AUTOMATICALLY GENERATED TRAINING DATA FOR LAND COVER CLASSIFICATION WITH CNNS USING SENTINEL-2 IMAGES

M. Voelsen[1,*], J. Bostelmann[2], A. Maas[2], F. Rottensteiner[1], C. Heipke[1]

[1] Institute of Photogrammetry and GeoInformation, Leibniz Universität Hannover, Germany
(voelsen, rottensteiner, heipke)@ipi.uni-hannover.de
[2] German Land Survey Office of Lower Saxony (LGLN), Hannover, Germany
(jonas.bostelmann, alina.maas)@lgln.niedersachsen.de

Commission III, WG III/7

KEY WORDS: Remote Sensing, Sentinel-2, Land Cover, CNN, Deep Learning, Semantic Segmentation

ABSTRACT:

Pixel-wise classification of remote sensing imagery is highly interesting for tasks like land cover classification or change detection. The acquisition of large training data sets for these tasks is challenging, but necessary to obtain good results with deep learning algorithms such as convolutional neural networks (CNN). In this paper we present a method for the automatic generation of a large amount of training data by combining satellite imagery with reference data from an available geospatial database. Due to this combination of different data sources the resulting training data contain a certain amount of incorrect labels. We evaluate the influence of this so called label noise regarding the time difference between acquisition of the two data sources, the amount of training data and the class structure. We combine Sentinel-2 images with reference data from a geospatial database provided by the German Land Survey Office of Lower Saxony (LGLN). With different training sets we train a fully convolutional neural network (FCN) and classify four land cover classes (`Building`, `Agriculture`, `Forest`, `Water`). Our results show that the errors in the training samples do not have a large influence on the resulting classifiers. This is probably due to the fact that the noise is randomly distributed and thus, neighbours of incorrect samples are predominantly correct. As expected, a larger amount of training data improves the results, especially for the less well represented classes. Other influences are different illuminations conditions and seasonal effects during data acquisition. To better adapt the classifier to these different conditions they should also be included in the training data.

1. INTRODUCTION

Automatic classification and analysis of remote sensing imagery is highly interesting for current and future applications in land surveying and related disciplines such as navigation and city planning. Deep learning techniques, like convolutional neural networks (CNN) for image data, provide powerful state-of-the-art methods to solve these tasks, including semantic segmentation, change detection and object delineation. The performance of a contemporary classifier highly depends on the amount and quality of available training data. These training data must consist of remote sensing imagery with the corresponding labels.

Since the launch of the first Sentinel mission in 2014 as part of the European Union's Copernicus project, image data has been continuously collected and made available free of charge (Fletcher, 2012). This data is already used for tasks like land cover classification and, due to its high temporal resolution, also for change detection and rapid mapping in case of natural disasters.

While more and more satellite images are freely available, large amounts of labeled remote sensing imagery, as needed for the training of a classifier, are not. This is a challenge because the manual labelling is time consuming and costly and existing real-world remote sensing training data sets like, e.g. (Daudt et al., 2018) created for their work, are limited in size if compared to general image data sets like ImageNet (Deng et al., 2009).

There are different strategies to avoid manual labelling: Using domain adaptation (DA) techniques a classifier that was trained on data obtained under different conditions is adjusted to the new data (Wang, Deng, 2018). Other approaches, like (Kemker, Kanan, 2017) or (Kolos et al., 2019) generate synthetic training data to pre-train the classifier. If there is no available data from a different domain weakly labeled data can be used as well. (Kaiser et al., 2017) showed that large amounts of noisy data from open street map (OSM) can replace a major part of manually labeled data and can improve the segmentation and generalization performance. (Wegner et al., 2016) use Google Maps images to classify tree species without any manually labeled training data. They rely on the learning algorithm to handle the noise and also achieve good results. Content-based image retrieval (e.g. (Sumbu et al., 2019)) also has the potential to alleviate the problem of manual labeling by providing automatically generated labels.

In this work we address this problem by combining Sentinel-2 imagery with an available topographic geospatial database (referred to as maps in this paper) to generate a large amount of training data, which results in two challenges that have to be considered: the images and the land cover data have to be compatible with respect to acquisition time, spatial resolution and class structure, and we have to consider the fact that the training data may contain a certain amount of incorrect labels. However, due to the large amount of training data we assume that the number of incorrect labels is relatively small and that the resulting effects can be considered as random noise.

[*] Corresponding author

To create training data with a small amount of label noise we consider different sources of label noise and try to avoid these in the selection of the training data: First, we combine map information with satellite images acquired at similar times. Thus, we minimize label noise caused by changes of land cover over time. Second, we use Sentinel-2 data with a low cloud coverage. Thus, label noise caused by clouds and cloud shadows interfering with land cover objects is reduced. Third, we combine samples of different classes to define a class structure containing classes, which are detectable in the given resolution of the satellite images.

With these training data we train a fully convolutional network (FCN) for a following classification using the U-net structure of (Ronneberger et al., 2015). In our experiments we combine the optical data from Sentinel-2 with data from the German Land Survey Office of Lower Saxony (LGLN). This database covers the whole area of Lower Saxony (47 600 km^2) and includes 24 different land use types. These 24 land use classes are aggregated to four classes to obtain a suitable class structure for land cover. The land cover data is rasterized to the resolution of the satellite images to obtain sensor data and labels in the same grid.

This paper is structured as follows: Chapter 2 gives an overview of pixel-wise classification with deep learning techniques and the generation of large amounts of training data for remote sensing images. In chapter 3 we introduce our network architecture and the training procedure. Afterwards we present the preprocessing of the data and our experimental set-up in chapter 4. In chapter 5 we present and evaluate the results before summarising and giving an outlook on future work in Chapter 6.

2. RELATED WORK

We start this overview with an introduction of methods for pixel-wise classification using deep learning techniques. After that we discuss different possibilities to create larger amounts of training data for remote sensing images including a short overview of domain adaptation and label tolerant classification.

For the classification of satellite imagery (called semantic segmentation in computer vision) every pixel of the image is assigned to a class. (Zhu et al., 2017) give an overview of deep learning techniques for this task using remote sensing data. A first strategy is the usage of a fully convolutional neural network (FCN) first introduced by (Long et al., 2015). In a FCN the spatial resolution of the input image is downsampled using convolution and pooling operations and upsampled again to the original size to obtain class probabilities for each pixel. FCN were further developed into symmetrically structured encoder-decoder networks, that use convolutions and pooling in the encoder part and transposed convolutions in the decoder part to upsample the images again (Noh et al., 2015). Due to the pooling operations in the encoder parts the receptive field is enlarged and more context information is involved, but the downsampling steps also lead to inaccurate object boundaries. One way to overcome this problem is to employ a conditional random field (CRF) at the last layer of the network, see (Chen et al., 2018a). Another possibility is the usage of skip connections, which were first introduced by (Long et al., 2015). Skip connections combine feature maps from different resolutions to obtain the final classification output with more precise object boundaries. Skip connections are now very popular and different variants exist to integrate them into the network. For example, (Badrinarayanan et al., 2017) save the indices of the max-pooling and integrate them in the decoder part of their network. (Chen et al., 2018b) concatenate low-level features from the encoder part at one point during the upsample process. They also use spatial pyramid pooling to be able to encode context information at multiple scales. Also, in the U-Net structure, introduced by (Ronneberger et al., 2015), the corresponding feature maps from the encoder and decoder part are concatenated for every spatial resolution, which leads to sharper object boundaries in the result.

The success of neural networks was possible due to the large amounts of data used for training (Krizhevsky et al., 2012). However, it is very time consuming and costly to manually create sufficient training data for a specific task and in remote sensing existing real-world data sets like (Daudt et al., 2018) are too limited in size. On the other hand a number of strategies exist to avoid manual labelling.

In transfer learning (TL), a classifier trained in a source domain is subsequently transferred to a target domain for which no or very few training data are available. For TL to work it is important that the data and the problem do not differ to much between source and target domain. A particular form of TL is domain adaptation (DA): A classifier is learned with the training data of the source domain to predict the labels of the target domain (Tuia et al., 2016). Again it is important that the joint distributions of the two domains do not differ to much and, for example, have the same class structure. (Wang, Deng, 2018) give a recent overview about DA based on CNNs, see (Vogt et al., 2018) for an example for DA with aerial images.

For multispectral or hyperspectral images DA is often not possible due to the lack of large-scale labeled data sets that contain more spectral bands than RGB. This challenge can be solved by simulation: (Kemker, Kanan, 2017) use the Digital Imaging and Remote Sensing Image Generation (DIRSIG) modeling software and create a large amount of synthetic multispectral training data. Subsequently, they fine-tune their model with a limited amount of real-world data and show a significant improvement of the results compared to a training only on real-world data. (Kolos et al., 2019) present another method to generate realistic synthetic remote sensing data: They use the Esri-City-Engine with cartographic data from OSM for geometry and the game engine Unity for data rendering. Their results for change detection, using a Siamese U-Net structure, also improve the performance and robustness of models for remote sensing applications with limited training data.

For our experiments a large amount of automatically generated training data is available that includes some amount of label noise due to the different data sources that are combined (see Section 4.1 for details). For that reason, in the following we give an overview about training under label noise and on how noisy data can improve classification results.

Training under label noise is important in many fields such as epidemiology, econometrics and computer aided diagnoses. Frénay and Verleysen (2014) distinguish three strategies for dealing with label noise: classifiers that are robust by design, data cleansing and classifiers that are robust after adaptation to the presence of noise.

The first strategy uses classifiers that by design are robust against some degree of label noise or a particular type of label noise. One example for such a classifier are deep learning methods. (Drory et al., 2018) show, that the ability of the network to deal with label noise depends on the type of distribution of the noisy labels. If the neighbourhood of a noisy sample contains mostly correct samples in feature space, e.g. if the noise is randomly spread across the training set, the influence of label noise is relatively low. On the other hand, locally concentrated errors can lead to a deterioration of the result. An example where this strategy is used is (Wegner et al., 2016). The authors use Google Maps images of tree species in a fully automated process relying on the learning algorithm to handle the noise. The second strategy tries to identify and eliminate incorrect training samples before the actual training procedure. In remote sensing, data cleansing seems to be the most commonly used strategy for coping with label noise, although data cleansing approaches tend to eliminate too many instances (Frénay, Verleysen, 2014). The third strategy is to use a classifier that is robust to label noise by adapting it to the presence of label noise. In probabilistic approaches, e.g. (Bootkrajang, Kabán, 2012), a noise model is used to consider label noise in the training process. Non-probabilistic methods typically do not estimate the parameters of a noise model, e.g. the label noise tolerant version of a Support Vector Machine (SVM) (An, Liang, 2013). (Maas et al., 2019), when classifying remote sensing data using outdated maps, deal with label noise by adapting a Random Forest classifier to include a complex noise model. The underlying assuption is that pixels with false labels form clusters in the image, as they are caused by land cover changes over time.

The key question is how much the label noise influences the classifier and how much any problems can be mitigated by one of these strategies. Of course the answer highly depends on the data set, the type of label noise and the classification method. In the following we present a few approaches that are close to our field, dealing with noisy remote sensing data and CNNs for semantic segmentation. In (Kaiser et al., 2017) the authors investigate the question whether a large amount of noisy data can replace a major part of the manual labelling process. They apply different experiments to compare models trained on a large amount of weakly labeled data from OSM, a smaller amount of manually labeled data and a combination of both. The result is that a large amount of noisy training data can in fact improve the segmentation and generalization performance of the model. Also, training only on noisy data leads to acceptable results, although these are far from optimal. In the work of (Maggiori et al., 2017) the authors come to a similar conclusion: They achieve good results using a two-step process: First they train their model on imperfect training data from OSM and afterwards fine-tune it with a small amount of manually labeled images.

For our experiments we have access to a large set of automatically generated training data in the form of a geospatial database that includes some amount of label noise (see Section 4.1 for details). Using this data we follow the first strategy for dealing with label noise: We rely on our classifier to ignore the noise instead of learning false class representations. We expect the noise to be distributed randomly, so that samples located next to an incorrect training sample should contain correct labels. For this reason the the label noise should not influence the classifier too much like (Drory et al., 2018) showed. For classification we use the U-Net structure presented by (Ronneberger et al., 2015) as a simple but effective network, especially to accurately localize object borders which is required in most land cover applications (Yang et al., 2019).

3. METHODOLOGY

3.1 Workflow

In figure 1 the workflow of the training and the evaluation of a classifier without using manually labeled data is depicted.

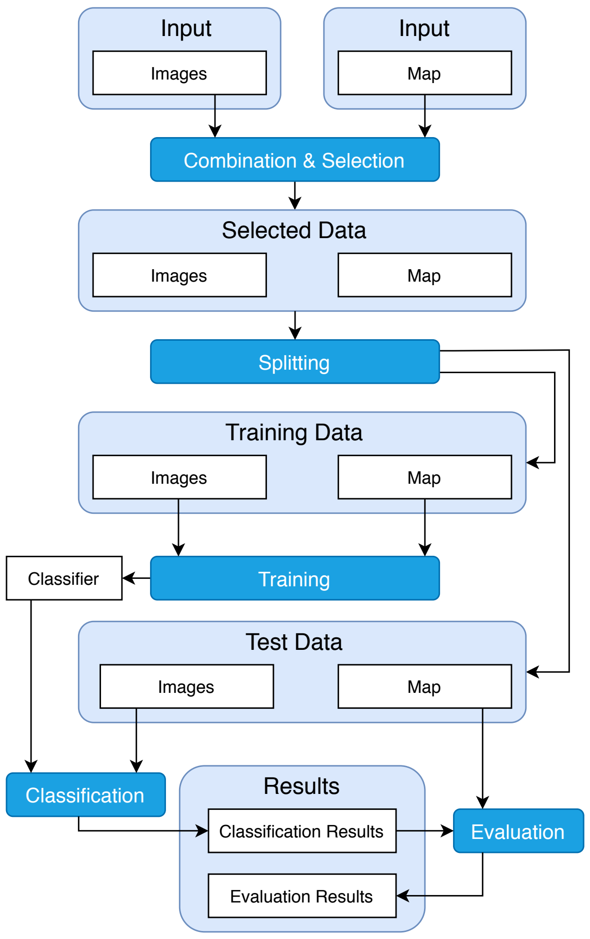


Figure 1. Workflow of training and evaluating a classifier using two large pools of remote sensing imagery and map data

The input are remote sensing images and topographic information of a geospatial database (abbreviated as map in the figure). We assume that for a given area of interest a large pool of images with a high temporal resolution and different versions of database content with distinctly fewer time stamps are available. In a first step we combine the two data sources. In the process, first the images with a minimum time difference to the latest update of the mapare selected.

Afterwards the data is split into training and test data. The training data is used to learn the classifier, which is subsequently employed to classify the images in the test phase. For evaluation we compare the classification result with the database information of the test data. For both processes the map data serves as reference information.

3.2 Network Architecture

For the pixel-wise classification we use the U-Net structure that was introduced by (Ronneberger et al., 2015). This encoder-decoder FCN uses skip connections to better preserve the geometric location of object boundaries, which is important for land cover applications (Yang et al., 2019). The input patches have a size of 256 x 256 pixels and contain channels for the ten spectral bands of the Sentinel-2 images (for more details see Section 4.1). In the encoder part of the network there are four layers, each consisting of two 3 x 3 convolutions with zero padding followed by a Rectified Linear Unit (ReLU) as activation function (Nair, Hinton, 2010). This is followed by a 2 x 2 max-pooling with stride 2 for down-sampling. We start with 16 feature channels and double this number in each of the four layers. Also in the coarsest resolutioon, two 3 x 3 convolutions are implemented. In the decoder part we have another four layers to upssample the images again. This is done with 2 x 2 transposed convolutions. These feature maps are concatenated with the corresponding feature maps from the encoder part (skip connections). Then, two 3 x 3 convolutions and a ReLU are applied per layer. In the last layer a 1 x 1 convolution maps the feature vectors to the number of classes, and we obtain scores for each desired class via the softmax activation function. The network is trained from scratch for all experiments. The parameters are initialized randomly by drawing them from a zero-mean normal distribution. We also apply dropout for regularization in each layer with a dropout rate of 0.1 (Srivastava et al., 2014).

3.3 Training

During training the loss function which measures the quality of the network predictions is minimized by iteratively comparing the predicted class labels and the ground truth for every training sample. The belief y_n^k of a network with the current parameters w that a sample x_n belongs to the class k results from the softmax activations:

$$y_n^k(x_n, w) = \frac{e^{\bar{y}_n^k}}{\sum_k^K e^{\bar{y}_n^k}} \tag{1}$$

with $\bar{y}_n^k$ the output of the last layer of the network and K the total number of classes. The classification error E of the network can now be calculated with the softmax cross entropy (Bishop, 2006):

$$E(w) = \sum_n E_n(w, x_n) = -\sum_n \sum_k C_n^k \cdot ln(y_n^k) \cdot cw_k \tag{2}$$

In Equation 2 the term C_n^k is equal to 1, if the n_{th} sample belongs to class k and equals 0 otherwise. This equation can be used for any desired number of classes. In order to reduce negative effects stemming from unbalanced data, we next introduce weights cw_k for each class c that depend on the number of occurrences n_c in the data set, calculated as follows (Patel, 2020):

$$cw_k = ln(\frac{N}{n_c}) \tag{3}$$

with ln the natural logarithm and N the total number of pixels in all training patches. In the end all class weights are divided by the maximum of all calculated class weights. This weighting partly compensates effects from the unbalanced class distribution of the training data. By minimizing equation 2 the parameters w can be determined using stochastic gradient descent (Bishop, 2006). Details on the implementation are described in section 4.3.

3.4 Evaluation

To evaluate our results during and after the training process we calculate different accuracy measures when comparing the predicted results to the reference. The overall accuracy is the percentage of correctly classified samples (in our case the number of pixels) over all images. Since the class distribution of our data is unbalanced the F1-score is a better quality measure to compare the results for the different classes:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4}$$

with the precision defined as the percentage of samples predicted as a certain class that actually belong to that class in the reference and recall defined as the percentage of samples belonging to a class in the reference that were predicted as that class. The F1-score is the harmonic mean of precision and recall.

4. EXPERIMENTS

In this chapter the experiments are presented. We investigate the influence of the following three aspects of the training data: The time difference between the generation (or the last update) of the map information and the acquisition of the satellite imagery, the size of the training area and the chosen class structure. Using a small example, we finally assess the effects of label noise.

In the following the used data is introduced first. It consists of Sentinel-2 imagery of Northern Germany including the city of Hannover and a land cover database of the German landscape model of Lower Saxony. We describe the data characteristics and how the data sources are combined to generate the different training data sets. Afterwards the training configuration for the FCN is described.

4.1 Data

To create the reference data for the area of Lower Saxony, the official German landscape model ATKIS (Authoritative Topographic-Cartographic Information System) is used. This dataset is created, updated and revised by human operators mainly relying on digital orthophotos (DOPs). The underlying aerial images are updated every three years. The manual and time-consuming updating process can lead to information older than five years. For generating the training labels, the land use information is taken into account. The database consists of 24 different object types and covers the complete area of Lower Saxony. To serve as class labels for the training of a CNN the vector data are rasterized to the resolution of the satellite imagery. Sometimes the map from the LGLN is updated without considering imagery, for instance when there are major development projects about which geometrical information is produced in other ways, e.g. by geodetic survey. This can also lead to differences between the map and satellite data.

Classification is carried out based on images from the two Sentinel-2 satellites provided by the European Space Agency (ESA). The Level-2A products contain bottom-of-atmosphere reflectance and cloud masks from the top-of-atmosphere reflectance for every pixel (Fletcher, 2012). From the 13 spectral bands the four RGBI bands with GSD = 10 m and the bands with 20 m spatial resolution (six altogether) are used. The latter are upsampled to 10 m by bilinear interpolation. The images come in tiles of 100 x 100 km^2. In the first step, they are subdivided into partly overlapping 8 x 8 km^2 tiles. By setting a threshold for the mean value for the cloud cover of each tile, only tiles with no or very few clouds are selected and are used further. Another benefit of using tiles instead of the whole images is the fact that processing for large data sets can be scaled up by distributed computing.

4.2 Set up

To generate the training data, the satellite images are combined with the map information. For our experiments we use two Sentinel-2 images, and we use areas of different size for the different experiments. As mentioned, the time difference between acquisition of the two data sets should be as small as possible to minimize the amount of incorrect training labels. The part of the database we use in the experiments was last updated on September 30, 2019. The Sentinel-2 images closest in time that could be used were taken on September 22, 2019. At that date the whole training area was covered with images exhibiting a mean cloud coverage of 1.1 %, which is a very good value. Therefore, we did not add tiles from other dates. The database provides 24 different land use classes. To obtain useful classes for our experiments, we combined them to the following four classes: `Building`, `Agriculture`, `Forest` and `Water`.

In the following the preparation of different training sets for the different experiments is described. The first experiment investigates how the time difference between the acquisition of the images and the map information influences the training process. The map data for a training area of size 24 x 24 km^2 is combined with satellite images of different dates. We use two images: (a) The image acquired on September 22, 2019, i.e. eight days before the map was last updated, and (b) an image from June 29, 2018, i.e. taken 14 months before.

In the second experiment the influence of the size of the training area is evaluated. We use two different areas: a small one with a size of 24 x 24 km^2, shown in Figure 2, and a larger area of 104 x 104 km^2. The class distribution of the two training areas is shown in Table 1, the differences between the small and the large area are negligible. `Agriculture` covers about 56 % of the area, followed by `Forest` with more than 30 %. The class `Water` has only 1 % and is the least represented class. This unbalanced data distribution is considered by using class weights in the training process (see above).

In the last experiment the impact of the number of classes is evaluated. In order to do so the four chosen classes are compared to a binary classification for each of them, i.e. all other classes are combined into one background class and the focus is on the foreground class.

[%]	Small area	Large area
Building	11,3	11,9
Agriculture	55,7	56,4
Forest	32,1	30,6
Water	0,84	1,1

Table 1. Class distribution in the small and large training area

4.3 Training configuration

As mentioned, the network takes image patches of size 256 x 256 pixels (or 2.56 x 2.56,km^2) as input for training. We subdivide the training tiles (of 8 x8 km^2, see above) accordingly and accept some overlap. During the training process we minimize the loss function using stochastic gradient descent, as described in Section 3.3. We use the ADAM optimizer (Kingma, Ba, 2015) with the parameters $b_1 = 0.9$ and $b_2 = 0.999$. We apply hyperparameter tuning including different batch sizes, learning rates and stopping criteria. Due to the small amount of only 144 image patches for the small training area a split of 60 % training data and 20 % validation and test data each, is chosen. This is due to the fact that the results, especially for the less well represented classes depend strongly on the split of the data. We split the data for the small and the large training area once and use the same training, validation and test split for all experiments. During the training process we monitor the accuracy measures for the validation data. On this base the best parameters for the model are chosen.

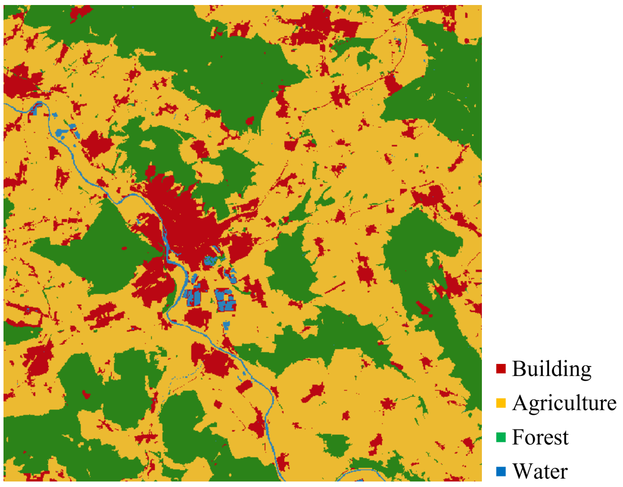


Figure 2. Reference for the small training area

For the given data we achieved the best results with a batch size of 4 and a learning rate of 0.001. In all experiments the training process was stopped if the results did not change for 150 epochs on the validation data. As stopping criteria the loss worked best for the multiclass classification and the F1-score achieved the best results for binary classification.

Finally, after the training process we compare the metrics on the test data to see how our models perform on data not seen during the training process.

5. RESULTS AND DISCUSSION

5.1 Impact of the time difference

In Figure 3 the two different satellite images for the training area are shown. The differences between the images are mainly due to phenological differences of the agricultural areas and slight illumination changes. In a visual inspection only minor changes of land cover, like the enlargement of a lake or property were found. For both satellite images the classifier is trained on the small area. Afterwards it is tested on the same and also on a part the another, larger area to be able to compare the results independently. The results of both tests are shown in Table 2.

If testing is applied in the same area as the training process, both classifiers achieve good results with 93.1 % and 94.3 % overall accuracy. Especially the classes `Agriculture` and `Forest`,

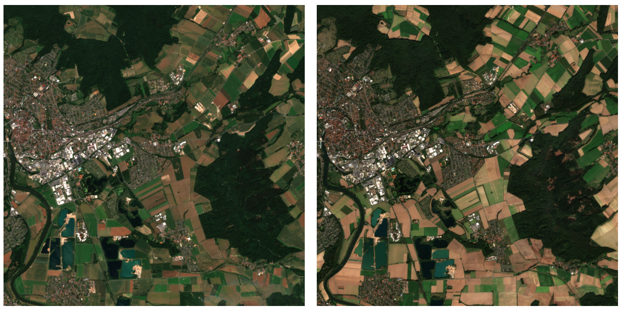

(a) 29 June 2018 (b) 22 September 2019

Figure 3. Satellite images for the small training area from June 2018 and September 2019 that are used for training

Test area	Small		Large	
Training data	2018	2019	2018	2019
F1-Score [%]:				
Building	80.3	81.9	51.6	64.6
Agriculture	94.8	95.3	77.1	84.6
Forest	95.1	96.9	84.7	85.8
Water	75.3	69.0	65.7	78.5
Mean F1-score	86.4	85.8	69.8	78.4
Overall Acc.	93.1	94.3	74.7	82.1

Table 2. Overall Accuracy and F1-score for the classification with 4 classes trained with satellite data from different dates

which occur most often (see again Table 1), achieve high F1-scores between 94.8 % and 96.9 %, independent on the Sentinel image used for training. For the less respresented classes `Building` and `Water` the results are considerably lower. Surprinsingly the result for the class `Water` becomes better with the classifier trained with Sentinel data from 2018, this may be due to the split of the small amount of data into training and test set.

If testing is done on the large area that had not been seen by the classifiers before, the overall accuracy and F1-score drop by a large amount. The main reason for this drop is probably the size of the small training area that leads to overfitting to the training data. As a result the classifier can not generalize well to the slightly different data used for testing. The classifier trained with data from 2019 achieves an overall accuracy of 82.1 % that is 7.4 % higher than the performance of the classifier trained with outdated satellite data. The F1-scores for the 2019 image are also better, especially for the less represented classes with a difference of more than 12 %.

This result can also be seen in a visual comparison of the classification results: In Figure 4 the classification results for both classifiers are shown in comparison to the reference. The classifier trained with satellite data from 2018 has major difficulties to correctly delineate the classes, especially the class `Building` that covers big areas of the class `Agriculture` in the classification result.

Overall, the results show that the classifier is sensitive to changes in illumination and phenological states of the agricultural land that was shown in Figure 3. Major land cover changes are not present in this experiment and thus can not significantly influence the results. To make the classifier independent of the illumination and ground conditions it is important to generate training data from different satellite images. Here images from different seasons and thus different illumination conditions should be included.

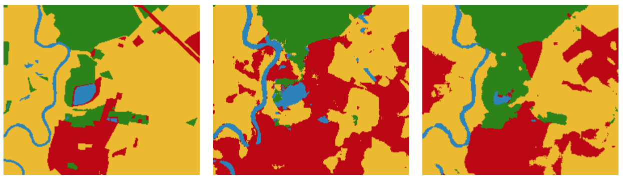

(a) Reference (b) Classifier 2018 (c) Classifier 2019

Figure 4. Classification results for the classifier trained with satellite data from 2018 and 2019, part of the large area

5.2 Impact of the size of the training area

The results for the classifiers, trained on the small (24 x 24 km^2) and the large (104 x 104 km^2) area are shown in Table 3.

Training	Small	Large	Small	Large
Testing	Small	Large	Large	Small
F1-Score [%]:				
Building	81.9	86.7	64.6	87.0
Agriculture	95.3	95.5	84.6	96.4
Forest	96.9	94.5	85.8	97.1
Water	69.0	86.9	78.5	83.1
Mean F1-score	85.8	90.9	78.4	90.9
Overall Acc.	94.3	94.0	82.1	95.4

Table 3. Overall Accuracy and F1-score for the classification with 4 classes for different combinations of training and testing

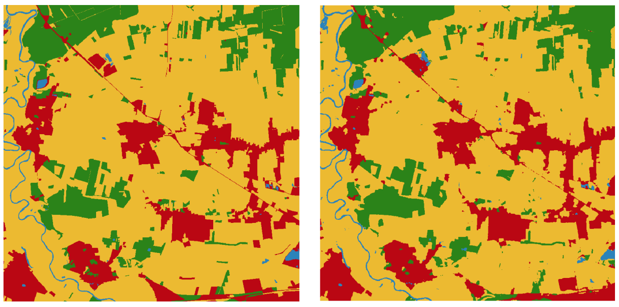
As in the first experiment both classifiers achieve very good results with 94.0 % and 94.3 % overall accuracy if testing is applied in the same area as the training process. Again the classes that occur most often achieve high F1-scores with values between 94.5 % and 96.9 %, independent on the size of the training area. For the less represented classes `Building` and `Water` the results are considerably lower. On the other hand the F1-scores for these classes increase by 4.8 % for `Building` and 17.9 % for `Water` when the larger training area is used.

The impression that especially the classes with less pixels benefit from a larger training area is also confirmed when looking at the last two columns of the table. When training is run on the small area and the classifier is tested on the larger one, the results for all classes decrease, resulting in a mean F1-score of 78.4 % and an overall accuracy of 82.1 %. If the classifier that was trained on the large area is tested on the small one the results become better by a small amount of 1.4 % for the overall accuracy compared to testing on the large area. The only class whose F1-score drops by 3.8 % is the class `Water`. But the result of 83.1 % it is still at a good level.

Overall, our results show that a larger training area significantly improves the results. For the small area the classifier tends to overfit to the available training data. Especially for less represented classes it is important that there are enough data samples during the training process of the classifier.

5.3 Impact of class structure

The results for the classification with four classes compared to binary classification for each of them are shown in Table 4. For the less represented classes `Building` and `Water` the results improve by a small amount of 1.5 % and 1.1 %, respectively. For the classes `Agriculture` and `Forest` the results for binary classification decrease by a very small amount of 0.2 % and 0.3 %, respectively.



(a) Reference (b) Classification Result

Figure 5. Classification result for training and classification on the large training area

	4 classes [%]	Binary [%]	Difference [%]
Building	86,7	88,2	+1,5
Agriculture	95,5	95,3	-0,2
Forest	94,5	94,2	-0,3
Water	86,9	88,0	+1,1

Table 4. F1-score for classification with four classes compared to binary classification for each of the classes.

Overall the classification with less classes improves the accuracies for the less represented classes by a small amount.

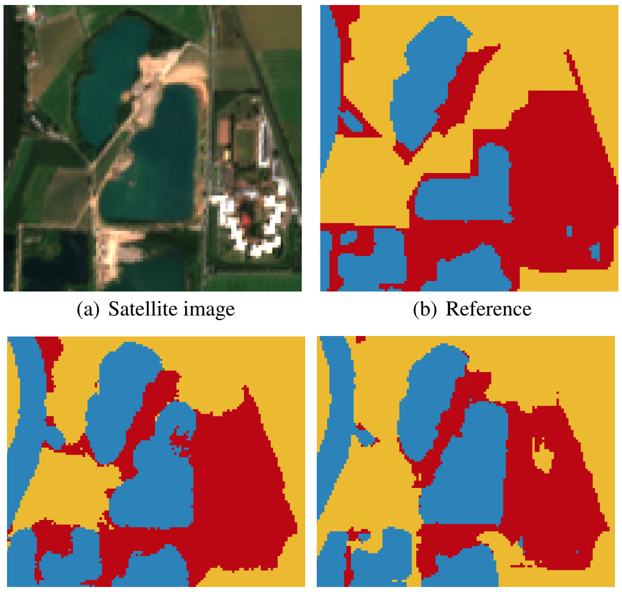
5.4 Incorrect training samples

When combining different data sources to generate a larger amount of training data, inevitably some label noise is introduced. An example is shown in Figure 6: The lake in the images was enlarged which can be seen in the satellite image (Figure 6(a)) but this change is not included in the map yet (Figure 6(b)). In Figures 6(c) and 6(d) the classification results of the classifier trained on the small area and on the large one is shown. Both correctly classify almost the whole lake as `Water`. As a result, in this case the label noise does not significantly influence the classification results, similarly to (Drory et al., 2018). Of course the influence of the label noise can be further reduced by other techniques like the identification of incorrect data.

6. CONCLUSION

In this paper an approach to combine satellite imagery with data from a geospatial database to create a large training data set for pixel-wise classification was presented and tested under different conditions.

In our experiments we could demonstrate that with an increasing time difference between image data acquisition and map update the classification results become worse, which is not surprising. As within one year the actual land cover changes in our test area were limited, so was the amount of introduced label noise due to the higher time difference between the acquisition of the satellite image and the update of the basemap. Consequently, we attribute the difference in overall accuracy of more than 7 % to changes in season which include illumination conditions and the phenological state of vegetation. Other experiments showed that, as expected, a larger amount of training data improves the results. Especially for the less represented classes the classifier can generalize much better with more



(a) Satellite image (b) Reference

(c) Result trained on small area (d) Result trained on large area

Figure 6. Results for trained classifiers on both areas for an area with outdated ground truth

training samples. The results for these classes were further improved when binary classification was used and the training process was adjusted accordingly.

We conclude that the introduced approach to create a large training data set has great potential to be used for task of pixel-wise classification. To further improve the results, it is important to include satellite images from different seasons into training. The existing label noise, caused by the combination of two different data sources did not influence the results at a noticeable scale.

For future work we plan to use the whole area of Lower Saxony for training. Map data from several epochs and satellite imagery covering the course of a whole year are expected to further improve the results. Besides, we will increase the number of classes to provide data for more applications. Finally, we will investigate the effects of fine-tuning the classifier with a small hand-labelled training set, and we will explore other strategies to decrease the influence of label noise.

ACKNOWLEDGEMENTS

We thank the Landesamt für Geoinformation und Landesvermessung Niedersachsen (LGLN) for providing the data of the geospatial database and for their support of this project.

REFERENCES

An, W., Liang, M., 2013. Fuzzy support vector machine based on within-class scatter for classification problems with outliers or noises. *Neurocomputing*, 110, 101–110.

Badrinarayanan, V., Kendall, A., Cipolla, R., 2017. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12), 2481–2495.

Bishop, C. M., 2006. *Pattern Recognition and Machine Learning*. 1^{st} edn, Springer, New York (NY), USA.

Bootkrajang, J., Kabán, A., 2012. Label-noise robust logistic regression and its applications. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 143–158.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A. L., 2018a. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4), 834–848.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H., 2018b. Encoder-decoder with atrous separable convolution for semantic image segmentation. *European Conference on Computer Vision (ECCV)*, 801-818.

Daudt, R. C., Le Saux, B., Boulch, A., Gousseau, Y., 2018. Urban change detection for multispectral earth observation using convolutional neural networks. *IEEE International Geoscience and Remote Sensing Symposium*, 2115–2118.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L., 2009. ImageNet: A large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition*, 248–255.

Drory, A., Avidan, S., Giryes, R., 2018. On the resistance of neural nets to label noise. *arXiv preprint arXiv:1803.11410*.

Fletcher, K., 2012. *Sentinel-2: ESA's optical high-resolution mission for GMES operational services*. ESA SP-1322/2, ESA Communications, Noordwijk.

Frénay, B., Verleysen, M., 2014. Classification in the presence of label noise: a survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5), 845–869.

Kaiser, P., Wegner, J. D., Lucchi, A., Jaggi, M., Hofmann, T., Schindler, K., 2017. Learning aerial image segmentation from online maps. *IEEE Transactions on Geoscience and Remote Sensing*, 55(11), 6054–6068.

Kemker, R., Kanan, C., 2017. Deep neural networks for semantic segmentation of multispectral remote sensing imagery. *arXiv:1703.06452*.

Kingma, D. P., Ba, J., 2015. Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations (ICLR 2015)*.

Kolos, M., Marin, A., Artemov, A., Burnaev, E., 2019. Procedural synthesis of remote sensing images for robust change detection with neural networks. In: Lu, H., Tang, H., Wang, Z.(eds), *Advances in neural networks – ISNN 2019* Lecture Notes in Computer Science, 11555, Springer, Cham. 371–387.

Krizhevsky, A., Sutskever, I., Hinton, G. E., 2012. ImageNet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 60(25), 84–90.

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3431 – 3440.

Maas, A., Rottensteiner, F., Heipke, C., 2019. A label noise tolerant random forest for the classification of remote sensing data based on outdated maps for training. *Computer Vision and Image Understanding*, 188, 102782.

Maggiori, E., Tarabalka, Y., Charpiat, G., Alliez, P., 2017. Convolutional neural networks for large-scale remote-sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(2), 645–657.

Nair, V., Hinton, G. E., 2010. Rectified linear units improve restricted boltzmann machines. *Proceedings of the 27th International Conference on Machine Learning*, 807–814.

Noh, H., Hong, S., Han, B., 2015. Learning deconvolution network for semantic segmentation. *IEEE International Conference on Computer Vision (ICCV)*, 1520–1528.

Patel, M., 2020. Notes on implementation of cross entropy loss. https://github.com/meet-minimalist/Notes-on-cross-entropy-loss (accessed 29/04/2020).

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 234–241.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15, 1929–1958.

Sumbu, G., Charfuelan, M., Demir, B., Markl, V., 2019. BigEarthNet: A large-scale benchmark archive for remote sensing image understanding. *IEEE International Geoscience and Remote Sensing Symposium*, 5901–5904.

Tuia, D., Persello, C., Bruzzone, L., 2016. Domain Adaptation for the Classification of Remote Sensing Data: an overview of recent advances. 4(2), 41–57.

Vogt, K., Paul, A., Ostermann, J., Rottensteiner, F., Heipke, C., 2018. Unsupervised Source Selection for Domain Adaptation. *Photogrammetric Engineering & Remote Sensing*, 84(5), 249–261.

Wang, M., Deng, W., 2018. Deep visual domain adaptation: A survey. *Neurocomputing*, 312, 135–153.

Wegner, J. D., Branson, S., Hall, D., Schindler, K., Perona, P., 2016. Cataloging public objects using aerial and street-level images - urban trees. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6014–6023.

Yang, C., Rottensteiner, F., Heipke, C., 2019. Towards better classification of land cover and land use based on conolutional neural networks. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-2/W13, 139–146.

Zhu, X. X., Tuia, D., Mou, L., Xia, G.-S., Zhang, L., Xu, F., Fraundorfer, F., 2017. Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine*, 5(4), 8–36.