for classifying image data, because their hidden layers are not only a string of neurons, but they apply activation functions over sliding windows that scan the picture retaining the spatial relation of pixels. CNN's are usually a sequence of convolutional layers with a Rectified linear unit (ReLU) activation function followed by pooling layers for feature learning. After this sequence, also referred to as downscaling, the following layers are fully connected layers. The last layer is a softmax or sigmoid layer in the end to perform the classification Badrinarayanan et al. (2015); Kendall et al. (2015). In this project the structure suggested by Ronneberger et al. (2015) for biomedical image segmentation was used. It is structured similar to SegNet suggested by (Badrinarayanan et al., 2015) (figure 3.5) with a different mechanism upscaling the picture. Since Badrinarayanan et al. (2015) provided a superior visualisation of the network, it was chosen as a representative in this work.

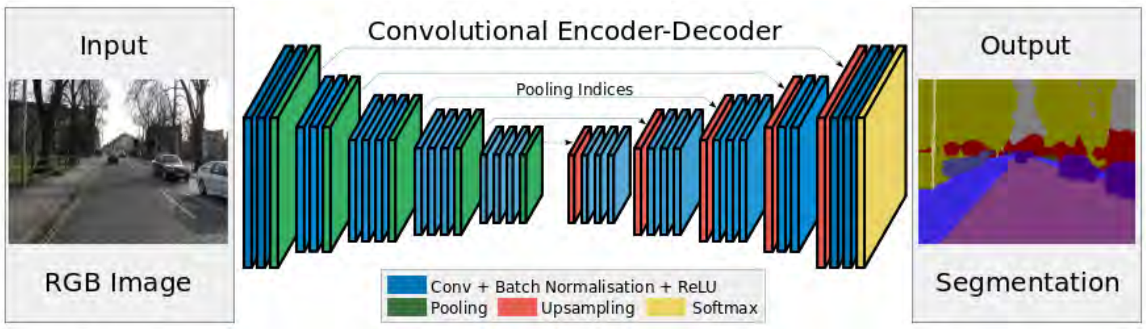


Figure 3.5: Exempalry scheme of a CNN from Badrinarayanan et al. (2015). Feature extraction is performed in the Encoder using convolution and maxpooling, afterwards the picture is upscaled and classified with a propability function. Batch normalization is used as a technique for regularization.

Convolution

Convolution can be seen as a way of scanning the picture and is a part of the downscaling and upscaling process. During convolution a filter is applied to the image. The filter is a fixed width and height window that slides over the image and performs a matrix multiplication using all the values on the picture (also called receptive field). Then the multiplied values are inserted into an new array which is called the feature/activation map. By using filters of different size, different features on different scales in the receptive field are considered. The filter can also be more specialized and sharpen the image or perform edge detection.
The parameters that need to be defined for the convolution layer are the window size, the number of filters, their spatial extent, the stride and the amount of zero padding. The stride is conventionally set to two and specifies the amount of pixels the filter moves in every step of convolution. The Zero padding refers to the way the boarders of the image are treated and weather or not zeros will be inserted in the feature map when the filter exceeds the edge of the image. For the SRTM and Lidar data the same window sizes of 3 x 3 pixels were chosen. Zero padding was not necessary and the stride was set to two.

Activation

Convolution results in a feature map of weighted values that will be passed on to an activation function as described in the introductory section about perceptrons. In CNNs the ReLU function is used frequently as an activation function of the convolutional layers. The ReLU function can be approximated by the following equation:

$$ReLU(x) = max(0, x) \approx log(1\ e^{x}) \tag{3.2}$$

The ReLU function increases training time compared to the classical sigmoid function.

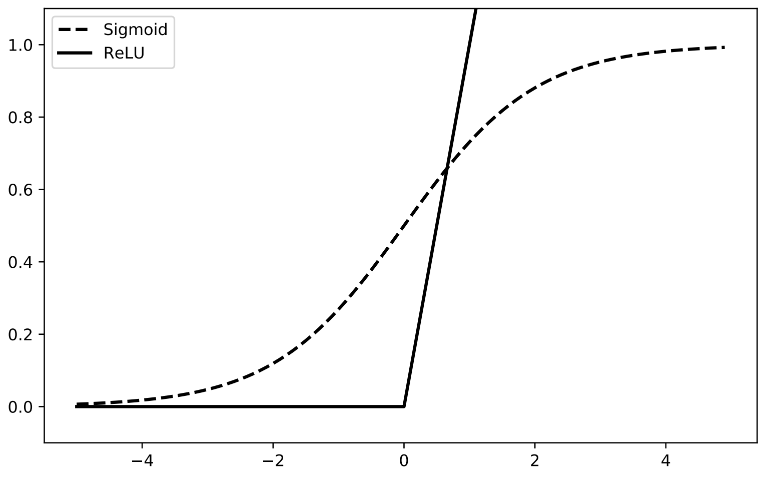


Figure 3.6: Diagram of the two activation functions used in this study (Li, 2019). The ReLU function was used in the hidden layers and the output propability was mapped using the sigmoid function.

Rescaling

Neural networks operating with ReLU functions suffer from mainly two numerical instabilities when faced with non-normalized data: Exploding gradient problem and dying ReLU problem. In this case the output of the activation layers is so large/small that the only way to recreate an accurate output is to scale the weights in opposite directions. The weights then become 0 or Infinite values that cause the network to collapse. On the other hand the combination of ReLU functions, weights and biases is unable to predict a propability which lies between 1 and 0 from values over a big range such as elevation which can range between values of 0 and several thousand meters. For these reasons the input data of the CNN must be normalized between 0 and 1 in order to avoid numerical instabilities. Therefore all values in this work were normalized and are unitless. Each tile was rescaled between its minimum and maximum value to 0 and 1 which is called feature scaling in machine learning. With x' being the rescaled data and x being the original values, rescaling was implemented as follows:

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{3.3}$$

Histogram equalization is a technique to improve contrast in images. Most frequent intensity values are spread out over the histogram and therefore the intensity range of the image is stretched. Histogram equalization was applied to the data before passing it to the network in this study.

Pooling

Pooling is a way of downsampling in which a pool of cells, for example a window of 2x2 cells like in figure 3.6, is summarised into one value. The average of the cells (figure 3.7, right) can be used as well as the maximum value of pixels in the pool (figure 3.7, left). The same parameters as for the convolutional layers have to be defined for pooling layers ecxept it is not common to include zero padding and there is only one filter, of which the size has to be defined.

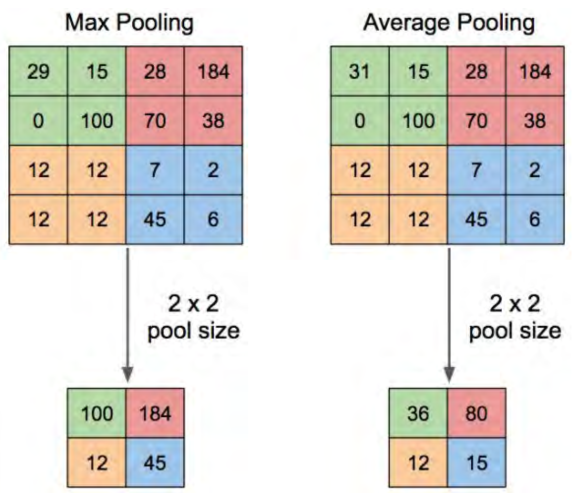


Figure 3.7: Downscaling the picture by choosing the maximum pixel value inside a window and mapping it to a smaller resolution image to extract features (Li, 2019).

Upscaling

In the process of upscaling the feature extraction is reversed and the downscaled image is brought back to its original format . The positioning information of the maximum value found in each window during the pooling operation is retained. In this step the features extracted in the different downscaling layers are extracted while the image is brought back to its original format allowing a link between the features and the output.

Classification

The classification layer, which lies at the end of the decoder and performs the final classification task uses a sigmoid activation function to map the values between 0 and 1 giving a class propability between the two possible outcomes. The sigmoid is pictured in figure 3.6 and is defined in equation 3.4. This function converts the input vector into a propability distribution that includes the same number of propabilities as the input vector. Simply the function returns a vector with a propability for each of the pre-defined classes. The sigmoid function can be described given the vector input (x) and the weight vectors (w_i) according to Li (2019):

$$sigmoid(x) = \frac{1}{1\ e^{-1}} \tag{3.4}$$

The output of the function is a vector of propabilities of each pre-defined class. In some of the networks trained in this study one-hot encoding is used for the predicted variables. Usually one-hot encoding is used for multi-classification problems. This encoding is binary and every class is assigned with a channel in an array. Each pixel belonging to class n would be assigned with 1 in channel in the array and all pixels belonging to other classes are assigned 0 in this channel. The network outputs a propability for every channel and the maximum is selected to predict the class.

Implementation

The project was realized using Tensorflow (Martin Abadi et al., 2015) with a Python frontend. As shown in figure 3.8 the final model consists of three convolutional blocks containing convolutional, max pooling, batch normalization, and a ReLU activation layer. For a comparison of two different loss functions two further models with an additional convolution block were trained. The model trained on the high resolution data included an additional convolutional block as well. The output layer is one-hot encoded, when suitable for the chosen loss function. The resulting models had 537203 / 536934 trainable parameters for the SRTM/Lidar data including weights of the convolutional filters and biases.

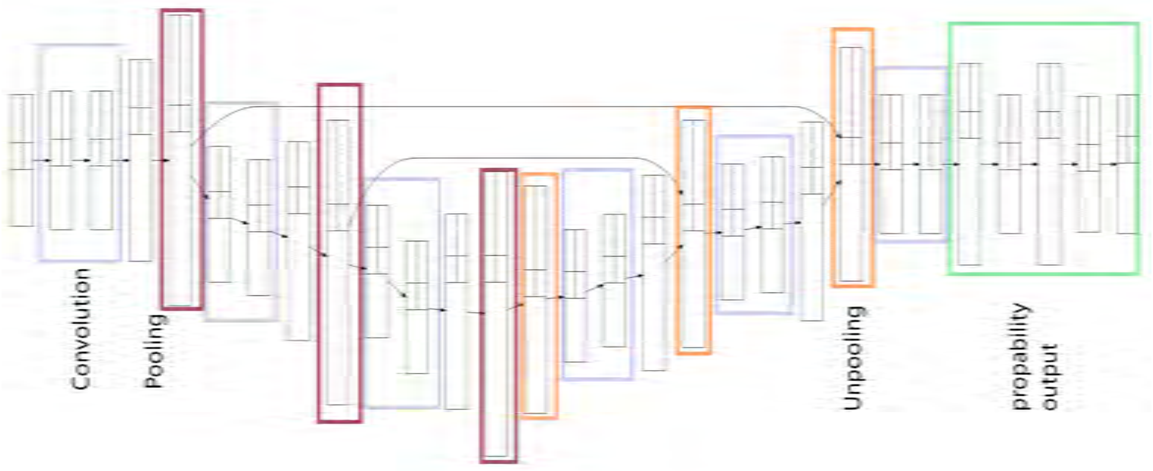


Figure 3.8: Final model structure with 3- stage convolution and deconvolution. Unmarked boxes implicate batch normalization layers.

Training was performed on an Intel(R) Core i7 CPU in combination with 8GB RAM. Although a Nvidia GPU was accessible with tensorflow, the available space limited the batch size to three pictures at a time which is too low for a reliable gradient estimation for optimization (Keskar et al., 2016).

3.4 Training

The CNN was trained using backpropagation method and adam optimizer, which is a stochastic gradient-based optimization method (Kingma and Ba, 2014). Adam has proven to be a robust optimizer for ML purposes and is more efficient than the standard stochastic gradient descent optimizer (Loshchilov and Hutter, 2015). A data generator performing real-time augmentation, rescaling and histogram equalization was used to feed a number of images (batch) at a time to the network. For the SRTM data 3-channel images including elevation, slope and flow direction (see figure 3.4) were yielded directly from the GeoTIFF file and passed to the network as a three channel array. In every step of the training a number of 64 images was randomly chosen from the training data and loaded into Python. For the high resolution data only the elevation was used as a predictor. Due to memory limitations the batch size for Lidar data was limited to two. Keskar et al. (2016) found that batch sizes of 32-512 datapoints perform best on estimating the gradient. The CNN was trained for 100 epochs, with one epoch corresponding to an iteration over the total number of images in the training dataset.

After undergoing the steps described in the first sections of this chapter the difference between the model output and the expected output (loss function) was calculated for every picture of the batch. The weights and biases are adjusted iteratively for the distribution of predictions to match the distribution of expected values. The magnitude of adaptation of the parameters is controlled by the learning rate, with higher learning rates corresponding to a quicker reaction to gradients by the optimizer (Loshchilov and Hutter, 2015). To eliminate sources of error associated with the choice of loss function or model structure two models with an additional convolutional block and two different loss functions were trained.

Loss Function

The loss function is a way to assess the error that comes along with the output a model produces. The loss is calculated separately on the training and testing data. The optimizer iteratively adjusts the weights and biases in the network with an objective of minimizing the loss, hence the Error, of the model. For the main model a sigmoid categorical crossentropy function was chosen. Crossentropy functions are widely used for classification problems with the sigmoid crossentropy being a special case for binary classification. The function is suitable for one-hot encoded outputs and with y_t being the expected output and y_p being the predicted output the loss function is defined for $y_p \geq 0$ as follows:

$$H(y_p, y_t) = -y_p \ y_t \ log(1 \ e^{-|y_p|}) \tag{3.5}$$

Additionally the model was trained with the dice loss function suggested by (Shamir et al., 2018), described in equation 3.6 and the binary crossentropy loss from tensorflows default loss functions. Binary crossentropy is a special form of the sigmoid crossentropy (equation 3.5) considering exclusively two classes.

$$D(y_p, y_t) = \frac{2|y_p \cap y_t|}{|y_t|\ |y_p|} \tag{3.6}$$

The dice coefficient was developed for binary segmentation of medical images, which show large imbalances in class occurrence.

Accuracy

Accuracy is calculated additionally to the loss function. A lower loss usually corresponds to a higher accuracy. Accuracy is calculated not in order to learn the model Parameters but to give the user an idea of how well the model performs. Therefore it is usually a percentage or a understandable rate of correct predictions. In the case of karst prediction the accuracy of a picture which contains 100 pixels of which 60 were predicted correctly the accuracy is 60%.

Overfitting

A common problem of machine learning models is overfitting in which the model is performing very well on the training data but due to its over specialization fails to generalize. This lack of generalization leads to bad performance on the testing data and inability to predict unknown images. It is very common with deep networks containing large number of parameters, since they are very specialized.
To avoid the occurrence of overfitting regularization techniques are applied. In this study dropout layers were used to prevent overfitting. Dropout layers induce a certain percentage of random neurons in a layer to be deactivated. This prevents specialization of the network because the weights associated with other neurons in the layer

will be adjusted in a changed environment. Furthermore the pathways trough the network are diverged by putting blockages where the neuron is dropped out. In this case the network cannot be depending on only one way to pass trough the information from the input image. Dropout layers were implemented in the decoder of this network leaving out 50% of the connections between the neurons randomly.

Sensitivity Analysis

One of the objectives of this work is to find the influencing factors for the CNN to classify a pixel as karstified and which informatio is missing in case the model cannot be trained successfully. To find out if the CNN could identify karstified areas with more success if the differences in topography between karst and non-karst were more pronounced, the data was augmented. The model was trained with the same amount of pictures and under the same training conditions using three different inputs. The pixel values were increased by 5 %, 10 % and 20 % of the original value for elevation, slope and flow direction befor rescaling the data. Performance of the model was evaluated for all three scenarios. The increase in values for only one class is regarded as an increase in signal strength for the network, hence conclusions about the sensitivity of the model can be drawn. Augmentation was implemented in the generator, augmented data was created on the fly, so only the original tiles are saved as files.

3.5 Quantile Regression Forest

As described in Tyralis et al. (2019) a regression tree parts the variable space in two according to a rule or threshold. Within those two seperate variable spaces the same operation is applied with new rules. The splits are called nodes and at the end of the node there is a so-called leave that applies the classification. When applied in its original form (Breiman, 2001) the best model is estimated by minimizing the sum of

squares and finding the best possible estimate of the conditional mean. Furthermore a forest of regression tree is grown using a subset of the data. To make a prediction the average of all trees in the forest is used to estimate the output. Meinshausen (2006) suggests a forest regressor predicting quantile ranges of the estimated class. Therefore in a QRF the quantile loss is minimized to predict a certain quantile. Actually the QRF computes a distribution function for all estimates of the output.
The proposed algorithm was made available in the R package quantregForest which was used in this study. The number of trees grown was set to 500 and the trees were limited to have 20 endnodes to lower computational time and avoid memory issues and overfitting. Performance of the QRF was accessed as the percentage of correctly predicted pixels, as it is for the CNN. Since high accuracies can occur due to the uneven distribution of pixels belonging to a class the predictions of the QRF were benchmarked with a random model, which predicts a certain class with the proportion of its occurrence in the dataset.
To train the QRF the Raster was converted into a table linking the values for slope, elevation and flow direction to the occurrence of karst, as described for the binomial regression in the statistical pre-analysis. For the SRTM data a regular sample of 10 000 pixels was taken and split randomly into training and testing points that were scatered over the whole area. 60% of the number of observations were used for training and 40 % was used for testing. To train the QRF for the Lidar data 39 tiles of 1000 x 1000 pixels were randomly chosen. Each of them was resampled regularly to 1000 pixels due to memory availability using trial and error to find the maximum possible resampling rate. Each of the 1000 values of 39 pictures were then joined to train a regression forest. Training was performed on one tile with a resampling number of 10 000 pixels.
When using a regression forest for classification, the input is only the pixel values and the output is the class assigned to that particular pixel. Neighbouring pixels are not taken into account and the spatial distribution of the data is neglected.
It should be noted here that the QRF was mainly implemented to validate the sen-

sitivity analysis carried out on the CNN. In case of the CNN performing badly on the data with 20 % increased values for elevation, slope and flow directon, the QRF can be used to approve the false choice of computer vision methods, if it succeeds to model the occurrence of karst. Vice versa if it also fails to identify karstified areas the underlying data is probably not sufficient to model the geological parameter.

4. Results

The SRTM data was classified into ten bins of slope and elevation values with the same number of pixels in each bin. Figure 4.1 shows the occurrence of karstified pixels within each class. The sum of karstified and non-karstified pixels is always the same for each class. Pixels at a higher altitude and pixels with higher slopes (steeper topography) tend to show a higher ratio of karstified pixels. Figure 4.1 clearly shows a decline in pixels without karst and an increase in pixels that fall into karst areas with ascending elevation and slope values.

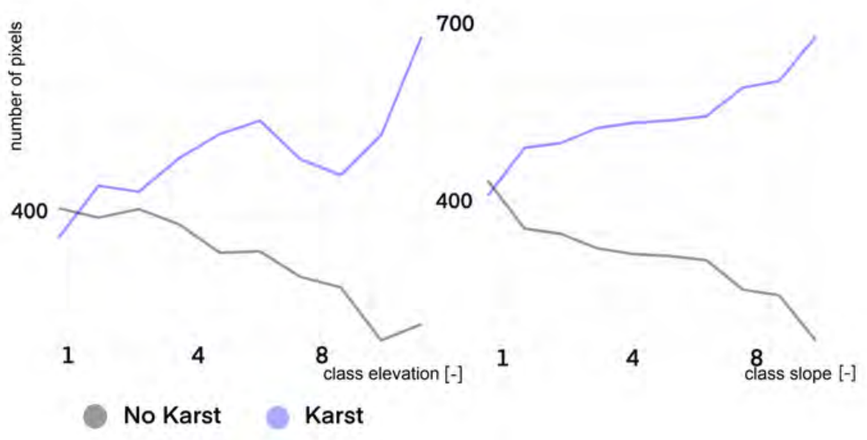


Figure 4.1: pre-analysis of the data classifying the raster values and relating Karst pixel occurrence to slope and elevation. A positive trend is visible for both predictors.

The statistical pre-analysis showed that, out of the seven predictors (elevation, slope, aspect, TPI, TRI, roughness and flow direction), elevation and slope were highly significant in explaining the presence of karst (p-value $2e^{-7}$ and $6.44e^{-15}$) and surface roughness was almost significant (p-value 0.0554) for the subset of 20 000 values. For the second subset of 10 000 values the slope and elevation also showed the highest significance (p-values $2e^{-7}$), roughness was even more significant as before and flow direction showed a high p-value of 0.06 as well. TPI, TRI and aspect were irrelevant in predicting the occurrence of karst and therefore they were discarded in the further study. In the case of 10 000 points the null deviance of 12 100 on 9999 degrees of freedom declined to 11 865 on 9992 degrees of freedom. For the subset of 20 000 values the deviance declined from 11469 to 11262 and the degrees of freedom lowered from 9504 to 9497. These declines indicate an increase in predictability of karst presence. The Cluster Analysis showed that the slope and roughness are very closely correlated and it was assumed that including both of them in the model does not add more information but duplicates a predictor. Therefore elevation, slope and flow direction were chosen as three predictors to be added to the model input.

CNN with SRTM Data

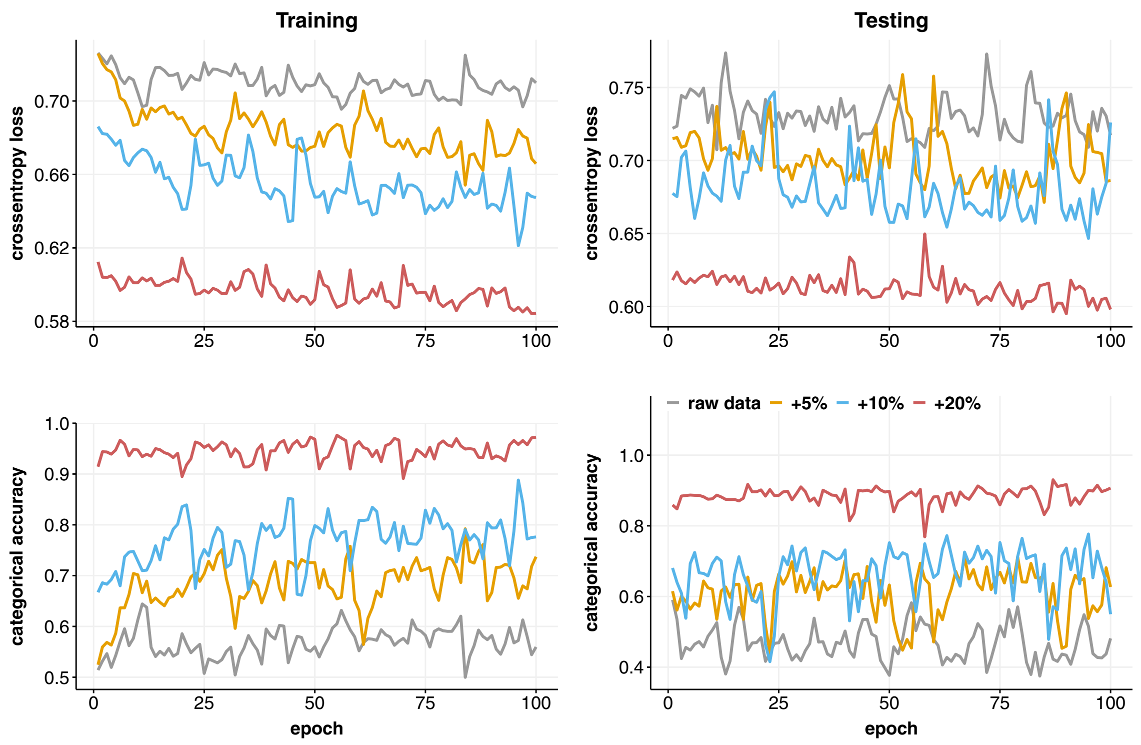


Figure 4.2: Development of the loss and accuracy during training of the model using the raw data and 5%, 10% and 20% increased values of elevation, slope and flow direction.

The final SRTM data consisted of 8827 (1 GB) image files to train and test the model. 7059 (80%) of those images were used for training and 1768 were used to test the model. The CPU reached its maximum during training with a batch size of 68 images at a time. Figure 4.2 shows the development of model loss and accuracy over 100 epochs. Ideally the testing and training loss as well as accuracy converges over the epochs, showing a decline in loss and a rise in accuracy. The model trained on the raw data (grey line in figure 4.2) oscillates randomly around a high loss (between 0.69 and 0.73) and a corresponding low accuracy (between 0.64 and 0.38). There is no trend apparent in the development of the training evaluation criteria. Figure 4.2 shows that the models trained with the augmented data (5 % 10 % and 20 %) are converging slightly during the training. Overall the span of accuracy and loss values explored by all the models is very narrow. This is also represented by the width of the boxplots in figure 4.3. The models derived from augmented data rise in accuracy and decline in their loss as visible in both, figure 4.5 and figure 4.3

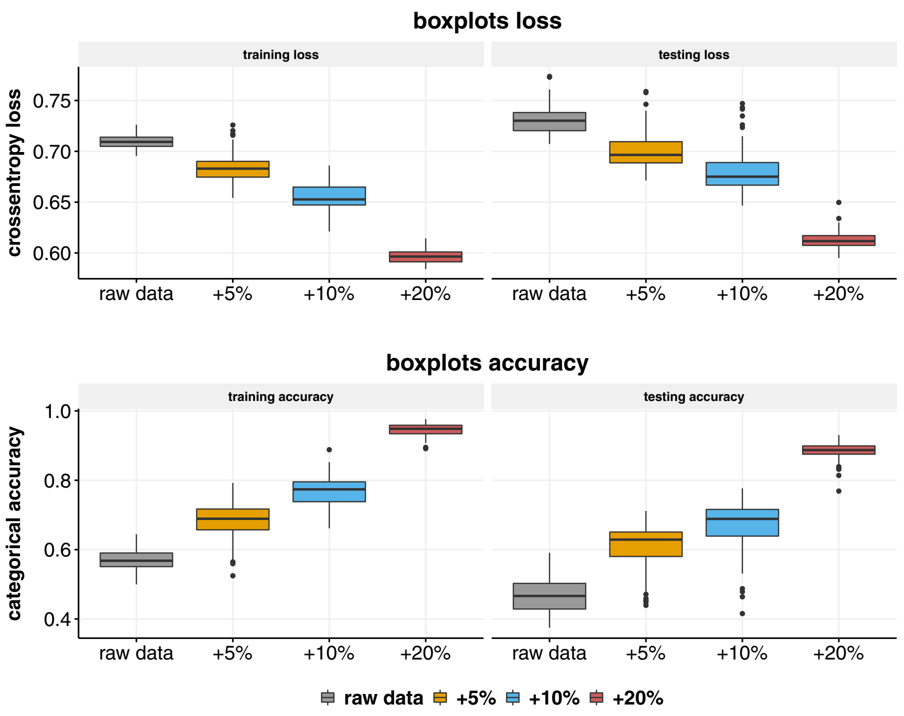


Figure 4.3: Boxplots of the measures for goodness of fit for the models trained on the raw data, and augmented datasets.

Table 4.1 summarizes the lowest losses and the corresponding accuracies, as well as the highest training and testing accuracies. It is noticeable that for the augmented data, the lower losses and higher accuracies occurred mainly in later stages of the training, except for the highest training accuracy of the dataset increased by 20 %, which occurred in the middle of the training. The models trained on the raw data reached the highest accuracy in training after 11 epochs, the lowest loss was found at epoch 56 and the testing accuracy didn't improve trough the whole training. The models trained on raw data reached a maximum training accuracy of 0.64 and a corresponding testing accuracy of 0.53 (see table 4.1). The maximum testing accuracy was at 0.59 with a parameter set that reached an accuracy of 0.51 on the training dataset.

The highest training accuracy reached by the network improved from 0.64 to 0.79 for an increase of topographical parameter differences by 5 %, followed by an increase to 0.89 and 0.98 for an augmentation of 10 % and 20 %. This development is as expected and the augmentation made it possible for the model to connect the input parameters to the output mask. The same trend is apparent in figure 4.3 where the median accuracy is increasing and loss values are decreasing with augmentation. Maximum training accuracy at an augmentation rate of 20 % was 34 % higher than with the raw data. Maximum testing accuracy increased at the exact same rate. In table 4.1 the high testing accuracies don't seem to correlate with high training accuracies for the raw data. To see the relationship between training and testing accuracies as well as losses the measures were plotted against each other in a scatterplot in figure 4.4.

For each training dataset the trends were estimated using a linear regression. The formula of the regression lines are shown on the left side of each plot. The slope represents the trend apparent in the data. In the top left plot the training loss is plotted against the testing loss. As expected, the slope of the regression line is positive which means that higher loss values for training yield higher loss values in testing. The same relationship is visible in the top right plot in figure 4.4 for testing and training

Table 4.1: Maximum accuracy, minimum loss and maximum validation accuracy and the corresponding measures for goodness of fit. Maximum validation accuracy was chosen to select the best model.

Highest training accuracy

	training loss	training accuracy	testing loss	testing accuracy	epoch
raw	0.70	0.64	0.71	0.53	11
5%	0.65	0.79	0.69	0.65	84
10%	0.62	0.89	0.68	0.65	96
20%	0.59	0.98	0.61	0.90	55

Lowest loss value

	loss	training accuracy	training testing loss	testing accuracy	epoch
raw	0.70	0.63	0.71	0.50	56
5%	0.65	0.79	0.69	0.65	84
10%	0.62	0.89	0.68	0.65	96
20%	0.58	0.97	0.61	0.90	99

Highest testing accuracy

	training loss	training accuracy	testing loss	testing accuracy	epoch
raw	0.73	0.51	0.72	0.59	1
5%	0.67	0.70	0.67	0.71	79
10%	0.64	0.79	0.65	0.78	95
20%	0.60	0.93	0.60	0.93	87

accuracies. R^2 values are very low throughout all regressions and the data points are scattered a lot around the lines. Although the R^2 values are very low and p-values are above 0.05 for all regression lines, conclusions about the development of loss and accuracy during the training can still be drawn, since the goal of this regression is not to predict but to see if there is a trend. The bottom plot in figure 4.4 shows the increase of testing accuracy, the criteria used to select the model, and the training loss, the value used to optimize the model. This is the most important relationship to be examined since if the training loss doesn't show an increase in testing accuracy there is no connection between the model optimization and the modelling goal.

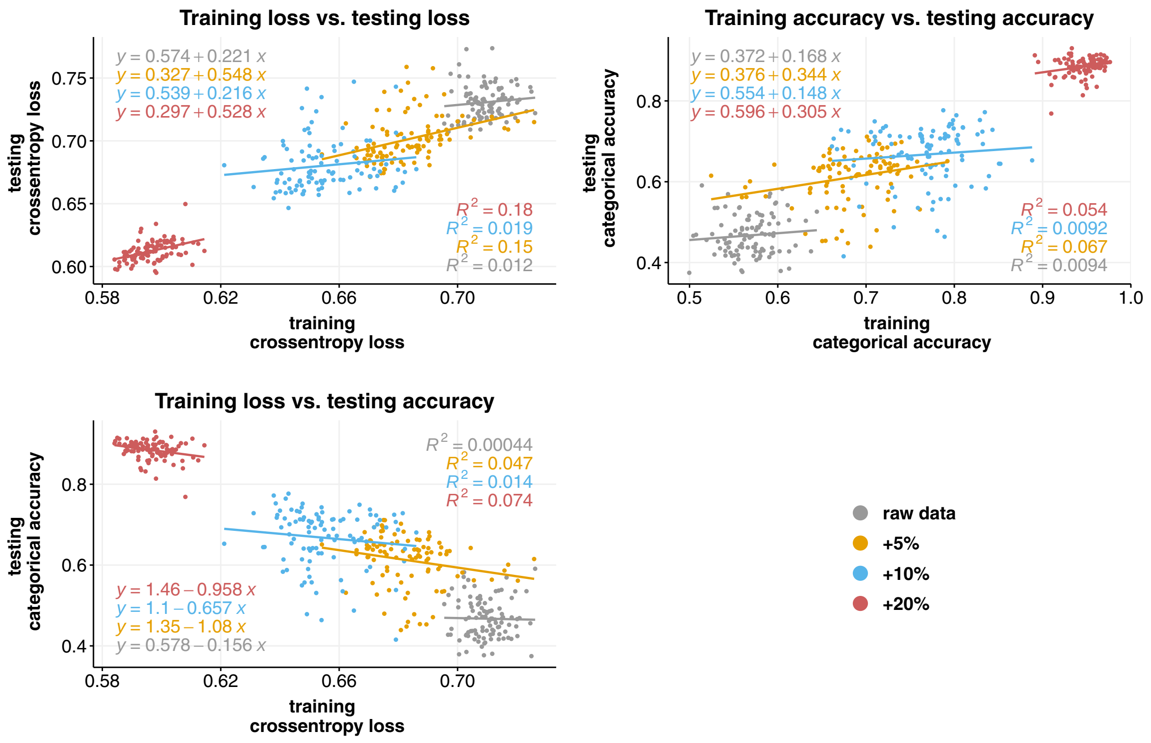


Figure 4.4: This figure illustrates the relationship between the training and testing loss, accuracy (top plots). The relation of the crossentropy loss and the testing accuracy.

For lower training losses the testing accuracies should be lower. This assumption excludes a situation of overfitting, which is not the case, since it is related to a very high performance on the training data. For the augmented data of 5 %, 10 % and 20 % the slopes of the regression lines are -1.08, -0.66 and -0.96. Only for the the raw data there is no relation between the testing accuracy and training loss (slope of the regression line: -0.1).

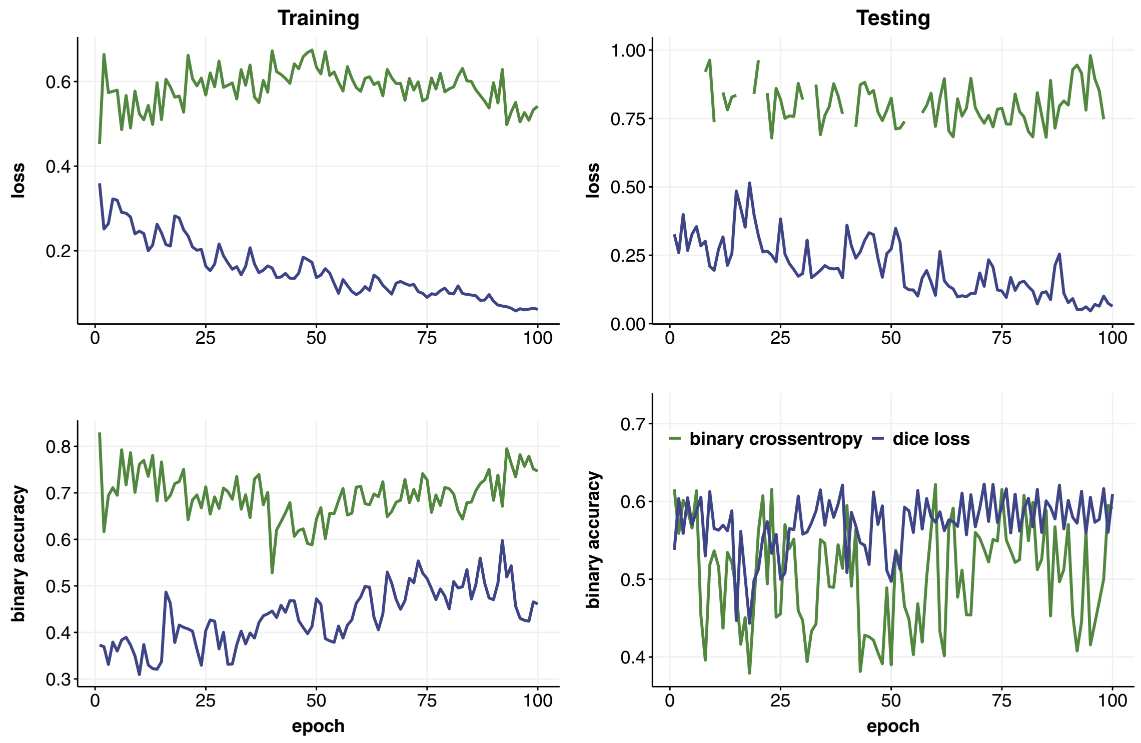


Figure 4.5: Development of loss and accuracy for dice loss function and binary crossentropy loss.

The next part summarizes the results of using two different loss functions on a deeper network with one extra convolutional layer. Figure 4.5 shows the development of the dice / binary crossentropy loss and binary accuracy. The testing binary loss function exceeded a value of 1.0 in 9 epochs, which were omitted from the data shown in figure 4.5 top right plot. The Dice loss function is converging nicely throughout the training, but the testing accuracy didn't improve. The two loss functions were only used to train a model with the raw data and the results didn't improve from the first try with a smaller model and categorical crossentropy function. An attempt to improve model performance was to also include class weights according to the proportion of class one with regards to class zero. In this idea the loss function would be weighted by the proportion of each class. The addition of a weighted loss function didn't have a significant effect on the model performance and was abandoned for that reason.

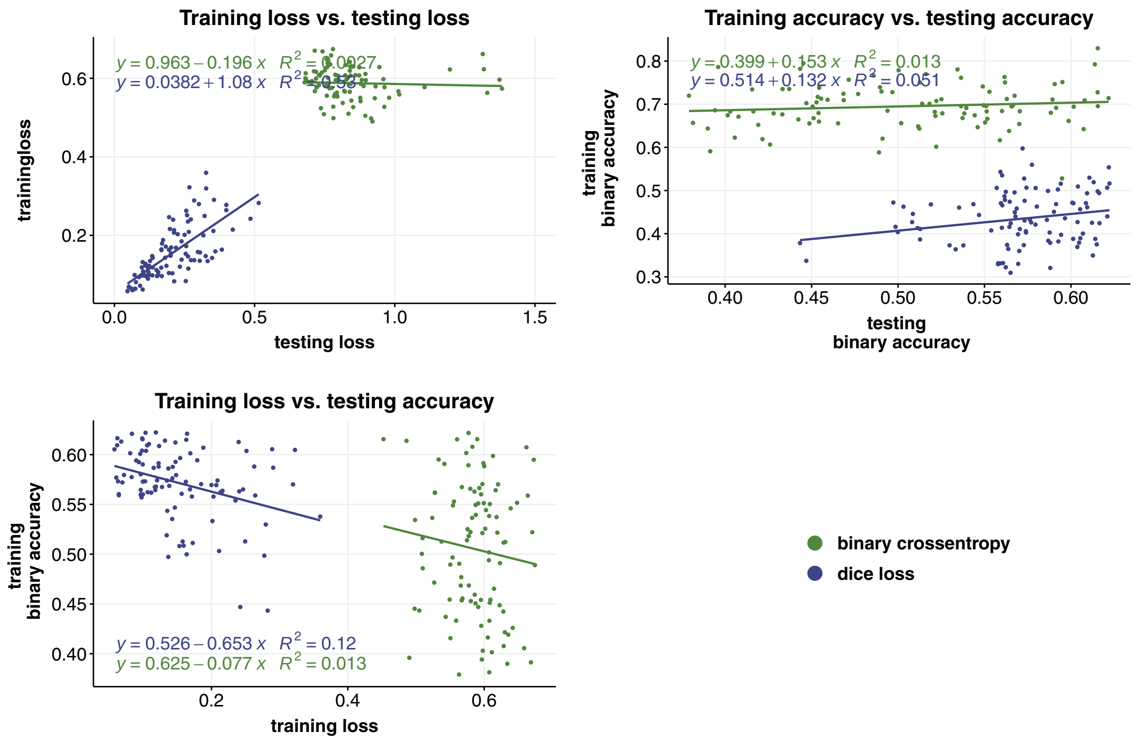


Figure 4.6: The relationship for loss and accuracy using a dice loss function and a binary crossentropy function using the SRTM data.

In the following figures 4.7 and 4.8 show the predictions for two exemplary pictures of the training batch. Figure 4.10 and figure 4.9 show the predictions on the testing batch. The figures show five rows of images with the same layout. In the first three rows the input channels are split up and from left to right the level of augmentation is increasing. The augmentation caused the picture to look as if a shadow was cast over the areas where there is no karst, which led to a better identification of such areas. The three input channels, especially the first channel (elevation), show some vertical lines. These lines occurred due to histogram equalization, which is used frequently in CNN preprocessing and was a necessary step for this project due to exploding gradients problem discussed in chapter 3. The fourth row of figure 4.7, 4.8, 4.10 and 4.9 shows the expected occurrence of karst with a black pixel (1) indicating a karstified area. In the last row the predictions made by the model with the highest testing accuracy is visualized. The pixels identified as karst are randomly scattered around the predicted mask throughout for the raw data. With only an increase of 5 % the network is able to identify patches of karstified pixels. There are examples in which the network managed to identify karstified areas precisely even with the 10 % increase in input signal (figure 4.7, 4.9 and there are pictures in which even an increase of 20% of the input signal didn't lead to a proper classification of the image (figure 4.8, 4.10). The predictions are extracted from the maximum probability of the two output channels of the one-hot encoded data.

The output for a testing tile of the models trained with different loss functions are visible in figure 4.11. In the second row the expected output is aligned with the probability output of the models chosen by the best validation accuracy. The values mapped represent a probability that the pixel is karstified, the closer the probability is to 1 the more likely it is to find karst in this area. It is visible that the loss functions don't have an impact on the quality of the predictions nor did the increased depth of the network make a difference in predicting the occurrence of karst. The only difference is that the dice loss function is able to predict patches of karstified pixels even with the raw data for some images, although they don't relate to the expected output (see appendix A.3).

To get a better understanding of the predictions achieved by the models the first ten images of the first training batch along with the augmented data are plotted in appendix A.1. The respective testing images can be viewed in the second appendix (A.2). For more information about the application of the different loss functions on the SRTM data, five images of the testing dataset are pictured in appendix A.3.

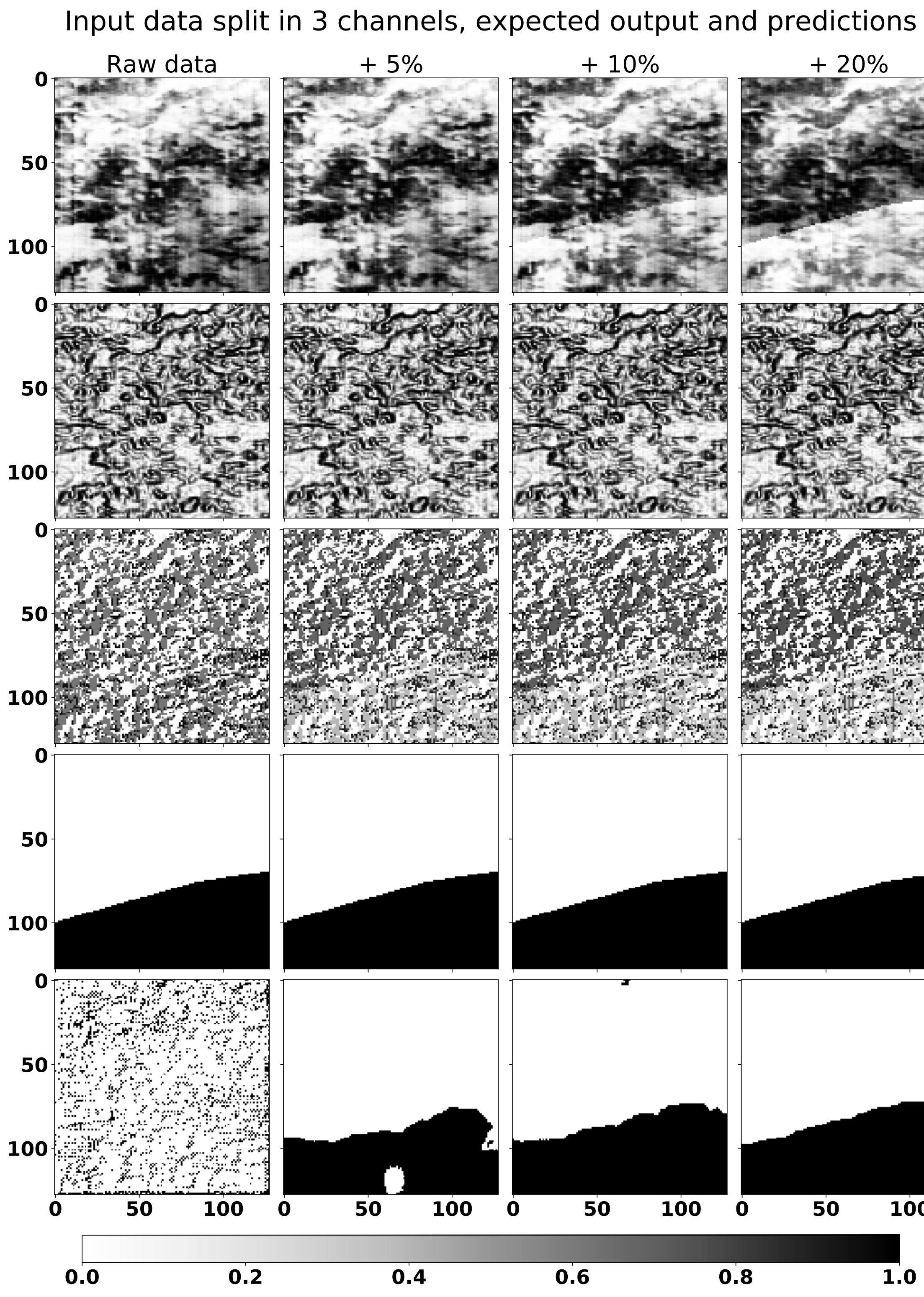


Figure 4.7: Input and Output of the third picture in the first batch of images for training.

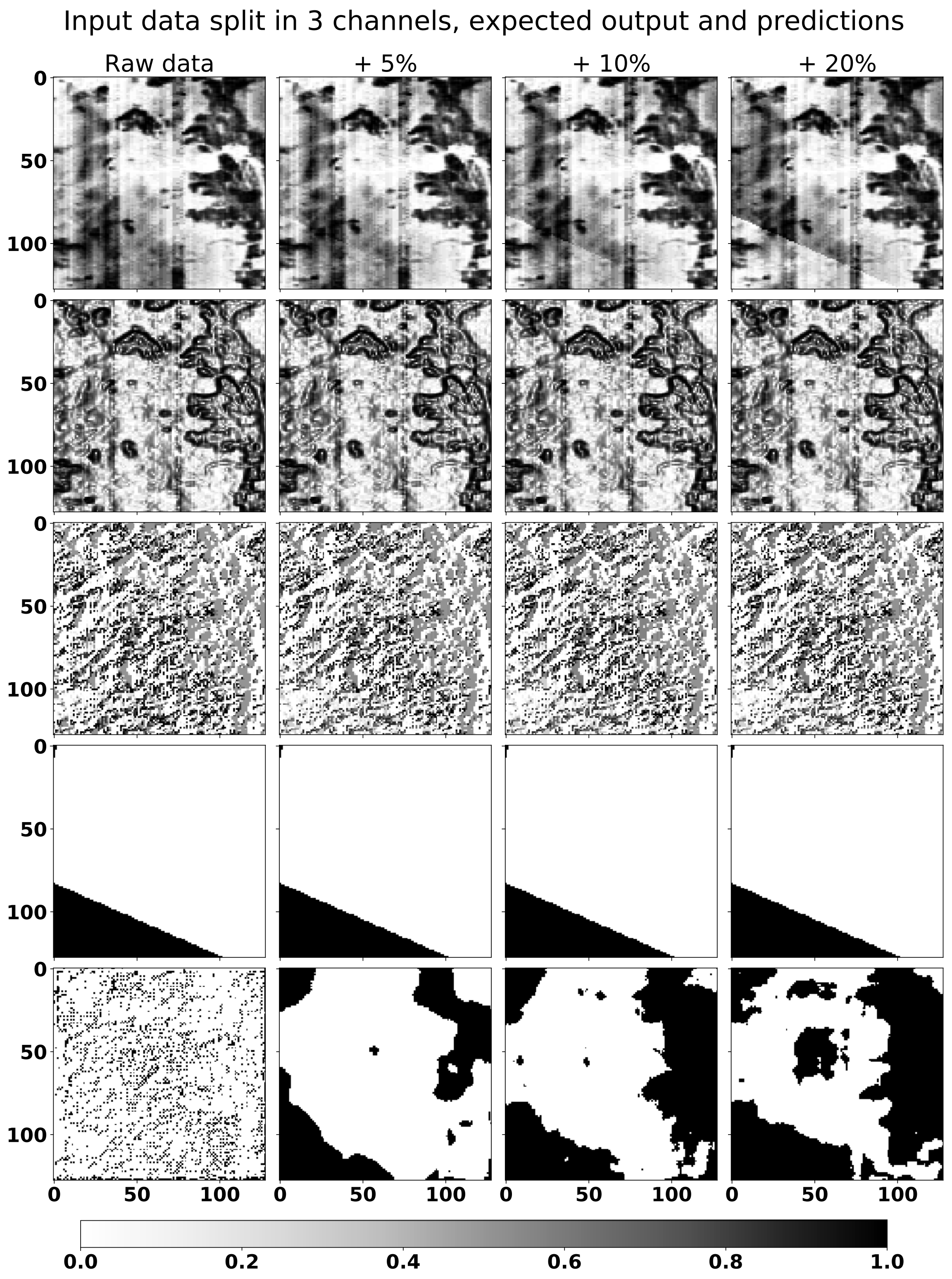


Figure 4.8: Input and Output of picture number 15 in the first batch of images for training.

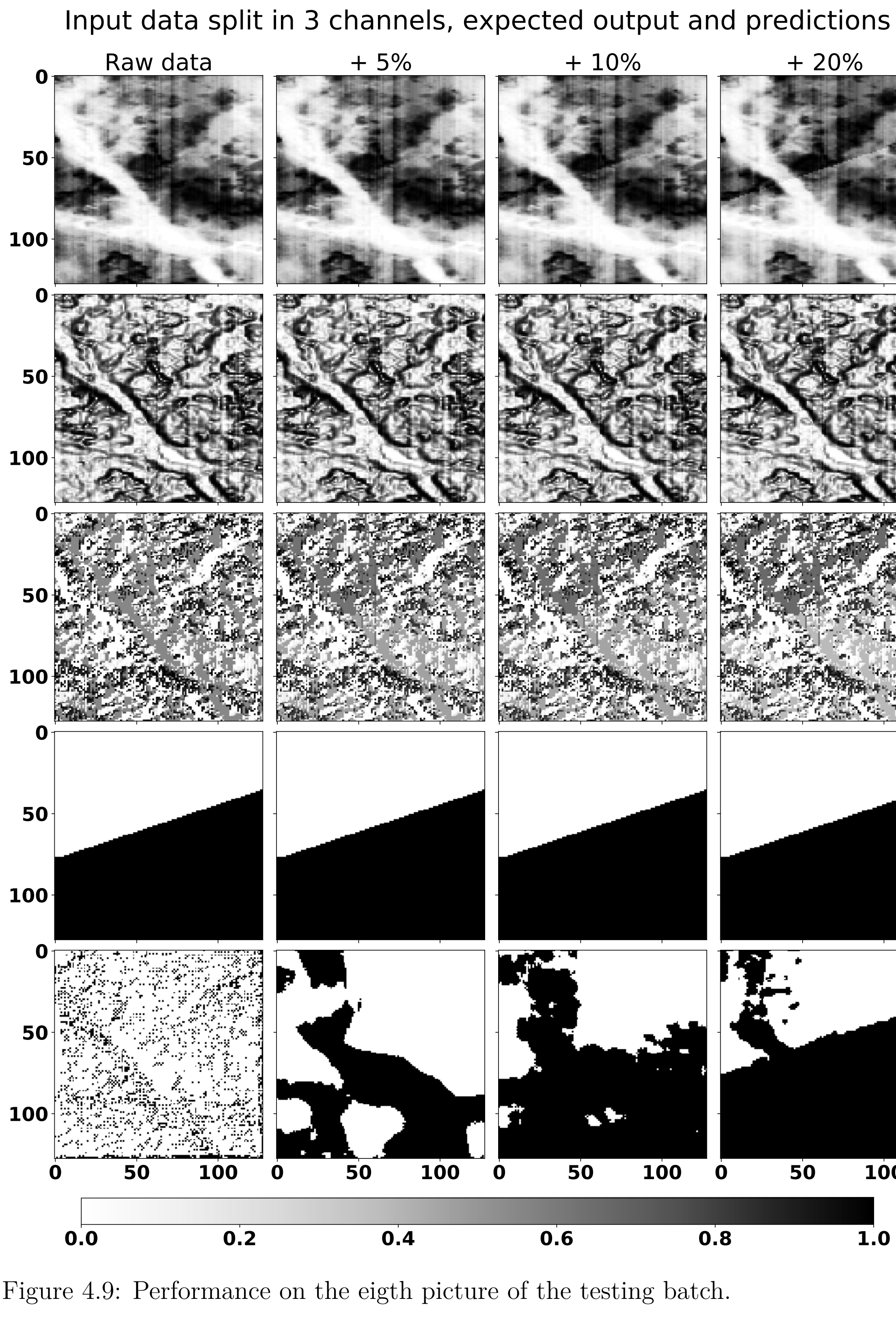


Figure 4.9: Performance on the eigth picture of the testing batch.

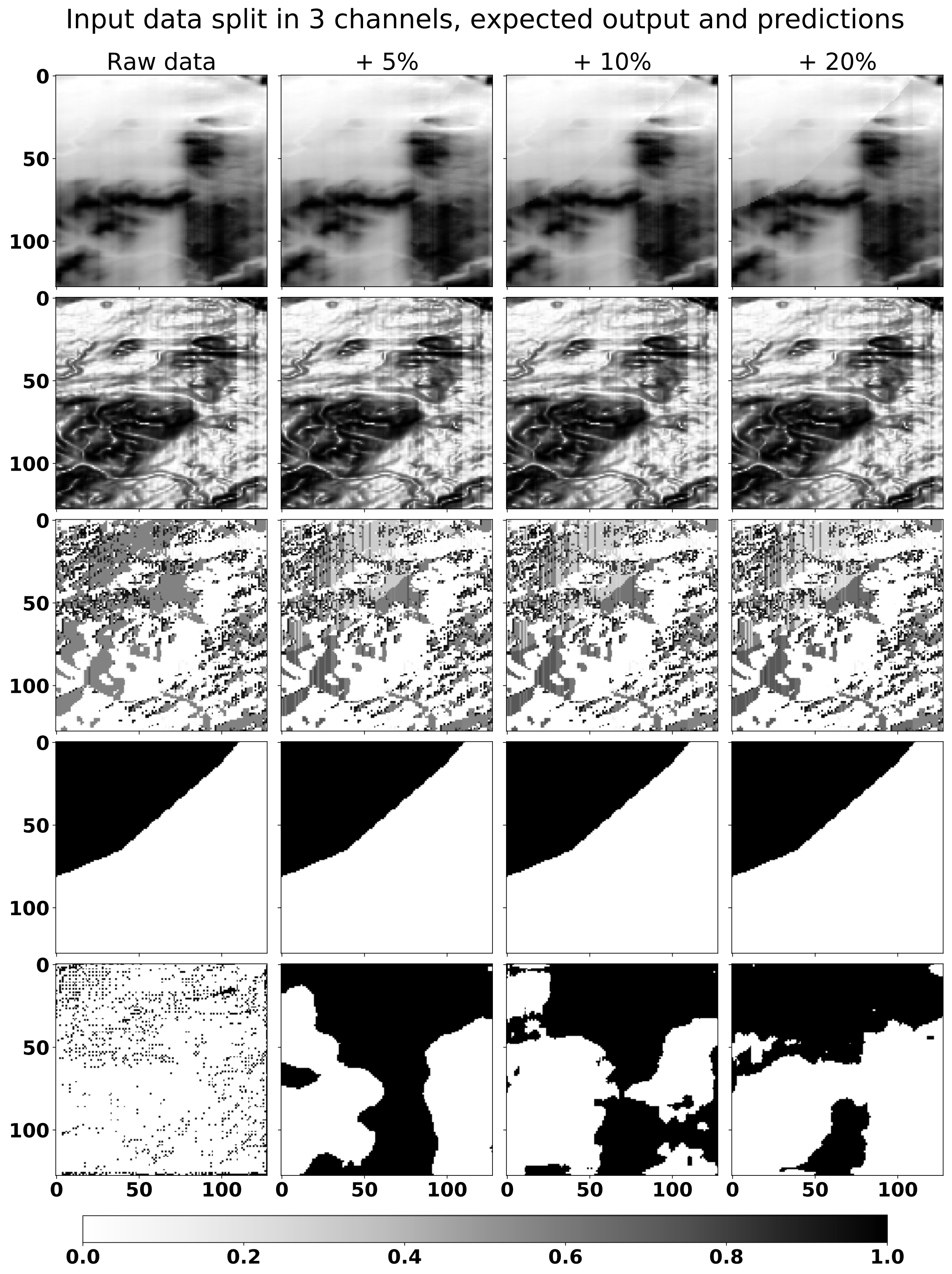


Figure 4.10: Performance on the 20th picture of the testing batch.

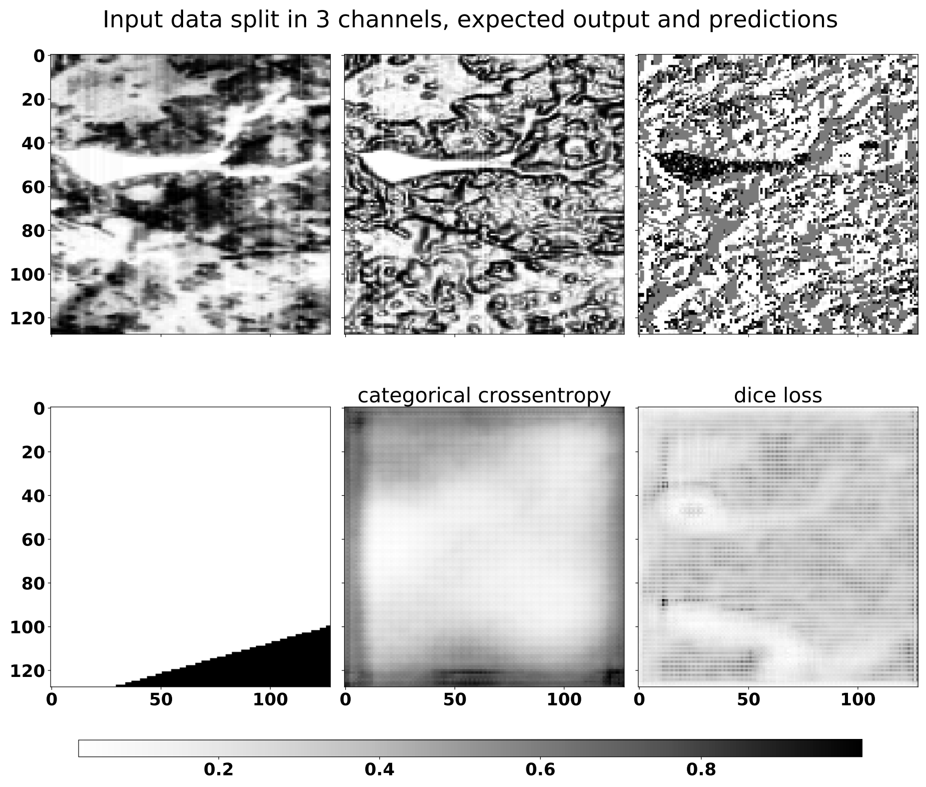


Figure 4.11: Prediction of the first picture of the testing batch.

CNN with Lidar data

The CNN was trained using 172 images with 1000 x 1000 pixels. Due to limited memory the batch size had to be lowered to two. Figure 4.13 shows the fifth picture of the training data along with the karstified regions covering the same picture. The class imbalance for the high resolution data was opposite of the imbalance for the SRTM data, since Lidar data from Slovenia was used and most of its surface consists of karst.

First the model was trained using a binary crossentropy function and accuracy for data without one-hot encoding (Figure 4.12, green line). When looking only at the loss and accuracy values the performance of the model is much higher than the models trained and tested on the SRTM data. The CNN yielded a maximum training accuracy of 0.817 with coresponding training accuracy of 0.96 in the last epoch (no.

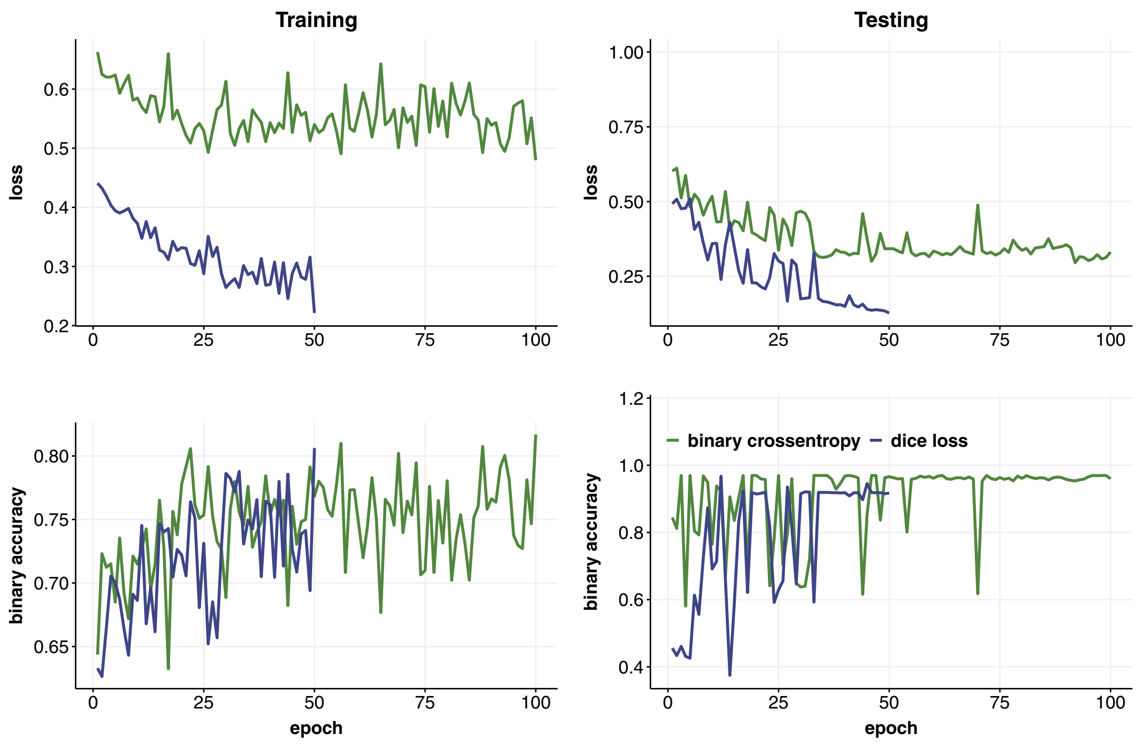


Figure 4.12: Training process for the high resolution data. The green line shows 100 epochs of training the CNN using a binary crossentropy loss and the blue line shows 50 epochs of training using a dice loss function. The CNN trained with the dice loss function also had two extra convolutional blocks and a higher number of filters.

100). After visualizing the predictions for the CNN trained using binary crossentropy loss it became clear that the loss function is too vulnerable towards class imbalances and another training was performed using dice loss. Dice loss function has already shown to be the most stable choice for the SRTM data (compare 4.5). The model trained with dice loss function was able to predict the occurrence of karst much more reliable at visual inspection of the pictures. Accuracies and losses cannot be directly compared in this case since the one-hot encoding therefore was chosen as the primary model to be discussed in this section. Both performances are converging towards an optimum unlike the development of the training for the SRTM data (Figure 4.2). The model trained with the dice loss function found a minimum loss of 0.2 during the training, which corresponds to a testing loss of 0.13 and accuracies of 0.81 and 0.92

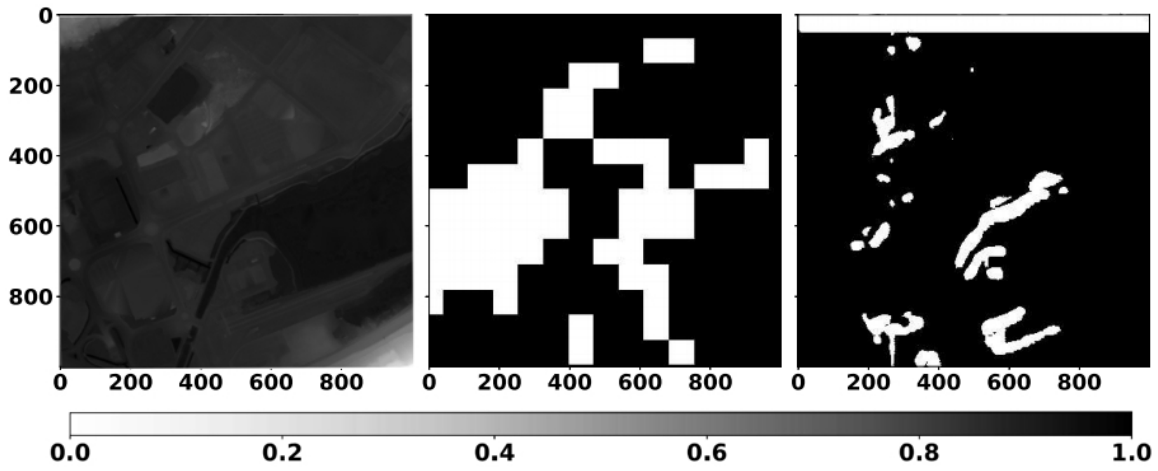


Figure 4.13: Performance of the CNN on the fifth picture of the testing data. The CNN was trained using dice loss function

for testing and training. The lowest loss for this model corresponds to the highest training accuracy and both of them were achieved after 50 epochs.

When comparing Figure 4.7 to 4.11 to the output given in Figure 4.13 the predictions are comparable with the augmented data rather than the random scatter in the raw SRTM data. Patches of karstified areas can be identified although they don't cover the whole extent of karst areas. The model is underestimating the occurrence of karst systematically, because of class imbalance.

QRF with SRTM Data

In figure 4.14 the resampled and rescaled values of elevation, slope and flow direction for Europe are visualized in the top row of the plot. The expected model output according to WOKAM is shown in the bottom left map. Table 4.2 summarized the dataset properties and accuracies for the QRF with SRTM data as well as the model for the Lidar data in comparison to a random number generator that accounts for the class propability. The SRTM training dataset consists of 2869 pixels, of which 31 % belong to class 1 and 69 % belong to class 0. The testing data contained 1913 pixels with a similar proportion between the classes (28 % and 72 %). Figure 4.14 shows the testing and training pixels, which were combined and remapped to the resampled raster extent. The accuracies reported for testing and training data is 0.73 and 0.72, which means that the QRF predicts the right class in 73/72 % of cases (figure 4.14, bottom middle). This would also be the case if it was predicting that there is no karst occurring all over Europe, but if we take a closer look at the pixels identified as karst in figure 4.14 the prediction is correct mostly for alpine regions, the Balkans and Spain. These are regions in which karst topography is highly developed. The random generator predicted the occurrence of karst in 60% of cases (figure 4.14, bottom right), performing worse than the QRF.

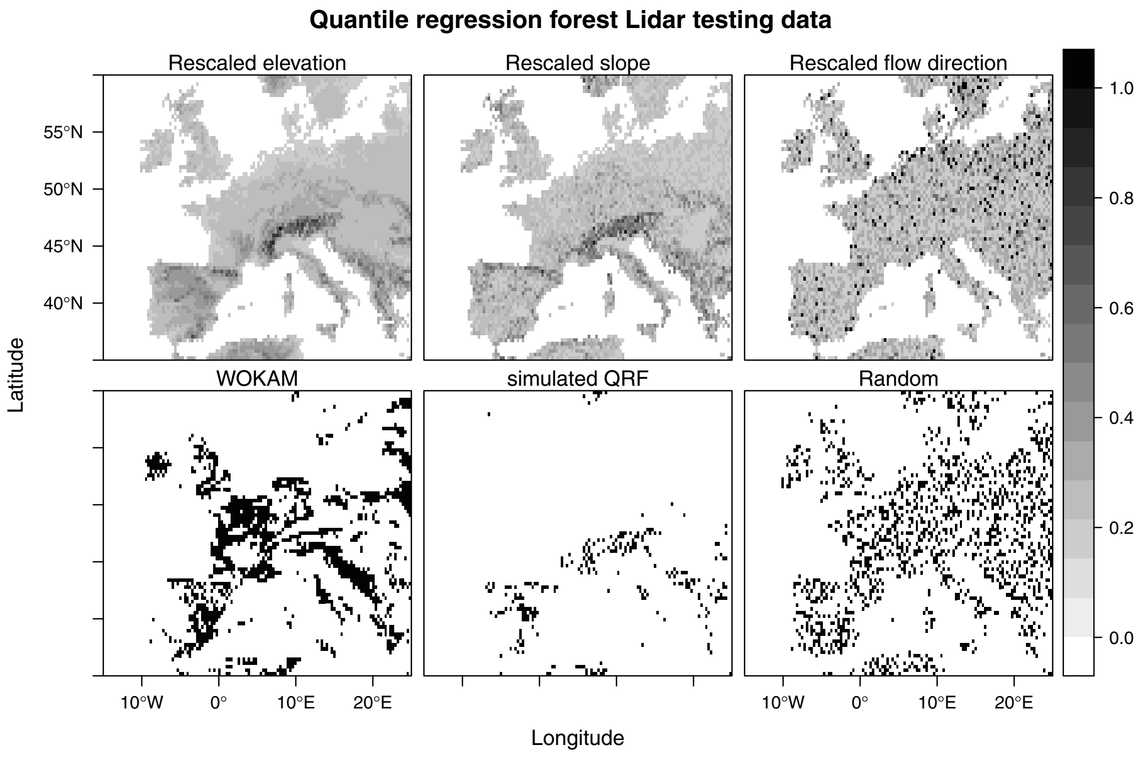


Figure 4.14: Visualization of the Input SRTM data for the QRF (top row) and the expected and predicted output as well as the output of a random generator. To split into testing and training random pixels were chosen from the data and the maps show training and testing data combined. For distinction of training and testing performance refer to table 4.2.

Table 4.2: Performance of the QRF on the SRTM and Lidar data in comparison to benchmark random generators.

	proportion class 1	proportion class 0	number of pixels	accuracy
QRF SRTM training	0.31	0.69	2869	0.73
QRF SRTM testing	0.28	0.72	1913	0.72
SRTM random				0.60
QRF lidar training	0.68	0.32	21536	0.81
QRF lidar testing	0.92	0.08	9700	0.92
Lidar random				0.91

QRF with Lidar Data

After resampling 39 tiles and removing missing values the training data contained 21536 values. Figure 4.15 shows the testing image for the QRF resampled to 10 000 pixels. Surface features such as streets are still visible at this resampling rate and especially slope values (figure 4.13, middle top) are outlining the features. The second row of rasters in figure 4.13 shows the ground-truth karst ocurrence and the simulation with random numbers. For this testing picture the QRF has classified a small number of pixels as not karstified, but they don't cover the same regions as the WOKAM. The simulated karst areas seem to follow lines across roads or fields along the DTM. The QRF reported very high accuracies for the Lidar data on the testing image (table 4.2), the training data was more difficult to simulate yielding an accuracy of 0.81. The very high training accuracy occurs only because 92% of the testing pixels are karstified.

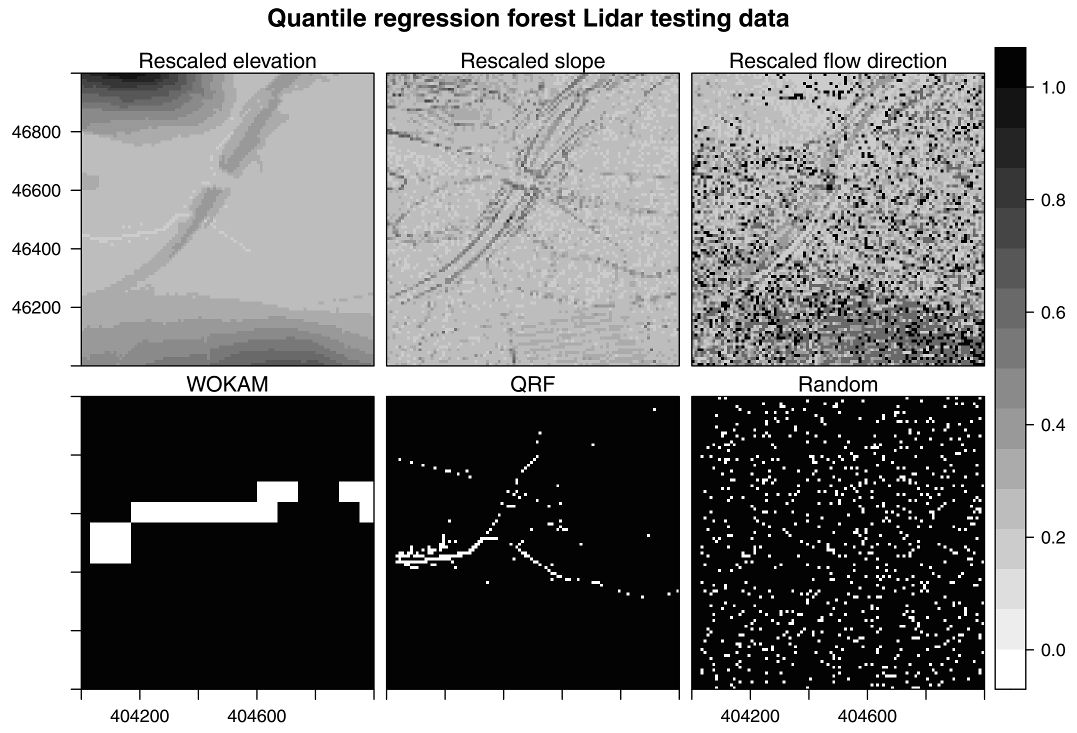


Figure 4.15: Maps of the resampled lidar tile used for testing the QRF trained on the Lidar data. The top row shows the Input data, the second row the predicted, expected and randomly generated values for this tile.

5. Discussion

5.1 Interpretation of the results

Due to the fact that the CNN was unable to reach a higher accuracy using the SRTM data and with a 5 % increase of differences the simulation accuracy improved significantly, it must be concluded that the SRTM data is not sufficient to predict karst areas. Especially the relationship between the loss function and testing accuracy is supporting the hypothesis that there is no way to model the occurrence of karst on the basis of SRTM data using a CNN, since the relationship in the training dataset doesn't correspond to the relationship between elevation and karst occurrence in the testing dataset. This may be a product of per sample rescaling, which couldn't be solved because a CNN cannot operate with the actual values and global rescaling led to the exploding gradients problem (which will be discussed in the next section). In any case the SRTM data has to be interpreted as insufficient for predicting karst areas.

The results considering the lidar data were promising. The CNN was able to extract features from the data and relate them to the karstified areas. The main concern considering this analysis is the batch size. Kingma and Ba (2014) who proposed the chosen optimizer suggest to use a batch size of 128 and other papers also report that the batch size influences the convergence towards the loss function minimum Keskar et al. (2016). By raising the batch size a better model performance could be assumed. Although the model achieved very high accuracy values, those have to be handled with care because of the class imbalances. Despite these concerns it can be concluded that the resolution is a main influencing factor for the network to be able to identify karstified areas.

The performance of the QRF on the SRTM data was only slightly better than the simulation generating random numbers. Nevertheless the regions identified by the QRF are plausible. A relation between the topography and the occurrence of karst was found especially for high altitude regions. For the QRF slope and flow direction had to be included in modelling the high resolution because the parameters used split the leaves of the trees had to be defined.

5.2 Uncertainty of the Approach

One of the biggest challenges in this project has been the preprocessing of the SRTM data. A dataset spanning the elevation range of Europe has a huge variation and the ReLU activation functions expect an input between 0 and 1. Now, if the elevation is normalized over the whole dataset, there is a large number of pictures that include only very low values, which causes the weights of the convolutional layers to converge towards infinity trying to compensate for that. In that case the loss function was not identifiable and the network cannot be trained. Even after addressing this problem by normalizing every image separately the weights were converging towards infinity. In this case the images associated with elevations close to sea level were indistinguishable from the ones in the alps considering the absolute elevation, which is not the case for a global normalization. To transform the data globally in a more suitable way, a stretch transformation (spreading the normalized values more evenly) was applied but it had no effect other than a minimum maximum scaling on the global data and had to be discarded. Another way to face the exploding gradients problem is to implement batch normalization layers after the activation function. In this case these additional layers didn't solve the issue. Every image had to be scaled individually and only after applying histogram equalization the network started to be numerically stable. Histogram equalization can be viewed as a way of increasing a pictures contrast. It is common practice to prepare images for a CNN in this way, but the ordinary application of CNNs is also limited to photos such as Satellite images or simple photographs of everyday objects or street scenes, in which case the increase in contrast is not interfering with measurements of a certain value such as elevation.

5.3 Transferability of the Approach

A possible reason for the failure of the application of the CNN for all of Europe and another factor influencing the successful application of the CNN on the Lidar data is the effect of spatial transfer Only images of the karst area in Slovenia, which show similar features throughout are compared. The relationship of testing and training accuracy for the Lidar data shows a higher positive correlation indicating that the patterns which occur in the training data are also present in the testing data. This shows that the approach is more promising if there is no spatial transfer included in the training and testing process. All successful applications of CNNs mentioned in the literature survey consider only small scales ranging from a geographical region (Marmanis et al., 2015) or cities such as Mumbai (Wurm et al., 2019).

6. Conclusion

In computer science the performance of deep learning algorithms is measured by their ability to segment pictures of clearly defined objects, which is a task the majority of humans are able to perform (Krizhevsky et al., 2012). The identification of objects or landforms in remote sensing images is a difficult task even for an expert in this field and deep learning algorithms may not be as skilled for more difficult computer vision tasks (Latifovic et al., 2018).

The results achieved with the networks trained in this study suggest that the application of a CNN to improve the state of mapping karstified regions is not possible yet. This conclusion is made under the current circumstances and availability of data. There is a relation between the topography and karstification which could be shown with the statistical analysis performed on the data and is also found since the CNN was able to yield slightly higher accuracy than a random classifier and the accuracy achieved by the quantile regression forest showed to be even higher. Still, with a 60 % probability of an area being classified correctly the predictions are not eligible to be an improvement of the current state of mapping and an approach based on expert knowledge should be preferred over a data driven approach.

It has been shown that the propability of specific, very distinct karst features such as dolines can be related to the topography (Latifovic et al., 2018; Artugyan and Urdea, 2016; Marjan Temovski and Ivica Milevski, 2015) but the identification of vast karstified areas via topographic parameters has not been shown before. It is questionable if the information of karst presence is present in a DEM. Since karstified areas show specific features apart from higher slope gradients and plateaus, such as the absence of surface waters and also very specific types of vegetation (Bonacci, 1987). These could be exploited using other data sources for which the CNN is more suitable and which show a bigger difference when comparing regions with and without karst.

Satellite images could be used to exploit the difference in landuse and vegetation and also the absence of surface waters could be extracted from satellite data. These further examinations are interesting approaches but due to the scarce availability of data even in countries that are mapped reliably in the WOKAM data. Similarly the Lidar data holds a promising approach to predict karst areas, but not to map them globally since Lidar data are rare and more rarely accessible. Furthermore not the whole potential of Lidar data has been exploited in this study. Maybe by including a vegetation model, which can be extracted from Lidar data, alongside the DTM predictions could be improved.

In any case the null hypothesis of this study has to be rejected. Given the current availability of earth surface data and the available computing power in this study, a CNN cannot be used to improve the state of mapping. It becomes clear that the application of ML algorithms is bound to the availability of large datasets and unfortunately freely available geodata and access to computers that can process such large datasets is not given at the moment.

Bibliography

Laurentiu Artugyan and Petru Urdea. Using digital elevation model (dem) in karst terrain analysis. study case: Anina mining area (banat mountains, romania). *Carpathian journal of earth and environmental sciences*, 11:55–64, 2016.

Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation, 2015. URL `http://arxiv.org/pdf/1511.00561v3`.

Trevor J. Barnes. Big data, little history. *Dialogues in Human Geography*, 3(3): 297–302, 2013. ISSN 2043-8206. doi: 10.1177/2043820613514323.

Thorsten Behrens, Karsten Schmidt, Robert A. MacMillan, and Raphael A. Viscarra Rossel. Multi-scale digital soil mapping with deep learning. *Scientific reports*, 8(1): 15244–15253, 2018. doi: 10.1038/s41598-018-33516-6.

Y. Bengio, Y. LeCun, C. Nohl, and C. Burges. Lerec: a nn/hmm hybrid for on-line handwriting recognition. *Neural computation*, 7(6):1289–1303, 1995. ISSN 0899-7667. doi: 10.1162/neco.1995.7.6.1289.

Ognjen Bonacci. *Karst Hydrology: With Special Reference to the Dinaric Karst*, volume 2 of *Springer Series in Physical Environment*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1987. ISBN 978-3-642-83165-2.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Zhao Chen, Augusto S. Auler, Michel Bakalowicz, David Drew, Franziska Griger, Jens Hartmann, Guanghui Jiang, Nils Moosdorf, Andrea Richts, Zoran Stevanovic, George Veni, and Nico Goldscheider. The world karst aquifer mapping project: concept, mapping procedure and map of europe. *Hydrogeology Journal*, 25(3):771–785, 2017. ISSN 1431-2174. doi: 10.1007/s10040-016-1519-3.

Anzi Ding, Qingyong Zhang, Xinmin Zhou, and Bicheng Dai. Automatic recognition of landslide based on cnn and texture change detection. In *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 444–448. IEEE, 2016. ISBN 978-1-5090-4423-8. doi: 10.1109/YAC.2016.7804935.

Carsten F. Dormann. *Parametrische Statistik: Verteilungen, maximum likelihood und GLM in R*. Statistik und ihre Anwendungen. Springer, Berlin and Heidelberg, 2013. ISBN 978-3-642-34786-3. doi: 10.1007/978-3-642-34786-3. URL `http://dx.doi.org/10.1007/978-3-642-34786-3`.

GDAL/OGR contributors. Gdal/ogr geospatial data abstraction software library, 2019. URL `https://gdal.org`.

John Gunn. *Encyclopedia of caves and karst science*. Taylor & Francis, 2004.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

Mohammad Havaei, Axel Davy, David Warde-Farley, Antoine Biard, Aaron Courville, Yoshua Bengio, Chris Pal, Pierre-Marc Jodoin, and Hugo Larochelle. Brain tumor segmentation with deep neural networks. *Medical image analysis*, 35:18–31, 2017. doi: 10.1016/j.media.2016.05.004.

Brandon Heung, Hung Chak Ho, Jin Zhang, Anders Knudby, Chuck E. Bulmer, and Margaret G. Schmidt. An overview and comparison of machine-learning techniques for classification purposes in digital soil mapping. *Geoderma*, 265:62–77, 2016. ISSN 0016-7061.

Robert J. Hijmans and Jacob van Etten. raster: Geographic analysis and modeling with raster data, 2012. URL `http://CRAN.R-project.org/package=raster`.

Nam Vu Hoai, Nguyen Manh Dung, and Soonghwan Ro. Sinkhole detection by deep learning and data association. In *2019 Eleventh International Conference* *on Ubiquitous and Future Networks (ICUFN)*, pages 211–213. IEEE, 2019. ISBN 978-1-7281-1340-1. doi: 10.1109/ICUFN.2019.8806136.

Wassily Hoeffding. A non-parametric test of independence. *Ann. Math. Statist.*, 19 (4):546–557, 1948. doi: 10.1214/aoms/1177730150.

B. K. P. Horn. Hill shading and the reflectance map. *Proceedings of the IEEE*, 69 (1):14–47, 1981. ISSN 1558-2256. doi: 10.1109/PROC.1981.11918.

Man-Sung Kang, Namgyu Kim, Jong Jae Lee, and Yun-Kyu An. Deep learning-based automated underground cavity detection using three-dimensional ground penetrating radar. *Structural Health Monitoring*, 45:1–13, 2019. ISSN 1475-9217. doi: 10.1177/1475921719838081.

Alex Kendall, Vijay Badrinarayanan, and Roberto and Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*, 2015.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima, 2016. URL `http://arxiv.org/pdf/1609.04836v2`.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL `http://arxiv.org/pdf/1412.6980v9`.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc, 2012. URL `http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf`.

Nataliia Kussul, Mykola Lavreniuk, Sergii Skakun, and Andrii Shelestov. Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, 14(5):778–782, 2017. ISSN 1545-598X. doi: 10.1109/LGRS.2017.2681128.

Rasim Latifovic, Darren Pouliot, and Janet Campbell. Assessment of convolution neural networks for surficial geology mapping in the south rae geological region, northwest territories, canada. *Remote Sensing*, 10(2):307–326, 2018. doi: 10.3390/rs10020307.

Fei-Fei Li. Convolutional neural networks for visual recognition. https://web.stanford.edu/class/cs231a/lectures/intro$_{c}$*nn.pdf*, 2019. 05.121.2019.

Lidar. high resolution digital terrain model of slovenia, 2010. URL `http://gis.arso.gov.si/evode/profile.aspx?id=atlas_voda_Lidar@Arso`.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Fully Convolutional Neural Networks for Semantic Segmentation*, pages 3431–3440, 2015. URL `http://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Long_Fully_Convolutional_Networks_2015_CVPR_paper.pdf`.

Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks, 2015. URL `http://arxiv.org/pdf/1511.06343v4`.

Marjan Temovski and Ivica Milevski. Dem based geomorphometric analyses of karst surface in the republic of macedonia. In Jaroslaw Jasiewicz, Zbigniew Zwoliński, Helena Mitasova, and Tomislav Hengl, editors, *Geomorphometry for Geosciences*, pages 65–68. Bogucki Wydawnictwo Naukowe, Adam Mickiewicz University in Poznań - Institute of Geoecology and Geoinformation, Poznań, Poland, 2015. ISBN 978-83-7986-059-3.

D. Marmanis, F. Adam, M. Datcu, T. Esch, and U. Stilla. Deep neural networks for above-ground detection in very high spatial resolution digital elevation models. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-3/W4:103–110, 2015. doi: 10.5194/isprsannals-II-3-W4-103-2015.

Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`.

Nicolai Meinshausen. Quantile regression forests. *Journal of Machine Learning Research*, 7(Jun):983–999, 2006.

Muhammed Zeynel Ozturk, Mehmet Şener, Mehmet Sener, and Mesut Şimşek. Structural controls on distribution of dolines on mount anamas (taurus mountains, turkey). *Geomorphology*, 317, 05 2018. doi: 10.1016/j.geomorph.2018.05.023.

José Padarian, Budiman Minasny, and Alex B. McBratney. Using deep learning for digital soil mapping. *SOIL*, 5(1):79–89, 2019. doi: 10.5194/soil-5-79-2019.

Mario Parise and Mariangela Sammarco. The historical use of water resources in karst. *Environmental Earth Sciences*, 74(1):143–152, 2015. ISSN 1866-6280. doi: 10.1007/s12665-014-3685-8.

Mario Parise, Franci Gabrovsek, Georg Kaufmann, and Natasa Ravbar. Recent advances in karst research: from theory to fieldwork and applications. *Geological*

Society, London, Special Publications, 466(1):1–24, 2018. ISSN 0305-8719. doi: 10.1144/SP466.26.

QGIS Development Team. Qgis geographic information system, 2009. URL `http://qgis.osgeo.org`.

R Development Core Team. R: A language and environment for statistical computing, 2008. URL `http://www.R-project.org`.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. URL `http://arxiv.org/pdf/1505.04597v1`.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386–408, 1958.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. ISSN 0028-0836. doi: 10.1038/323533a0.

Reuben R Shamir, Yuval Duchin, Jinyoung Kim, Guillermo Sapiro, and Noam Harel. *Continuous Dice Coefficient: a Method for Evaluating Probabilistic Segmentations*. Cold Spring Harbor Laboratory, April 2018. doi: 10.1101/306977. URL `https://doi.org/10.1101/306977`.

Chaopeng Shen, Eric Laloy, Amin Elshorbagy, Adrian Albert, Jerad Bales, Fi-John Chang, Sangram Ganguly, Kuo-Lin Hsu, Daniel Kifer, Zheng Fang, Kuai Fang, Dongfeng Li, Xiaodong Li, and Wen-Ping Tsai. Hess opinions: Incubating deep-learning-powered hydrologic science advances as a community. *Hydrology and Earth System Sciences*, 22(11):5639–5656, 2018. doi: 10.5194/hess-22-5639-2018.

Shuttle Radar Topography Mission. Resampled srtm data, spatial resolution approximately 250 meter on the line of the equator, 2000. URL `http://srtm.csi.cgiar.org/srtmdata/`.

Xiaodong Song, Ganlin Zhang, Feng Liu, Decheng Li, Yuguo Zhao, and Jinling Yang. Modeling spatio-temporal distribution of soil moisture by deep learning-based cellular automata model. *Journal of Arid Land*, 8(5):734–748, 2016. ISSN 1674-6767. doi: 10.1007/s40333-016-0049-0.

Kamal Taheri, Himan Shahabi, Kamran Chapi, Ataollah Shirzadi, Francisco Gutiérrez, and Khabat Khosravi. Sinkhole susceptibility mapping: A comparison between bayes–based machine learning algorithms. *Land Degradation & Development*, 30(7): 730–745, 2019. ISSN 1085-3278. doi: 10.1002/ldr.3255.

Travis Oliphant. Numpy: A guide to numpy, 2006–. URL `http://www.numpy.org/`.

Hristos Tyralis, Georgia Papacharalampous, and Andreas Langousis. A brief review of random forests for water scientists and practitioners and their recent history in water resources. *Water*, 11(5):910, 2019.

Margaret F. J. Wilson, Brian O'Connell, Colin Brown, Janine C. Guinan, and Anthony J. Grehan. Multiscale terrain analysis of multibeam bathymetry data for habitat mapping on the continental slope. *Marine Geodesy*, 30(1-2):3–35, 2007. doi: 10.1080/01490410701295962.

Michael Wurm, Thomas Stark, Xiao Xiang Zhu, Matthias Weigand, and Hannes Taubenböck. Semantic segmentation of slums in satellite images using transfer learning on fully convolutional neural networks. *ISPRS Journal of Photogrammetry and Remote Sensing*, 150:59–69, 2019. ISSN 09242716. doi: 10.1016/j.isprsjprs.2019.02.006.

Qizhe Xie, Eduard Hovy, Minh-Thang Luong, and Quoc Le V. Self-training with noisy student improves imagenet classification, 2019. URL `http://arxiv.org/pdf/1911.04252v1`.

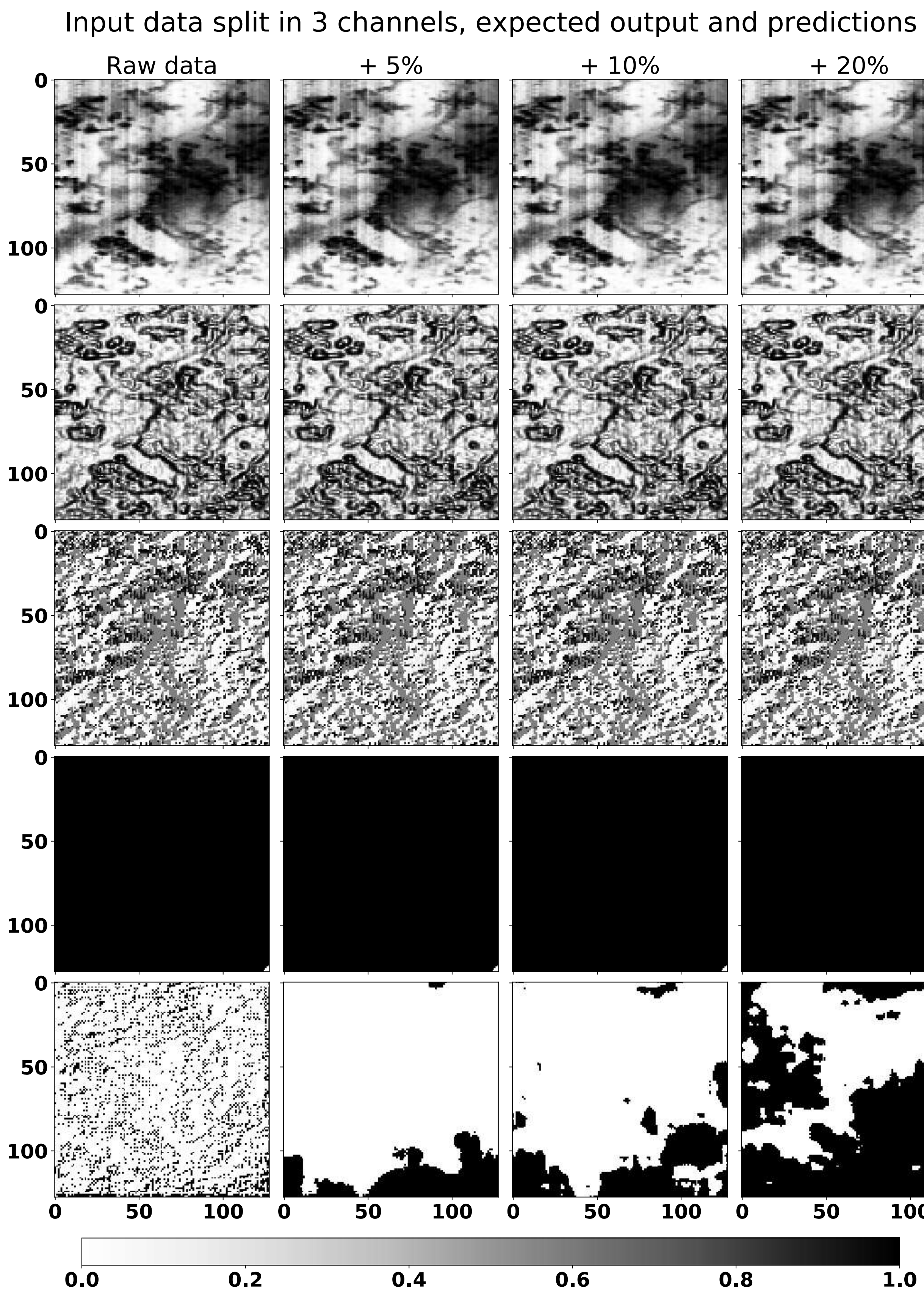
E.-Q. Xu, H.-Q. Zhang, and M.-X. Li. Object–based mapping of karst rocky desertification using a support vector machine. *Land Degradation & Development*, 26(2): 158–167, 2015. ISSN 1085-3278. doi: 10.1002/ldr.2193.

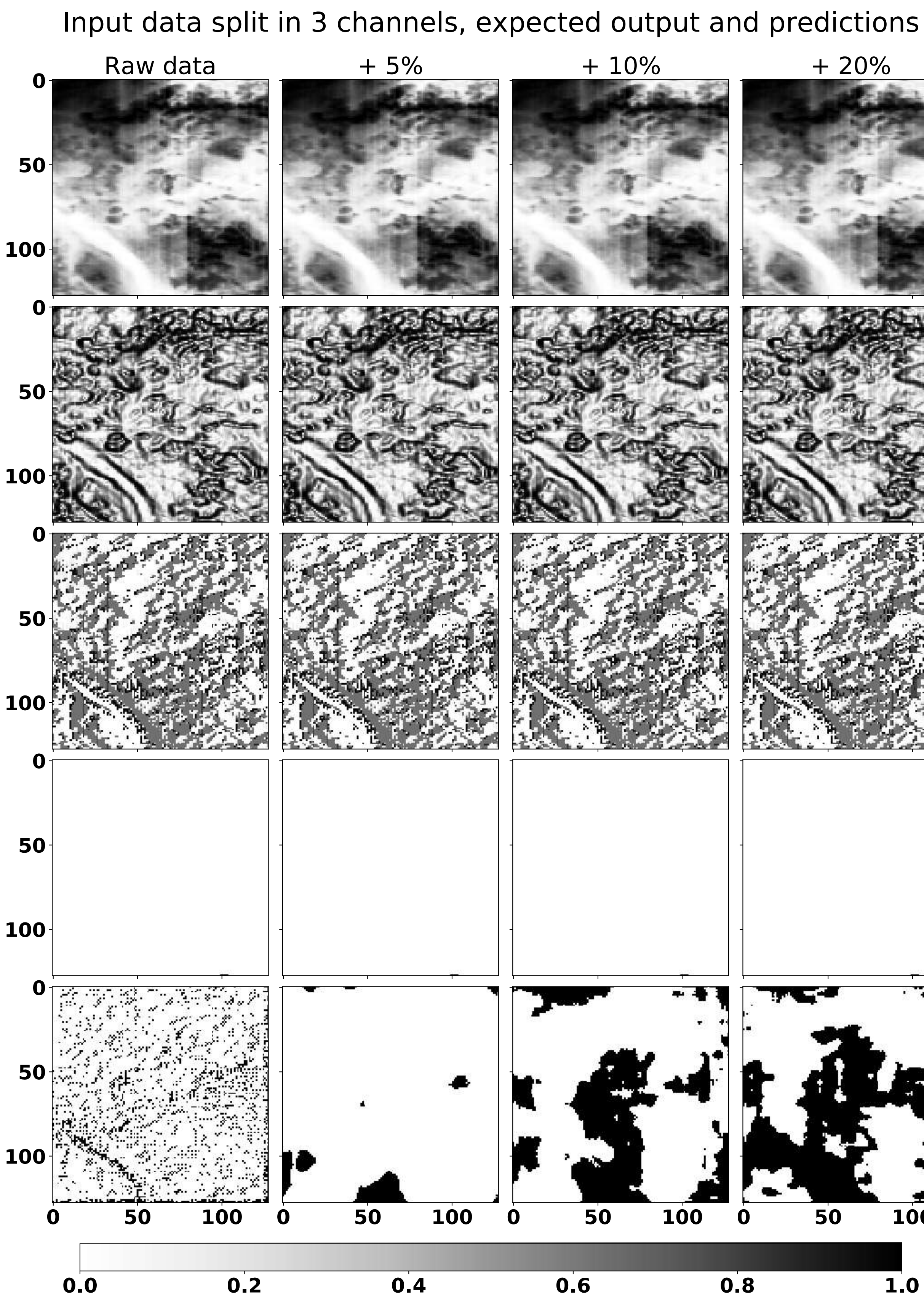
Qing Zhu, Yeting Zhang, and Fengchun Li. Three-dimensional tin algorithm for digital terrain modeling. *Geo-spatial Information Science*, 11(2):79–85, 2008. doi: 10.1007/s11806-008-0043-6. URL `https://doi.org/10.1007/s11806-008-0043-6`.

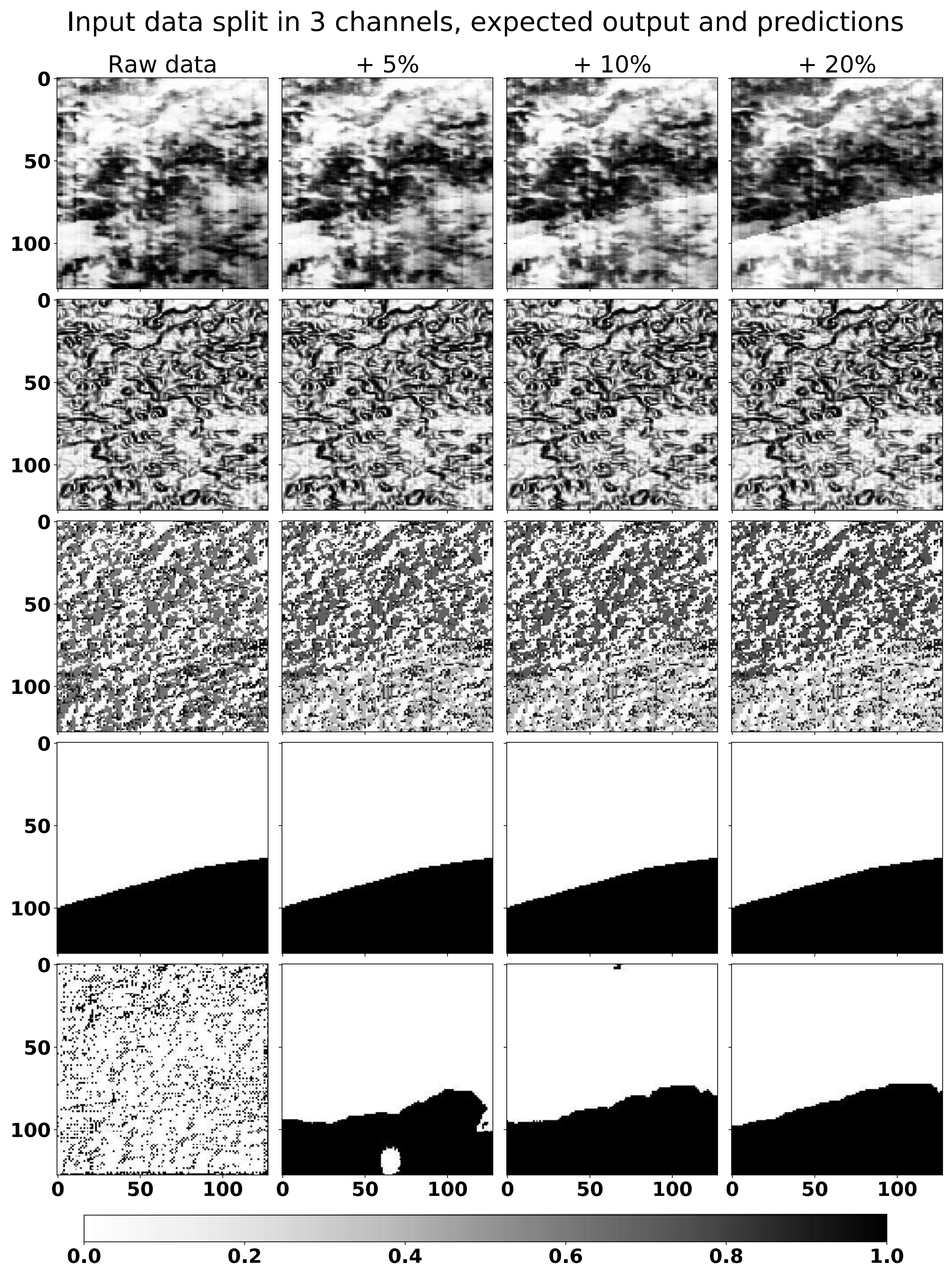
Xiao Xiang Zhu, Devis Tuia, Lichao Mou, Gui-Song Xia, Liangpei Zhang, Feng Xu, and Friedrich Fraundorfer. Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine*, 5(4):8–36, 2017. ISSN 2473-2397. doi: 10.1109/MGRS.2017.2762307.

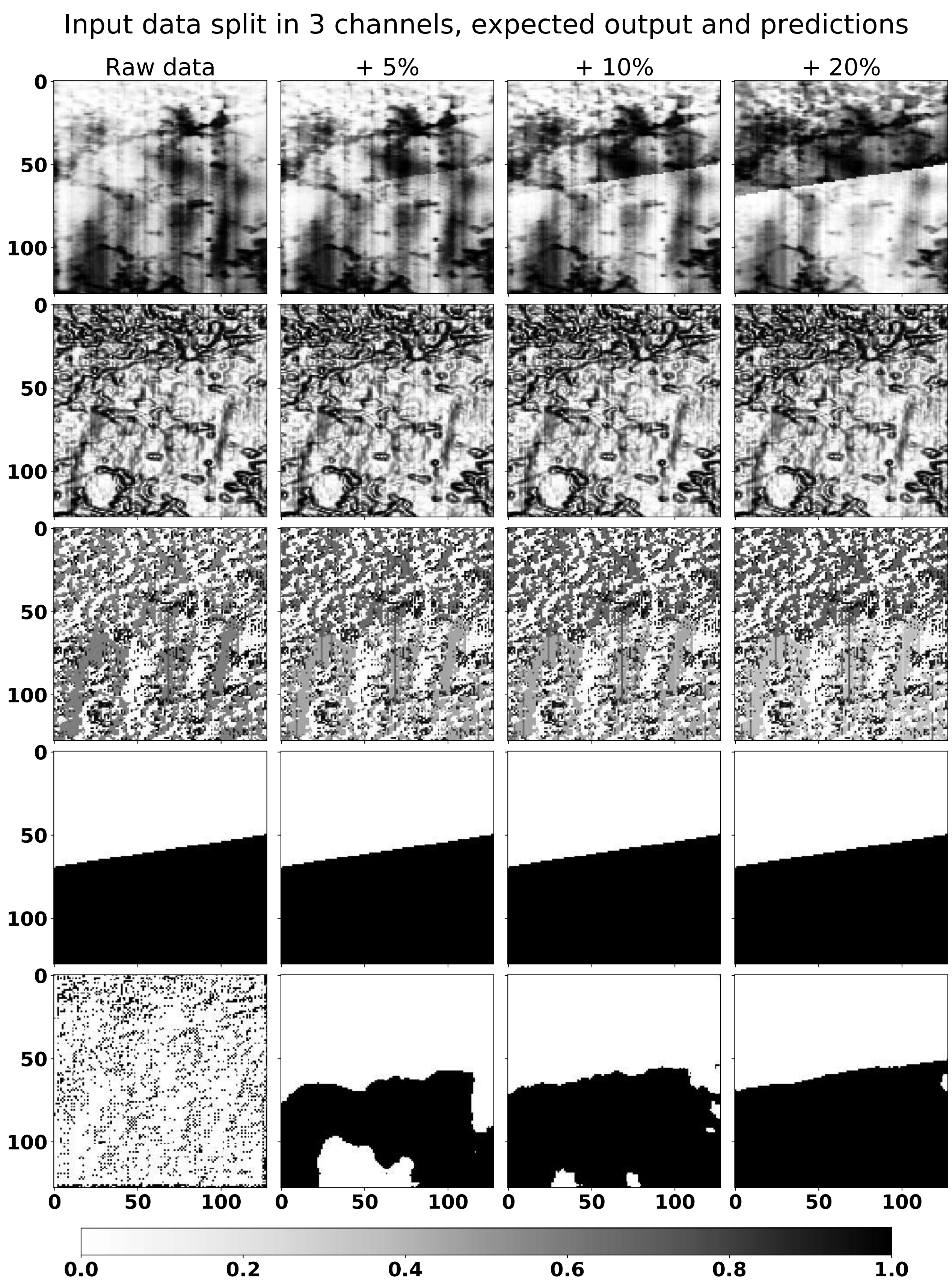
A. Appendix

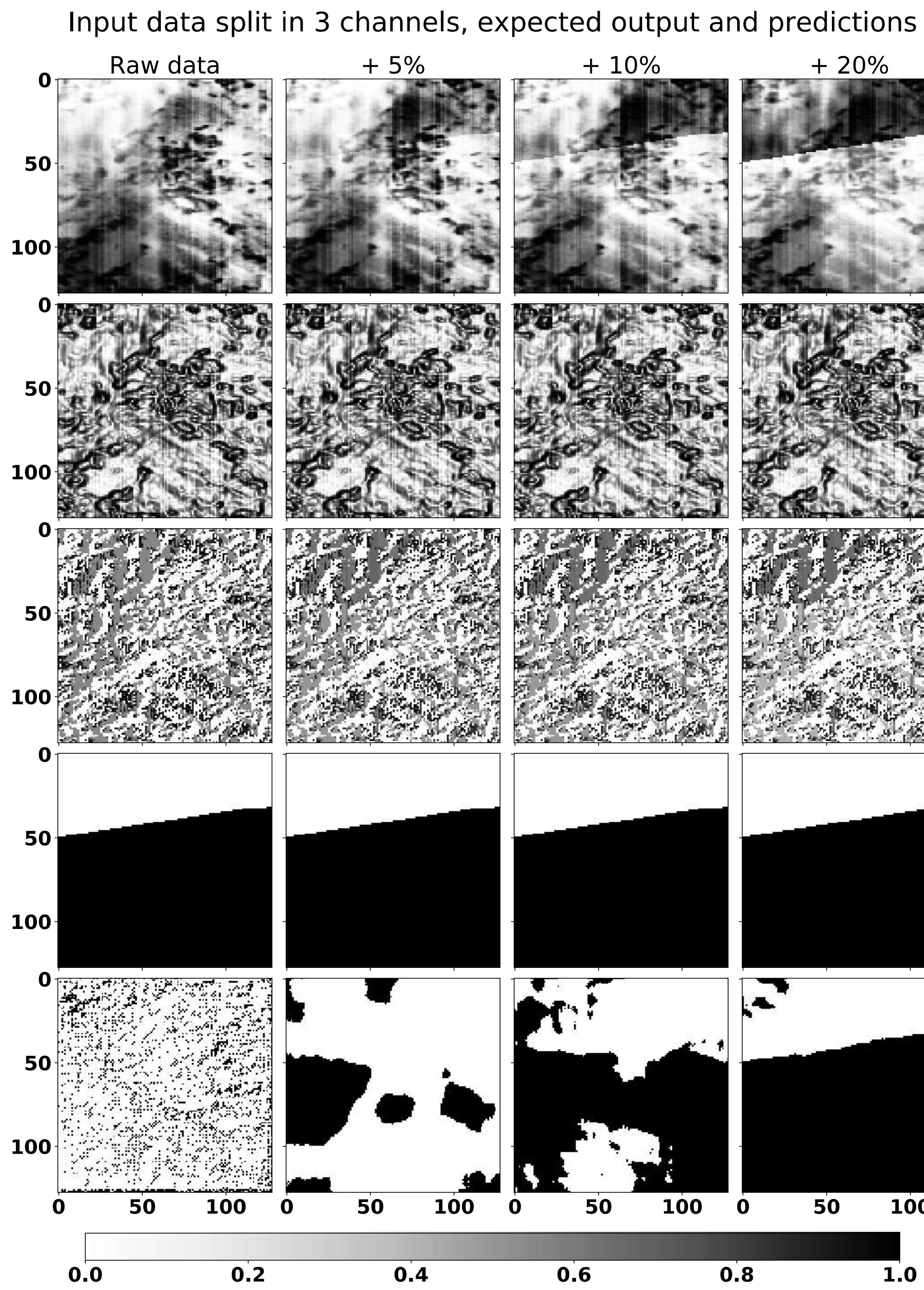
A.1 CNN training SRTM data


Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

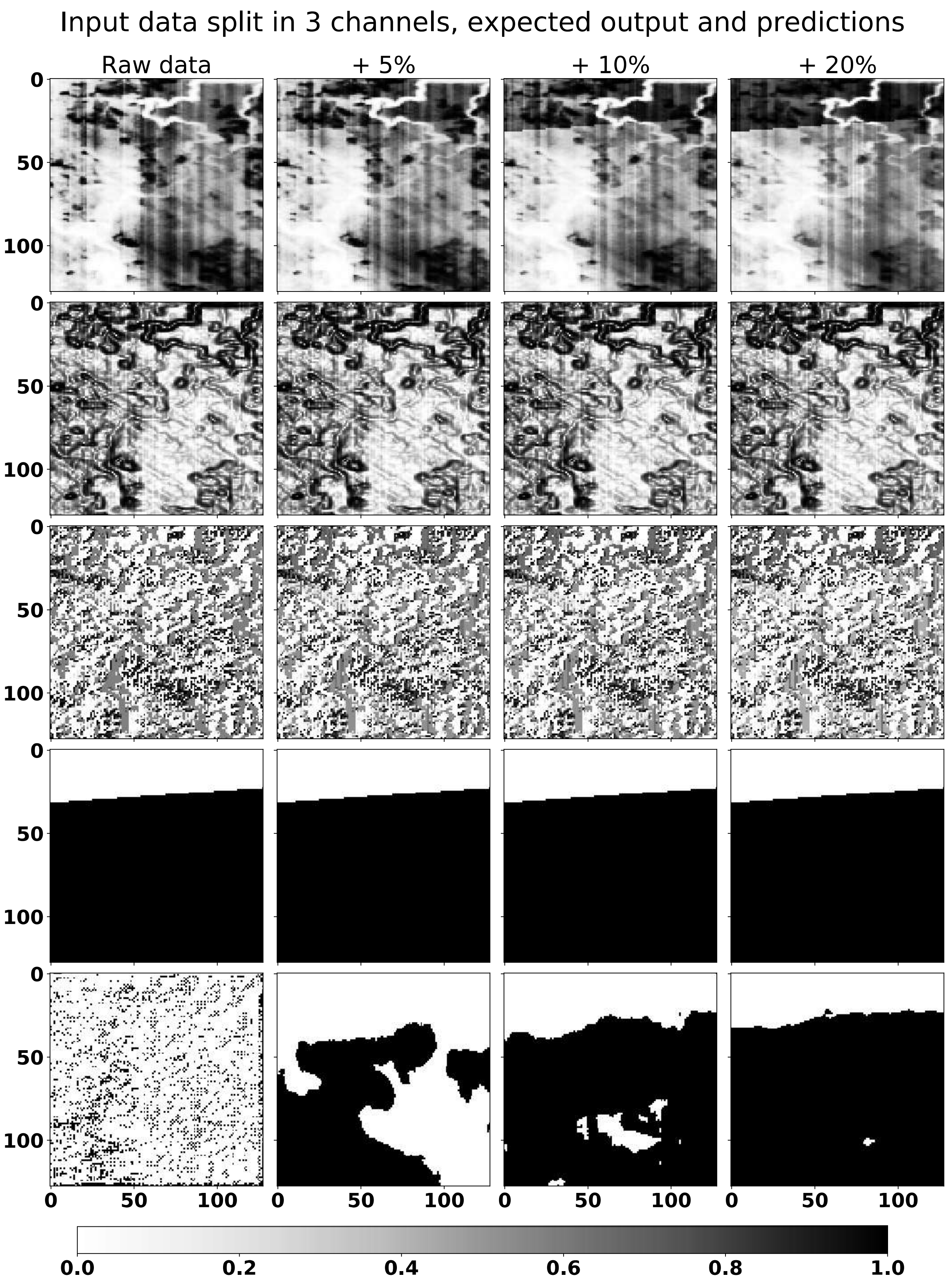

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

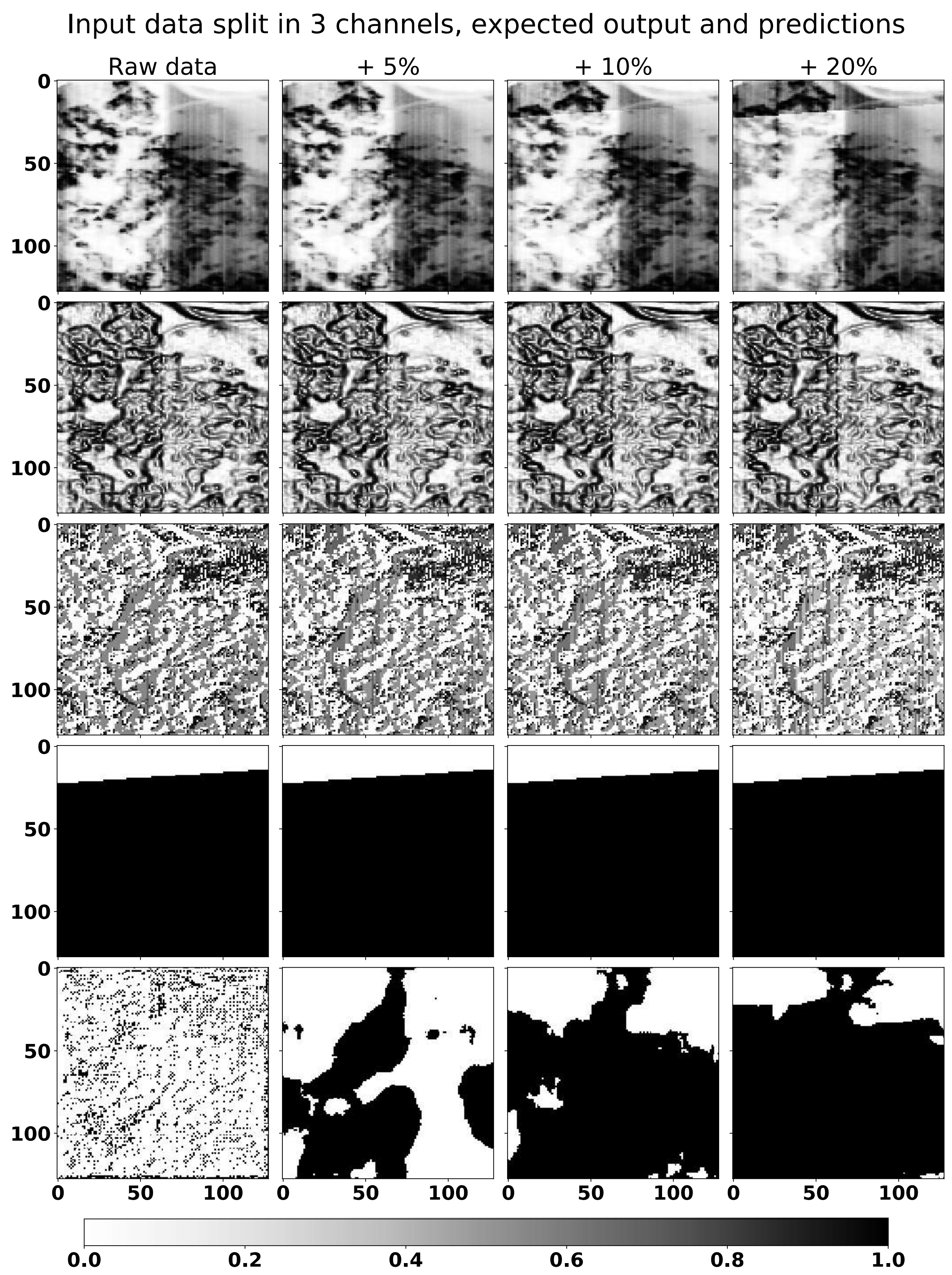

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

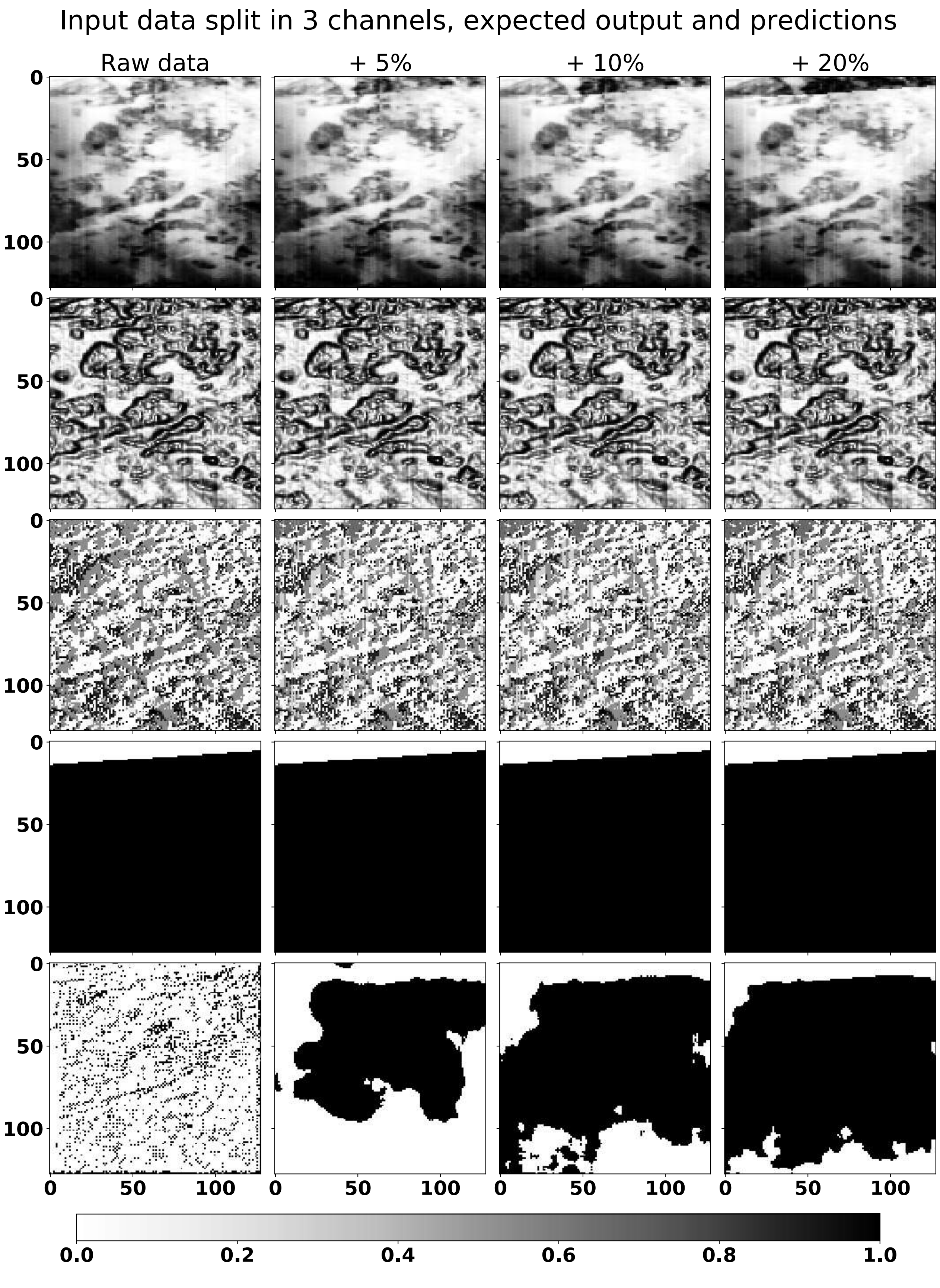

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

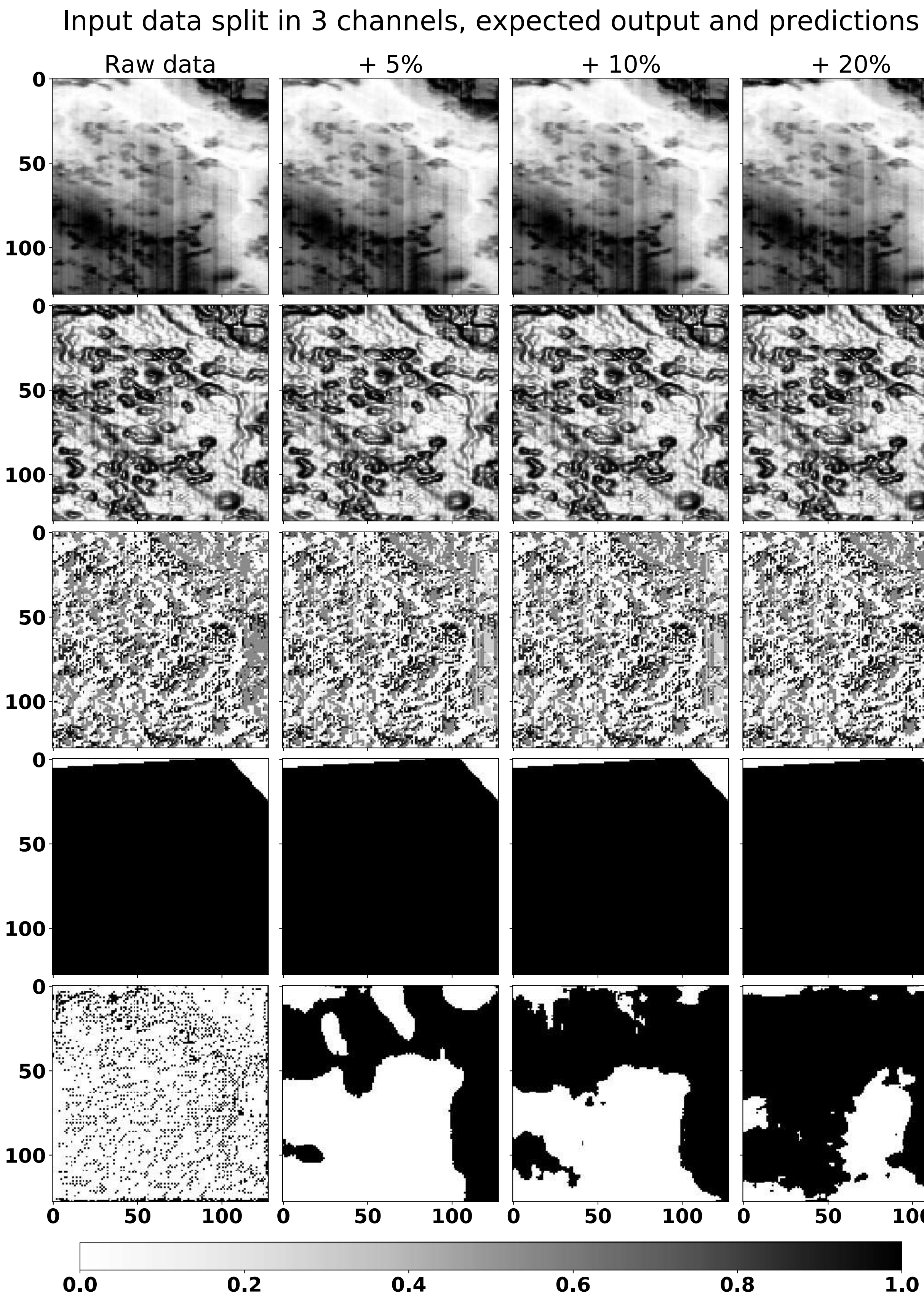


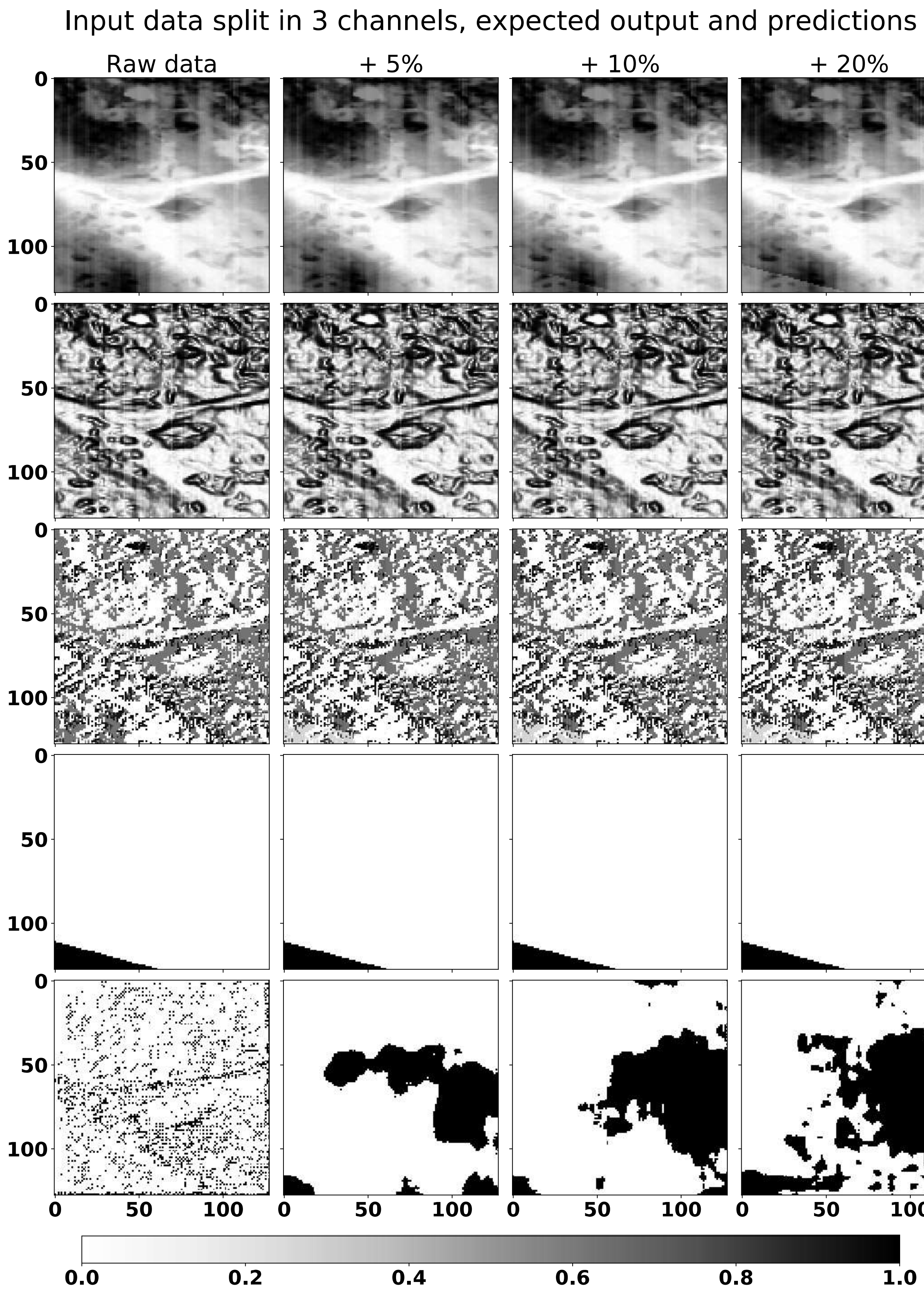
Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0


Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0


Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

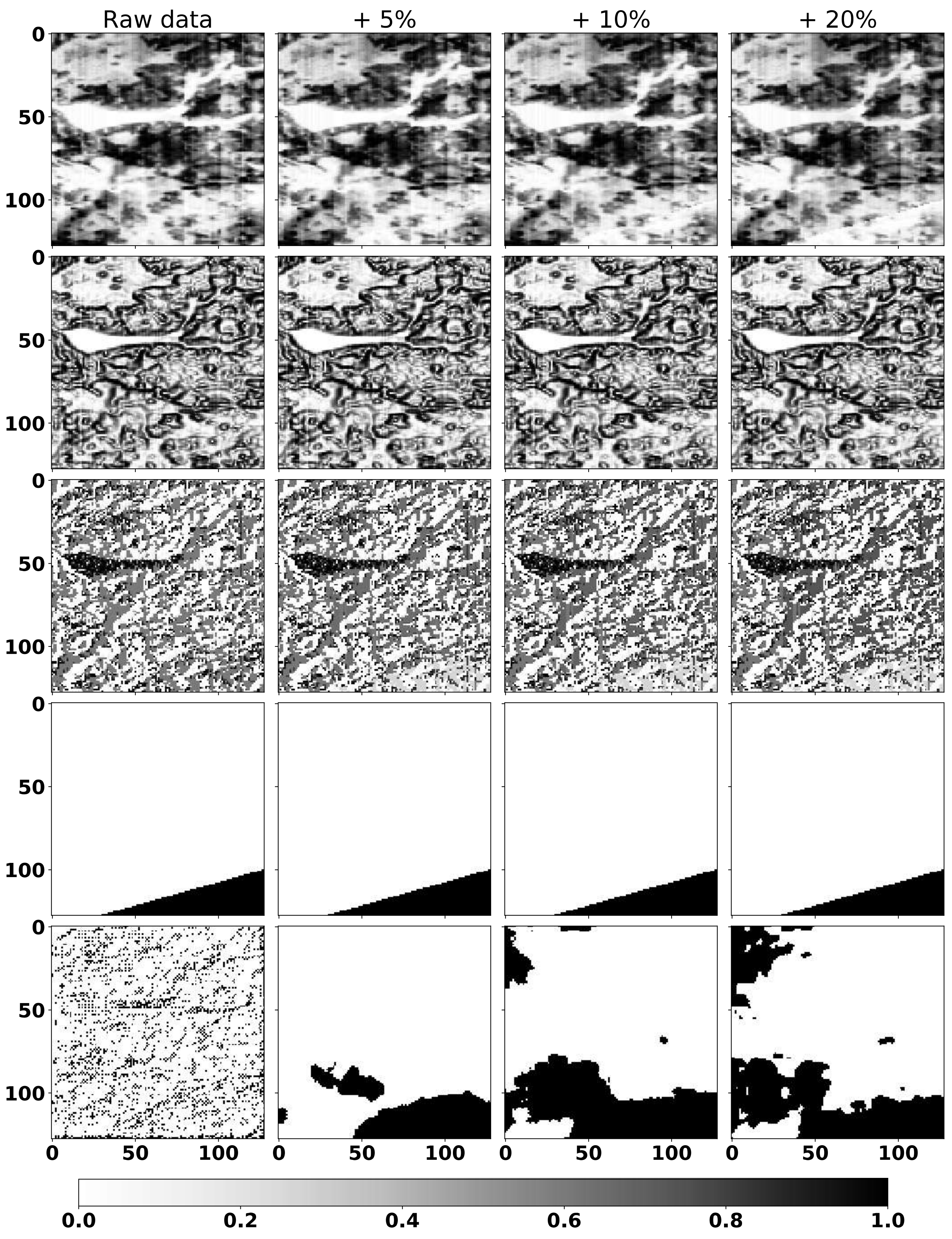

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

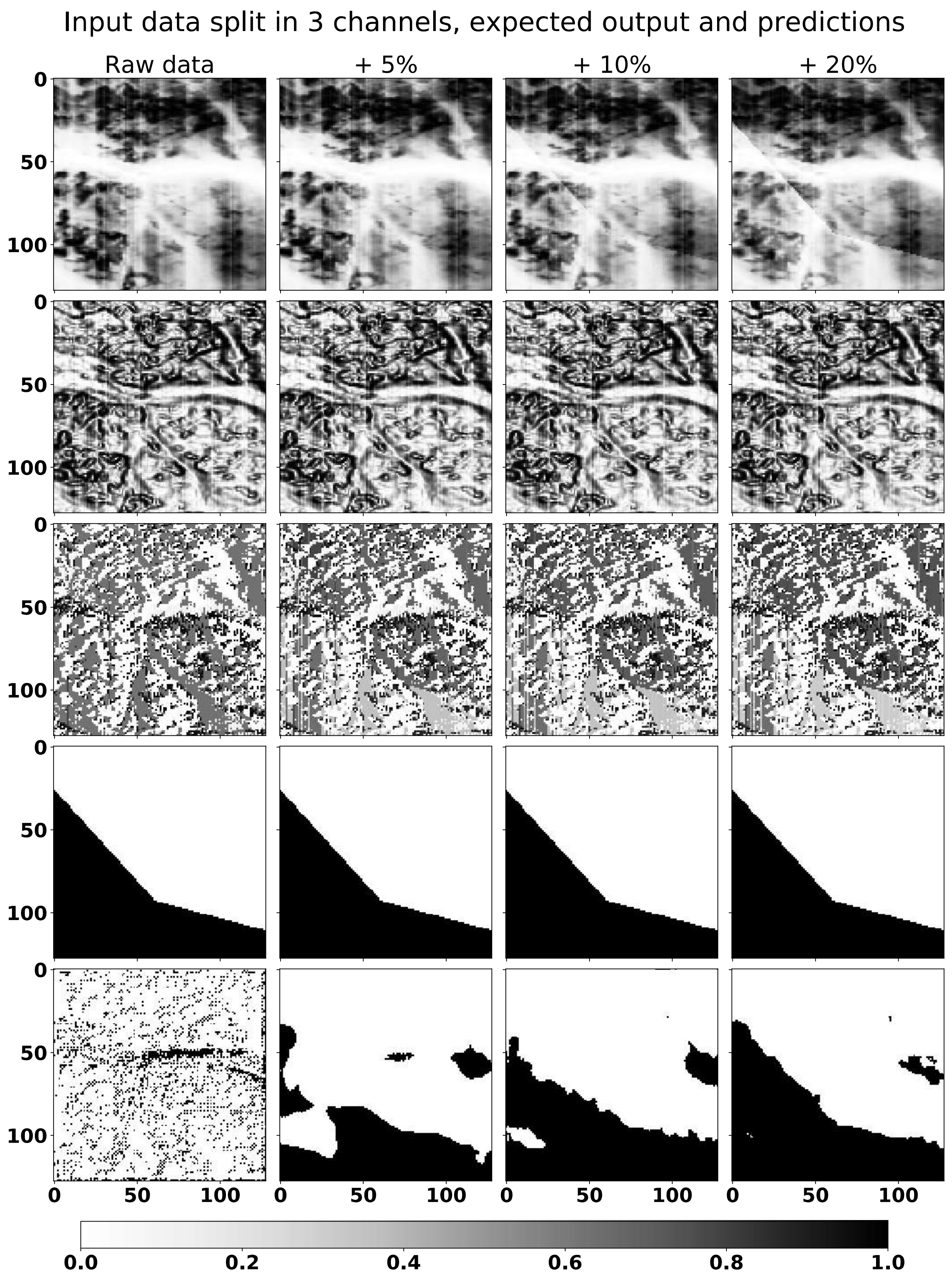

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

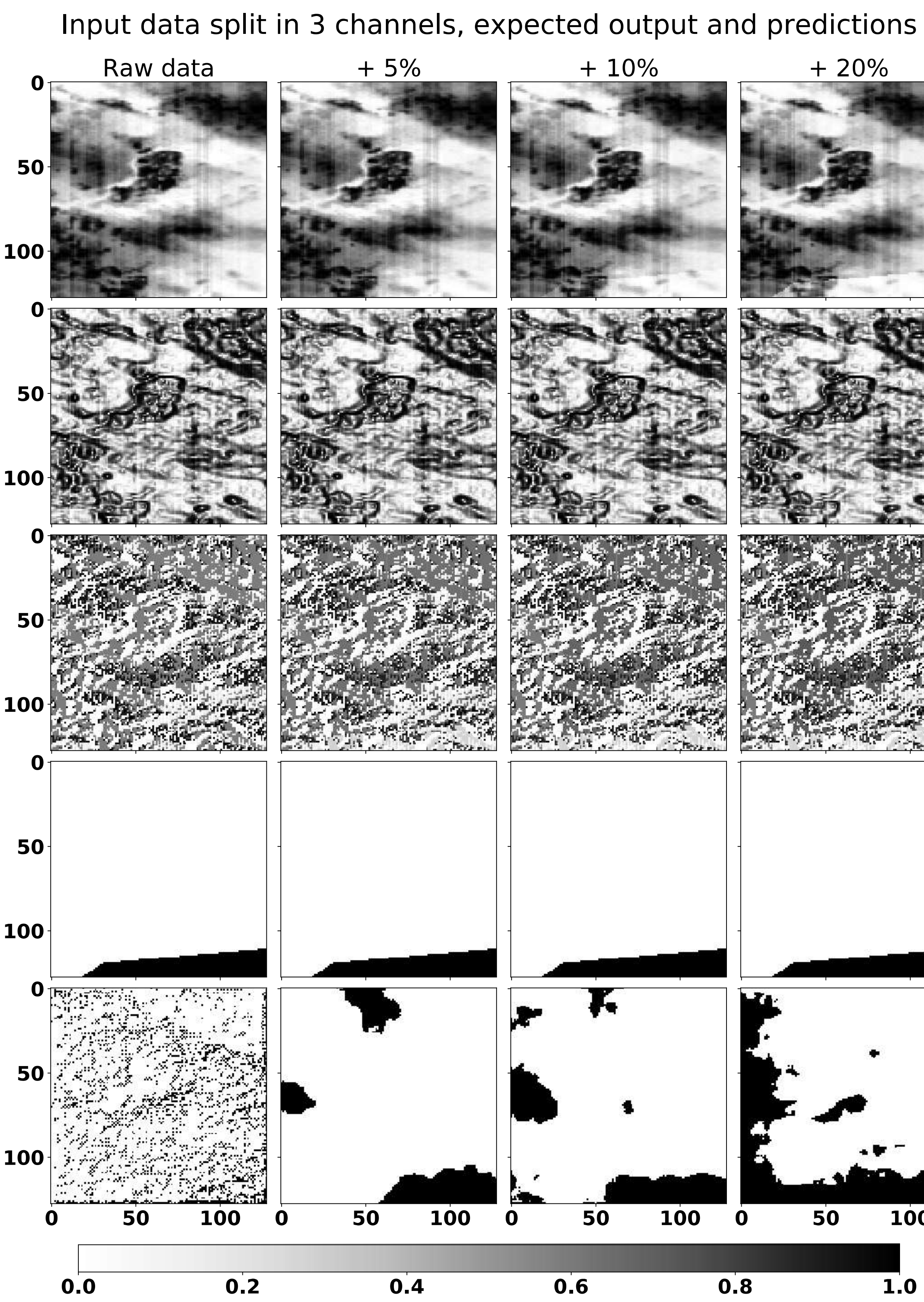

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

A.2 CNN testing SRTM data

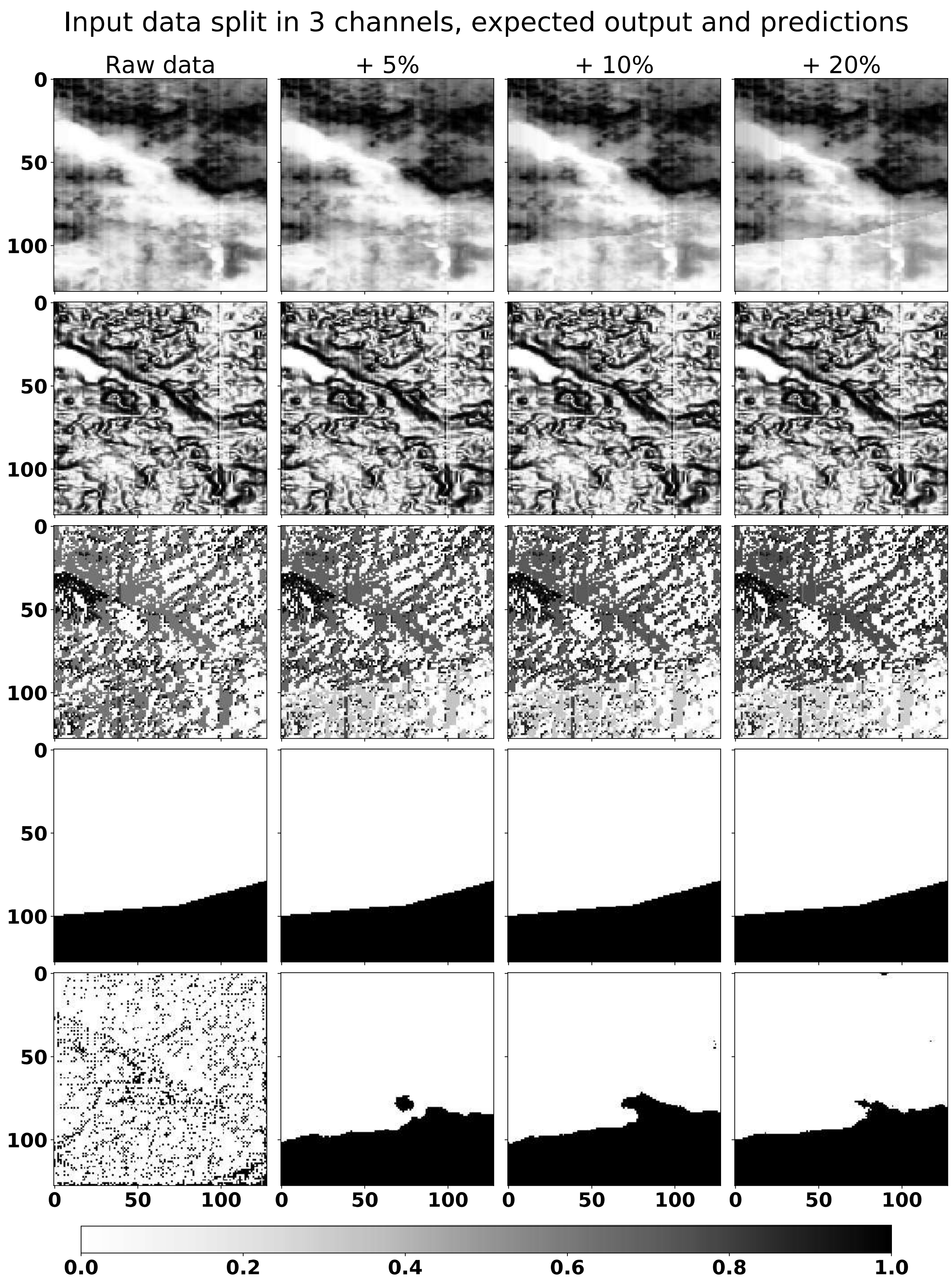
Input data split in 3 channels, expected output and predictions

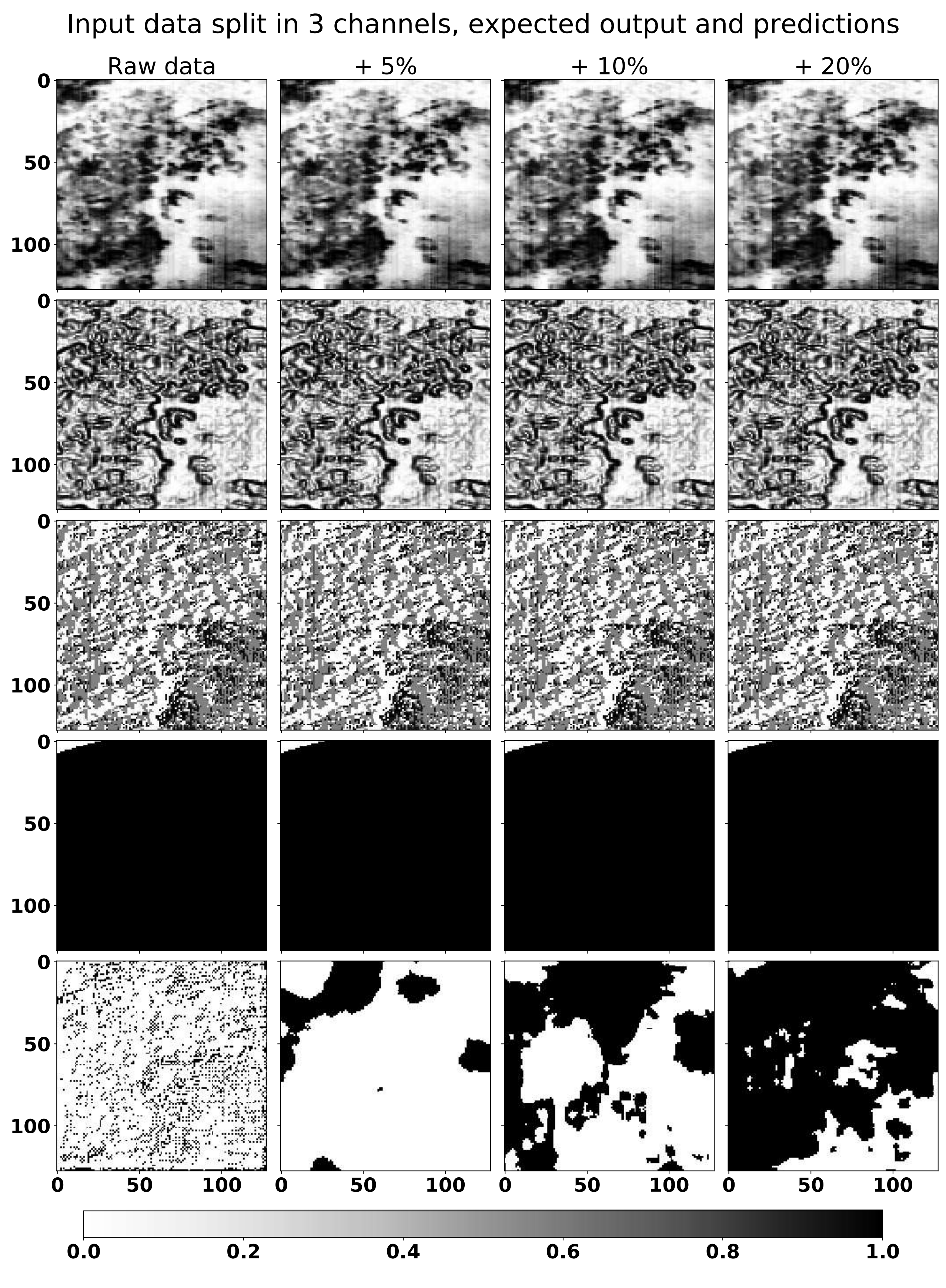


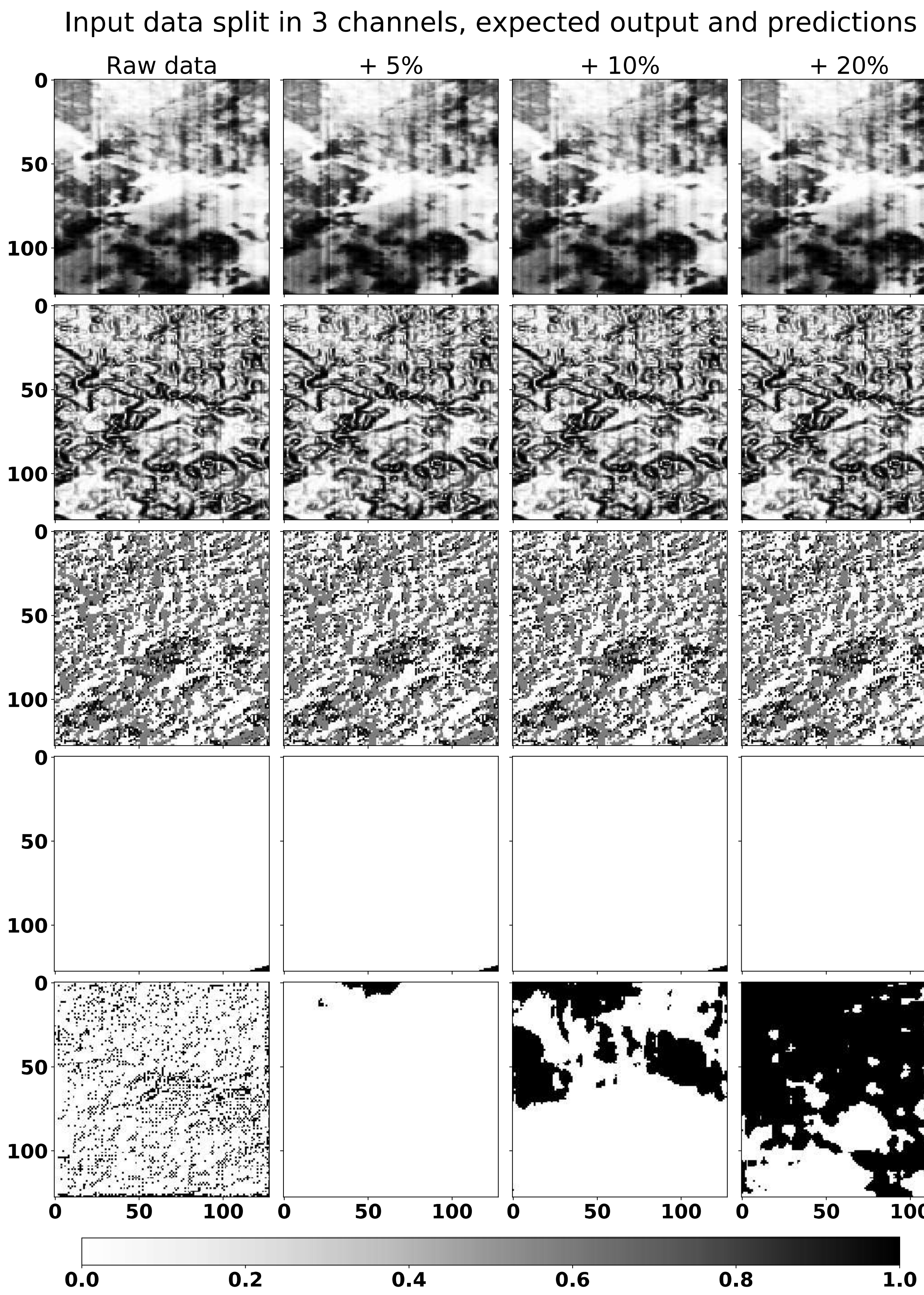

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

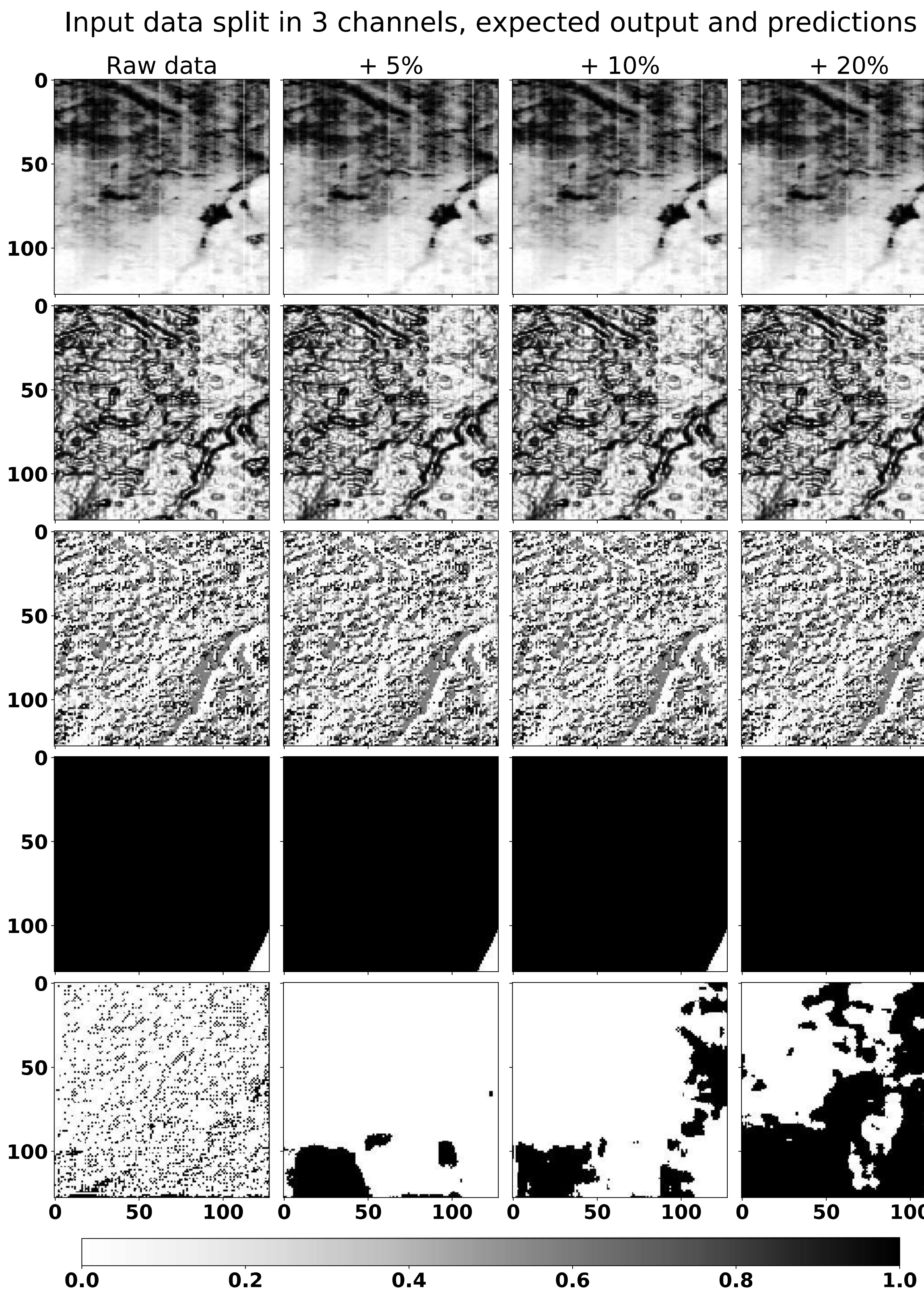


Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

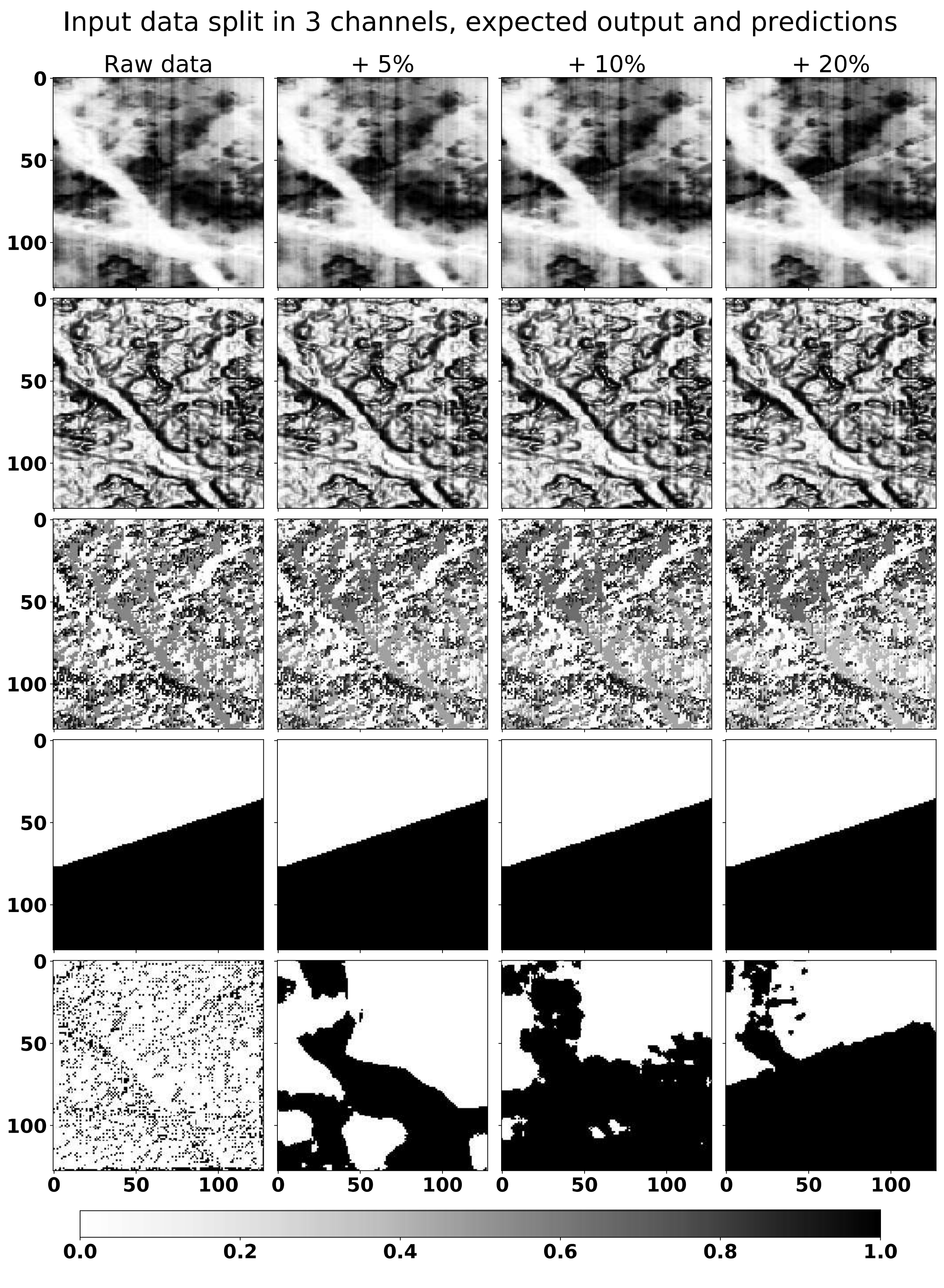

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

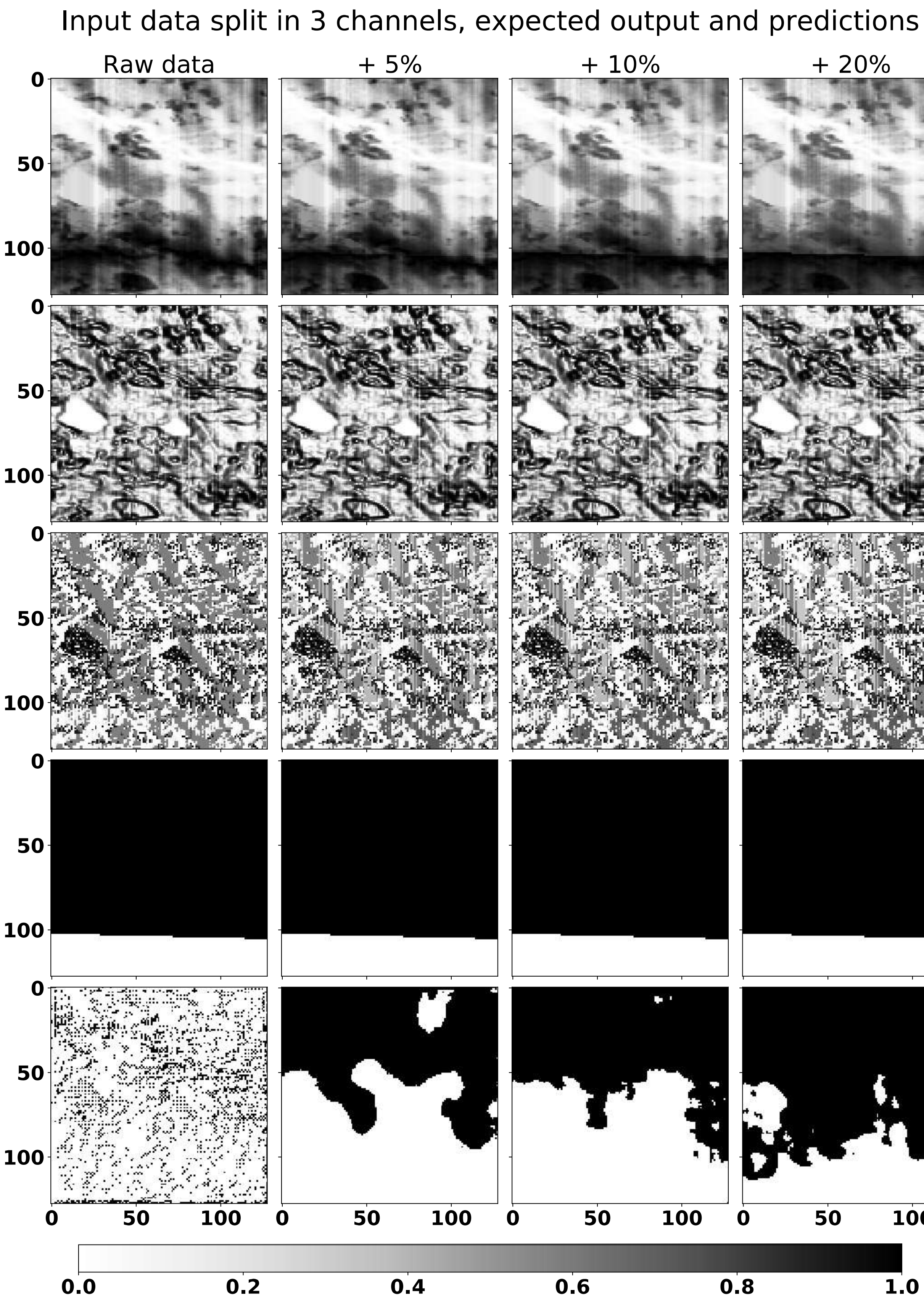

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

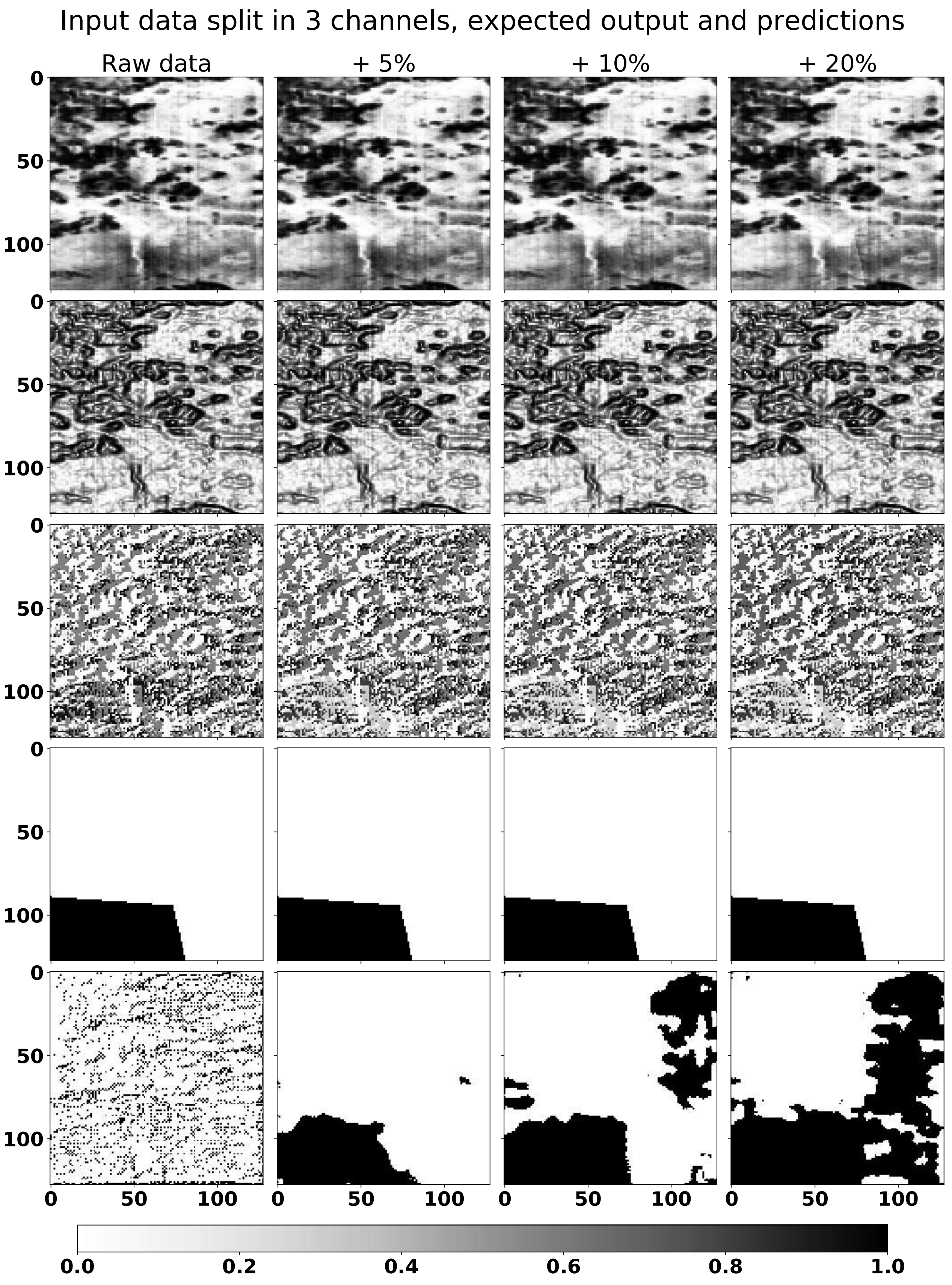

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0



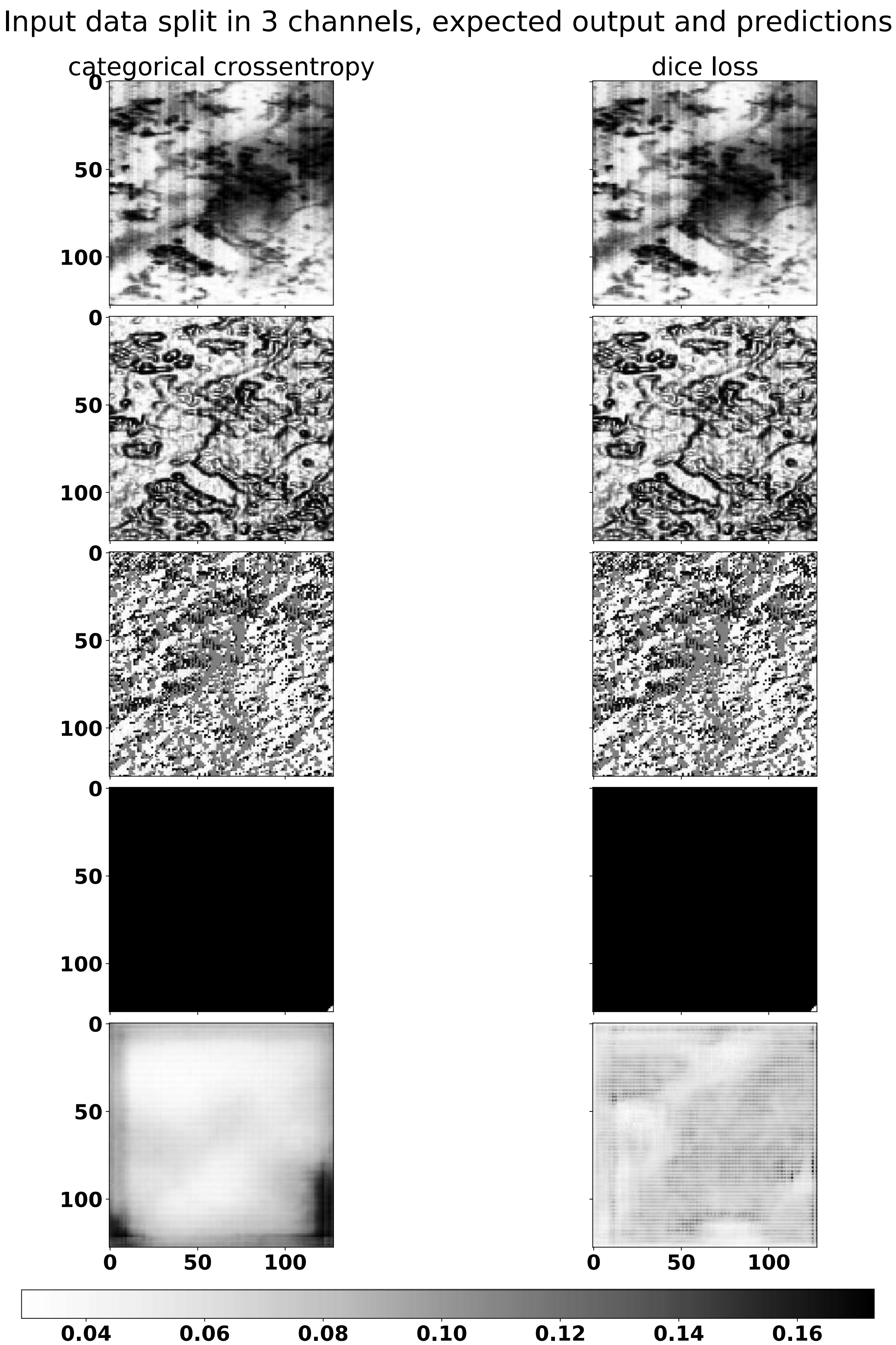
Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

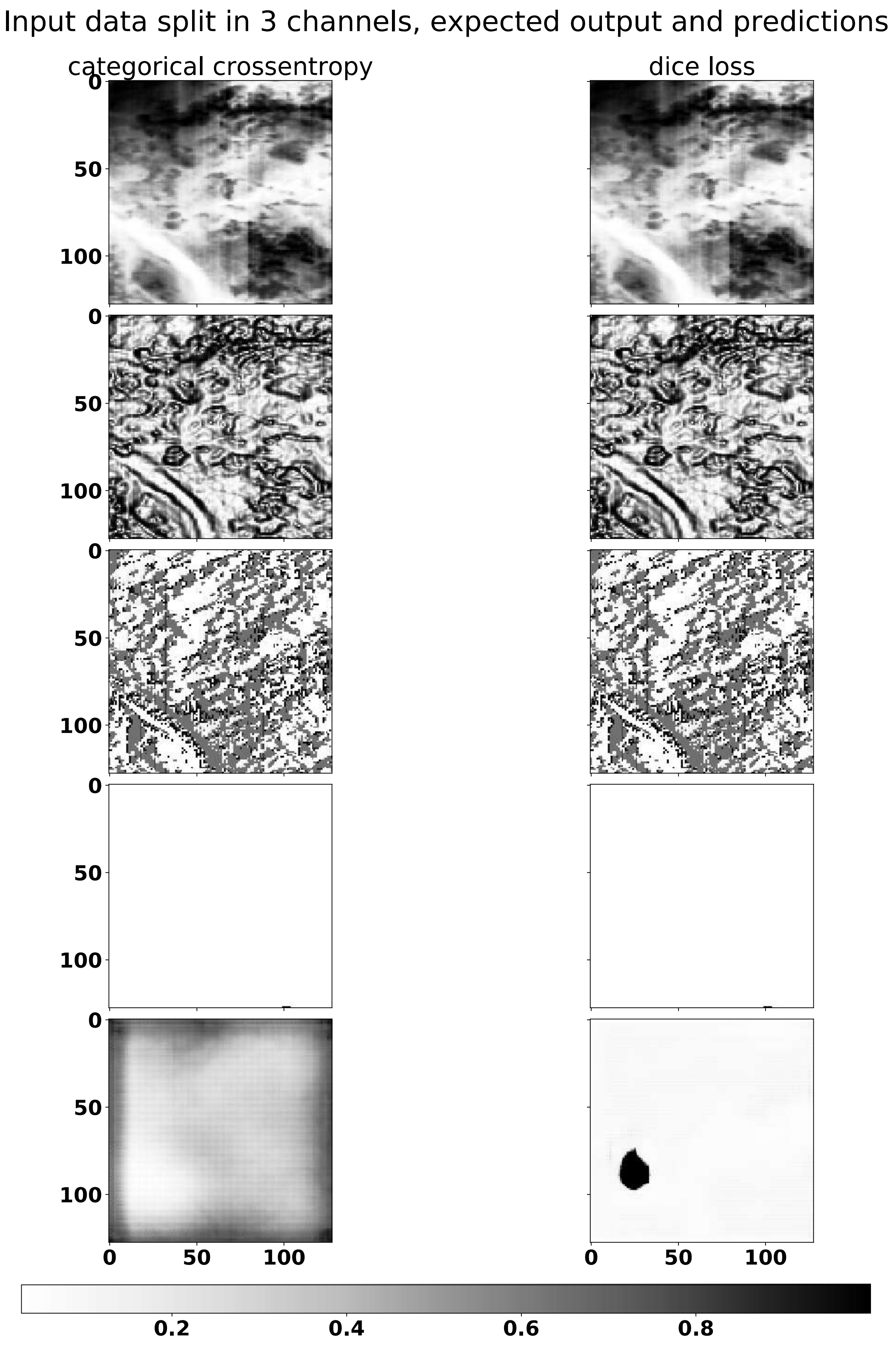

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

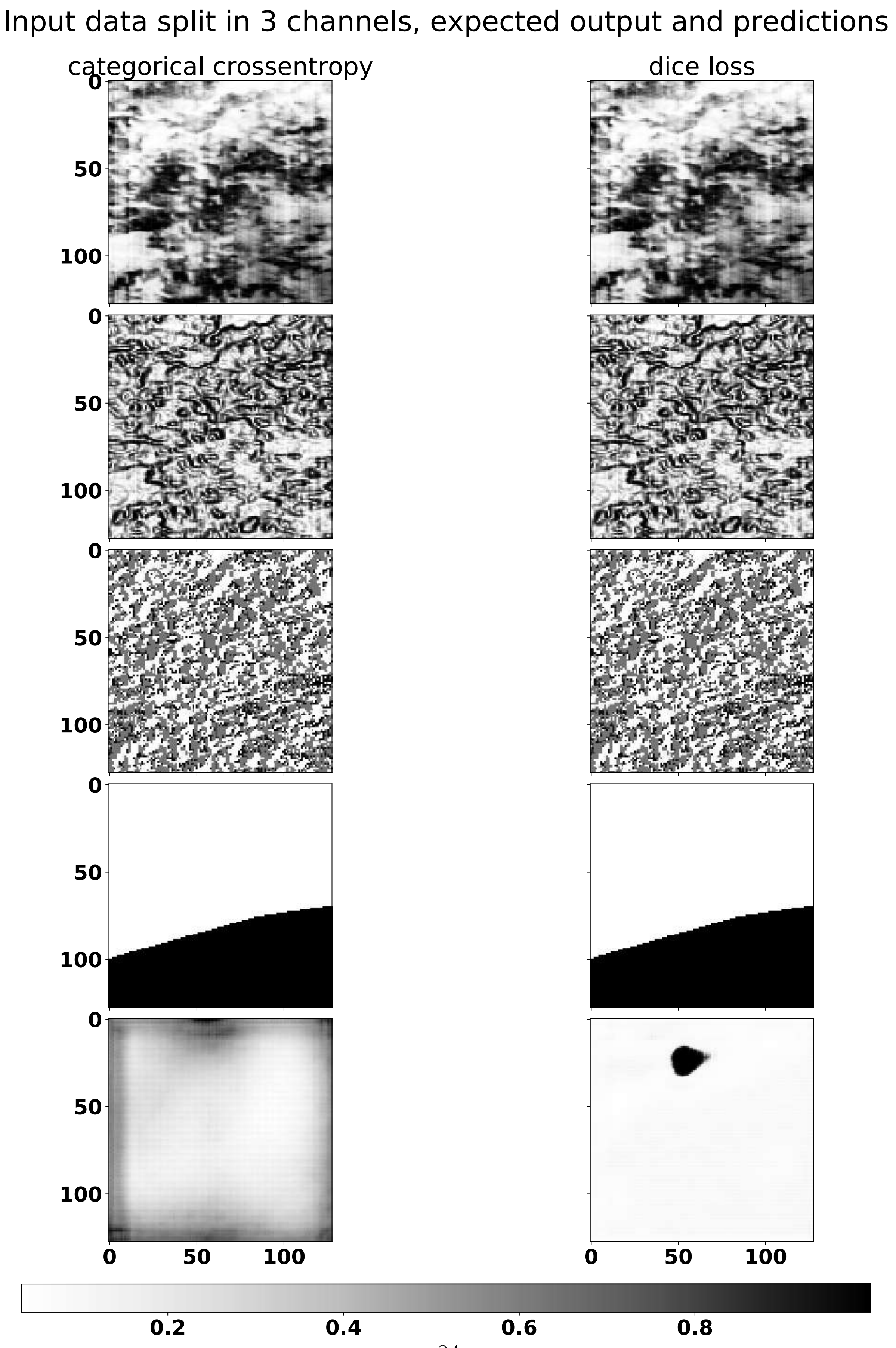

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

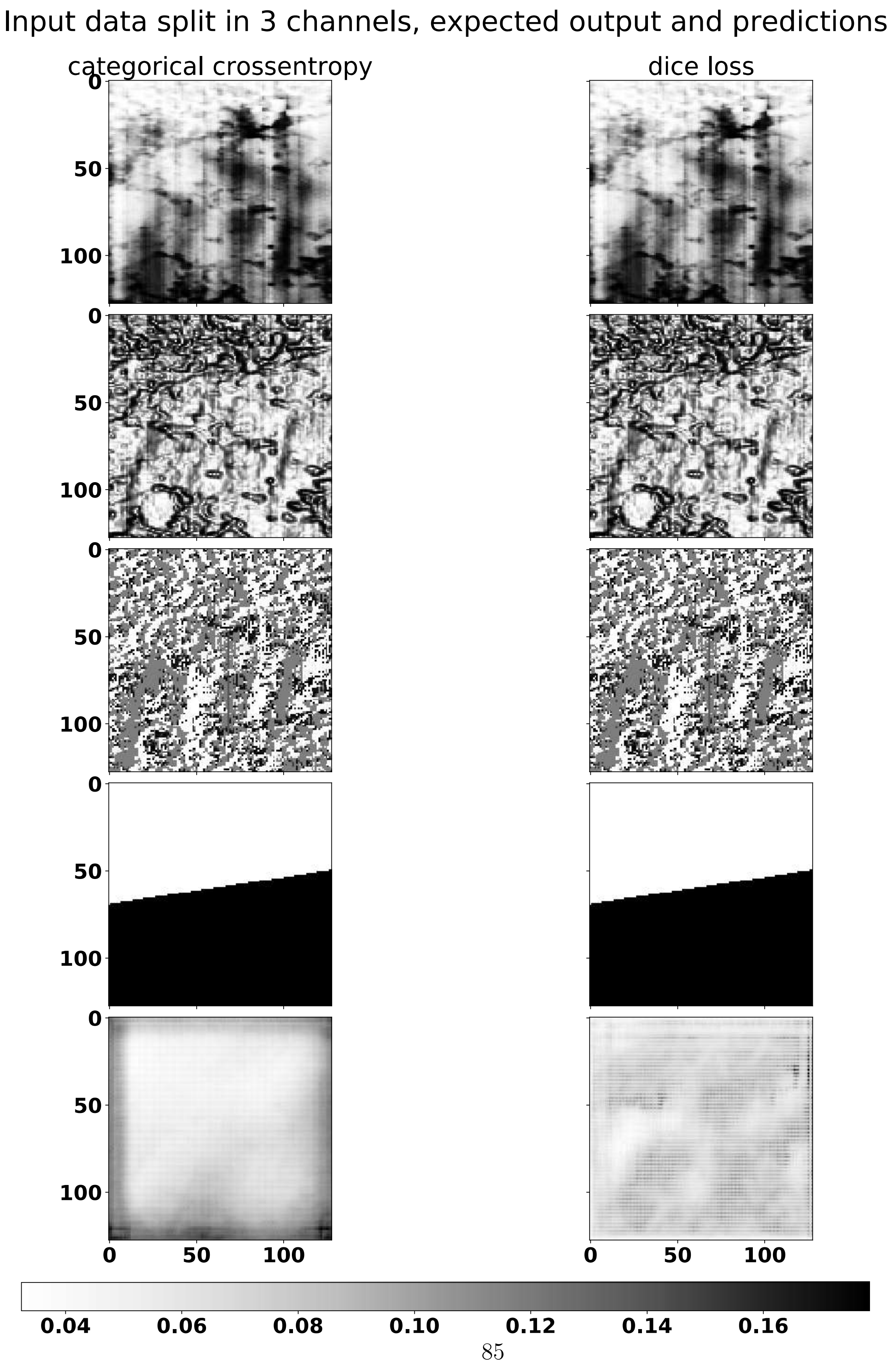

Input data split in 3 channels, expected output and predictions
Raw data
+ 5%
+ 10%
+ 20%
0
50
100
0.0
0.2
0.4
0.6
0.8
1.0

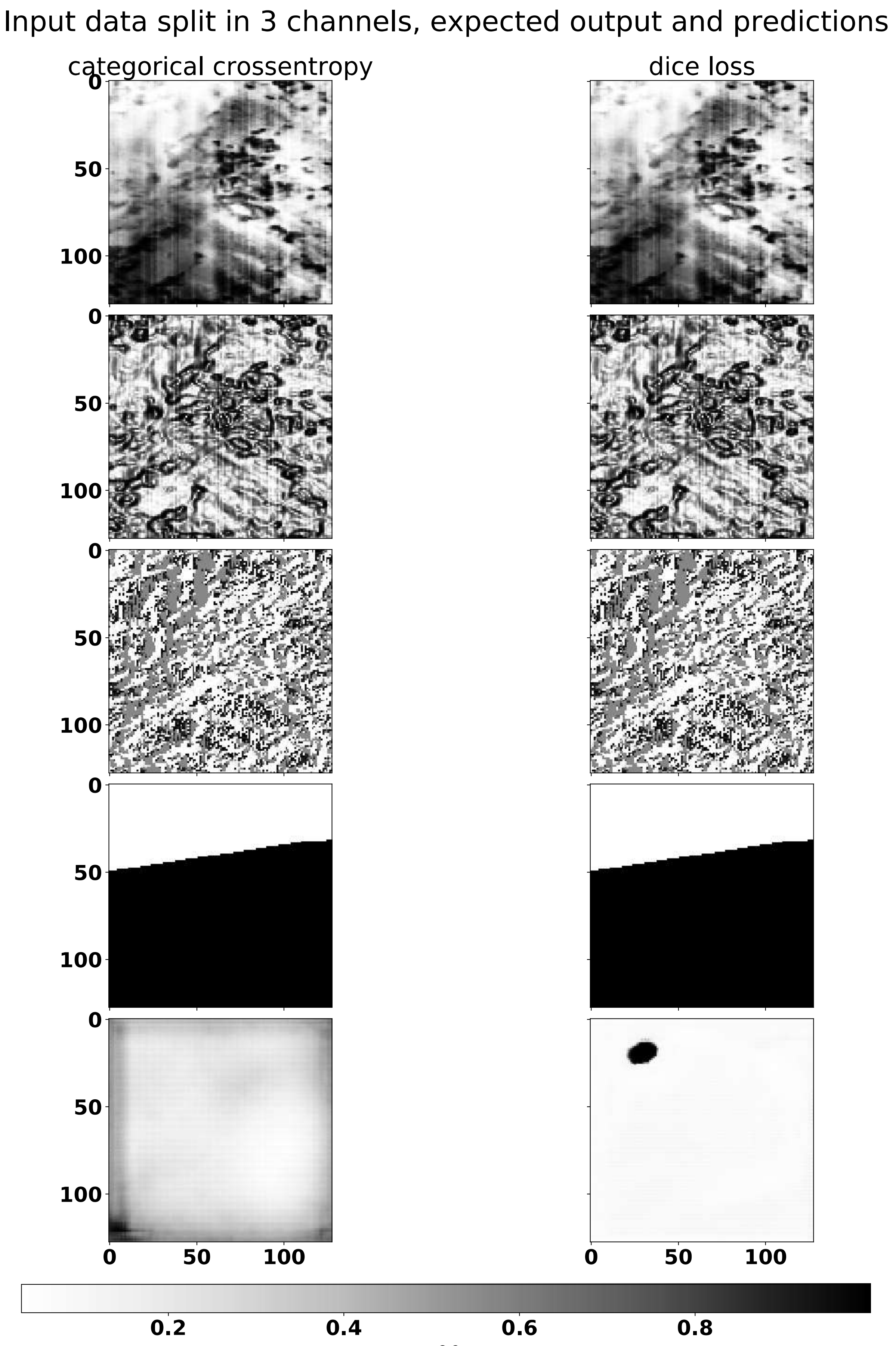
A.3 Multiple loss functions for CNN SRTM


Input data split in 3 channels, expected output and predictions
categorical crossentropy
dice loss
0
50
100
0.04
0.06
0.08
0.10
0.12
0.14
0.16


Input data split in 3 channels, expected output and predictions
categorical crossentropy
dice loss
0
50
100
0
50
100
0.2
0.4
0.6
0.8

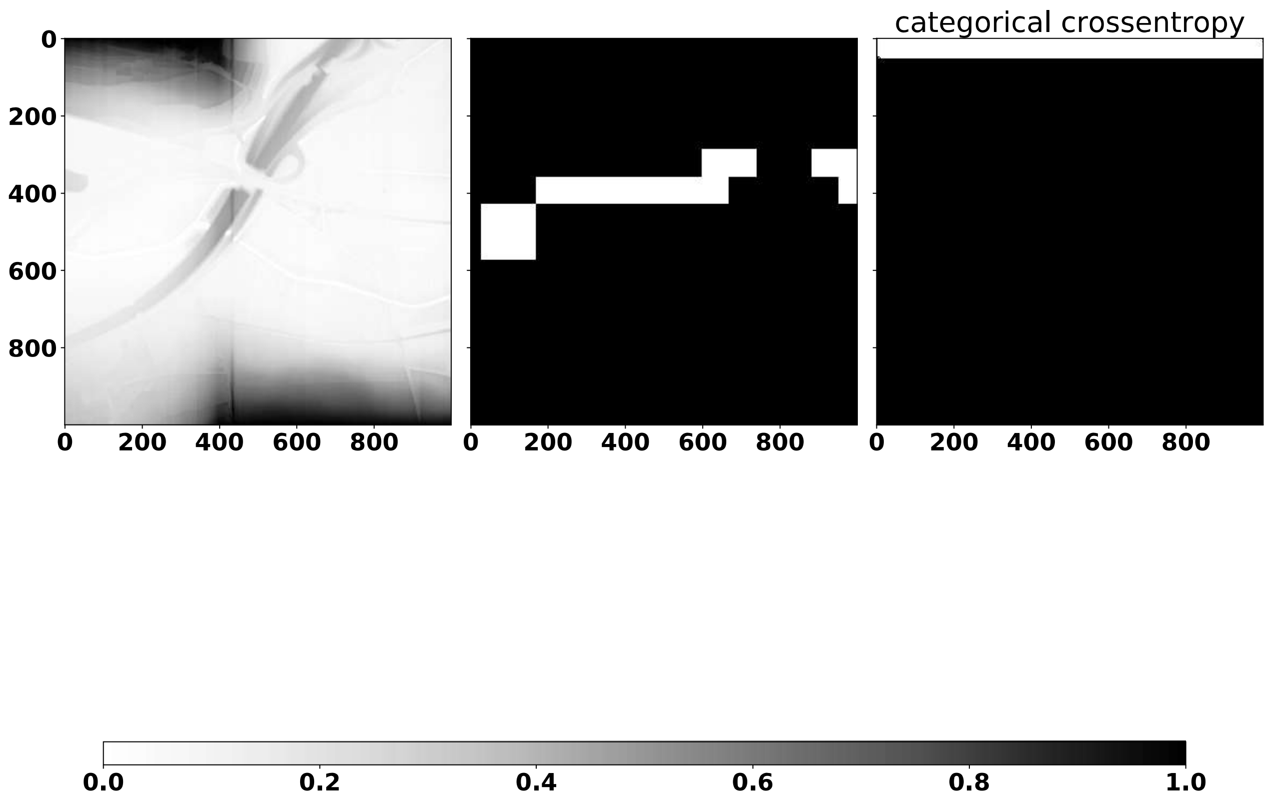

Input data split in 3 channels, expected output and predictions
categorical crossentropy
dice loss
0
50
100
0.2
0.4
0.6
0.8


Input data split in 3 channels, expected output and predictions
categorical crossentropy
dice loss
0
50
100
0
50
100
0.04
0.06
0.08
0.10
0.12
0.14
0.16

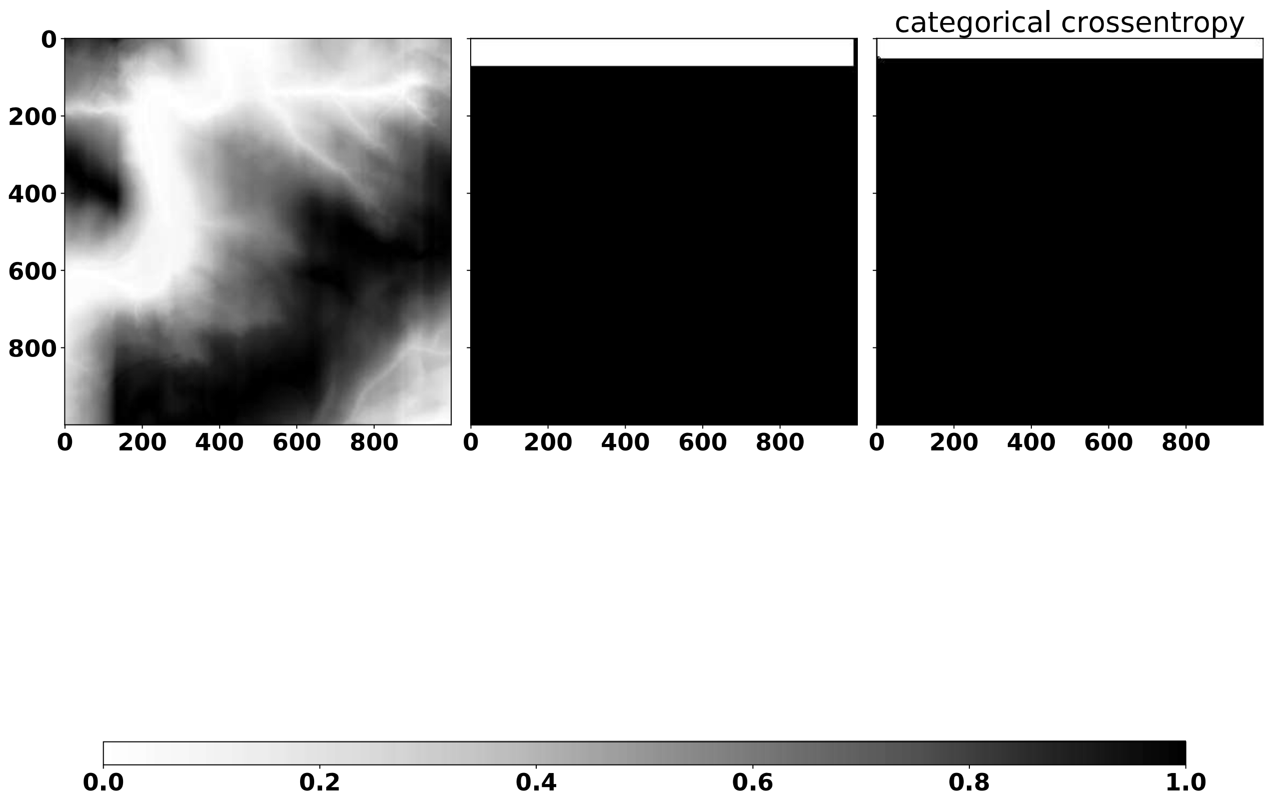

Input data split in 3 channels, expected output and predictions
categorical crossentropy
dice loss
0
50
100
0.2
0.4
0.6
0.8

A.4 Testing results for the Lidar CNN

Input lidar data, expected output and predictions



Input lidar data, expected output and predictions



Erklärung der Urheberschaft

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Freiburg, 1.Juli 2020 Unterschrift