CLASSIFICATION OF LAND COVER AND LAND USE BASED ON CONVOLUTIONAL NEURAL NETWORKS

C. Yang *, F. Rottensteiner, C. Heipke

Institute of Photogrammetry and GeoInformation, Leibniz Universität Hannover - Germany
{yang, rottensteiner, heipke}@ipi.uni-hannover.de

Commission III, WG III/7

KEY WORDS: Land use classification, CNN, geospatial land use database, aerial imagery, semantic segmentation

ABSTRACT:

Land cover describes the physical material of the earth's surface, whereas land use describes the socio-economic function of a piece of land. Land use information is typically collected in geospatial databases. As such databases become outdated quickly, an automatic update process is required. This paper presents a new approach to determine land cover and to classify land use objects based on convolutional neural networks (CNN). The input data are aerial images and derived data such as digital surface models. Firstly, we apply a CNN to determine the land cover for each pixel of the input image. We compare different CNN structures, all of them based on an encoder-decoder structure for obtaining dense class predictions. Secondly, we propose a new CNN-based methodology for the prediction of the land use label of objects from a geospatial database. In this context, we present a strategy for generating image patches of identical size from the input data, which are classified by a CNN. Again, we compare different CNN architectures. Our experiments show that an overall accuracy of up to 85.7% and 77.4% can be achieved for land cover and land use, respectively. The classification of land cover has a positive contribution to the classification of the land use classification.

1. INTRODUCTION

Classification of land cover is a standard task in remote sensing, in which each image pixel is assigned a class label indicating the physical material of the object surface (e.g. *grass, asphalt*). This task is challenging due to the heterogeneous appearance and high intra-class variance of objects, e.g. (Paisitkriangkrai et al., 2016). In contrast, *land use* describes the socio-economic function of a piece of land (e.g. *residential, agricultural*). A land use object can contain many different land cover elements to form complex structures, and a specific land cover type can be a part of different land use objects. Thus, land cover and land use classification based on remote sensing data are tasks pursuing different objectives (Barnsley & Barr, 2000).

The information about land use is often stored in geospatial databases, typically acquired and maintained by national mapping agencies. Such databases consist of objects represented by polygons that are assigned class labels indicating the objects' land use. In such a setting, which is also adopted in this paper, the primitives to be classified to derive land cover and land use are also different: land cover is determined for individual pixels, whereas land use is an attribute of polygons from an existing database. The primary goal of land use classification is updating the existing database, whereas land cover is an auxiliary product providing an additional (yet important) input for achieving that overall goal (Albert et al., 2017).

The classification of land cover and land use has mainly been tackled by supervised methods, because they are more easily transferable to other scenes than model-based techniques. A large variety of features and classifiers have been applied for that purpose, including methods incorporating context based on Conditional Random Fields (Albert et al., 2017). Recent work on the classification of images has focused on convolutional neural networks (CNN). Originally developed for predicting one class label per image (Krizhevsky et al., 2012), they have been expanded to pixel-based classification of images (*semantic segmentation*) (Badrinarayanan et al., 2017) and also to classification of land cover based on aerial images (Audebert et al., 2016; Paisitkriangkrai et al., 2016). CNN have outperformed other classifiers for pixel-based classification by a large margin if a sufficient amount of training data is available. For instance, the best-performing methods in the ISPRS semantic labelling challenge (Wegner et al., 2017) are based on CNN, e.g. (Marmanis et al., 2018). However, the application of CNN to the prediction of a land use label for an irregularly shaped polygon is not as straight-forward, because the convolution layers of a CNN need a regular image grid. To the best of our knowledge, up to date there is no application of CNN for the classification of land use polygons based on remote sensing data.

In this paper, we propose new methods for the classification of land cover and land use based on high-resolution digital aerial imagery and derived products such as a Digital Surface Model (DSM) and a Digital Terrain Model (DTM). The scientific contributions of this paper can be summarized as follows:

- We extend the SegNet architecture of Badrinarayanan et al., (2017) to other input data than RGB images and we propose a modified SegNet architecture having more layers while requiring fewer parameters.
- To the best of our knowledge, this is the first paper describing the classification of land use polygons based on CNN. This is achieved by converting the original input into a structure that can be classified by a CNN.
- For both tasks, we compare different network variants, and we assess the contributions of individual data sources to the classification results to highlight the benefits, but also to show remaining problems of the proposed methodology.

* Corresponding author

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-IV-3-251-2018 | © Authors 2018. CC BY 4.0 License.

The remainder of this paper is structured as follows. We start with a review of related work in section 2. Our approach for land cover classification is presented in section 3, whereas section 4 is dedicated to classification of land use objects. Section 5 describes the experimental evaluation of our approach. Conclusions and an outlook are given in section 6.

2. RELATED WORK

We start this review by a discussion of different strategies for land use classification from high-resolution remote sensing data, focusing on the overall strategy and the way in which land cover is integrated into the process. In the second part of this review, we discuss methodological aspects related to CNN.

Existing methods for land use classification differ by the primitives to be classified, the data sources, the features used for classification and the classifiers used to predict the class labels; cf. (Albert et al., 2017) for a recent overview. Many approaches rely on a strategy consisting of two steps (Hermosilla et al., 2012): After determining land cover in the first stage, the results are used to support the classification of land use in the second stage. The spatial distribution of land cover types within a land use object is analysed to define features that are combined with features derived from the sensor data. *Spatial metrics* quantify the spatial configuration of the land cover elements within a land use object, describing the size and the shape of the land cover segments (Hermosilla et al., 2012) or their spatial configuration (Novack and Stilla, 2015). *Graph-based-metrics* describe the frequency of the local spatial arrangement of land cover elements within a land use object, e.g. based on an adjacency-event matrix (Barnsley & Barr, 1996; Walde et al., 2014). Recent work has focused on including context into the classification process by using context features (Hermosilla et al., 2012) and Markov or Conditional Random Fields (CRF) (Montanges et al., 2015; Novack and Stilla, 2015; Albert et al., 2017). Sometimes, the contextual model is simple, involving assumptions about smoothness of labels at neighbouring sites that are not justified for larger entities such as land use objects (Montanges et al., 2015). Other approaches involve complex models of context based on classifiers that need a large amount of training data and are difficult to optimize (Albert et al., 2017). In the inference procedure for the simultaneous classification of land cover and land use described by Albert et al. (2017), the accuracy of land cover could be improved, but this was not the case for land use, which indicates that the procedure may have gotten stuck in a local optimum.

All the methods described so far rely on *hand-crafted features*. Recent advances in image-based classification that were also adapted for land cover classification (Paisitkriangkrai et al., 2016; Marmanis et al., 2018) relied on CNN, see also the recent overview of (Zhu et al., 2017). This resulted in a considerable improvement in the classification accuracy that can be achieved, which is usually attributed to the fact that using CNN, high-level features can be learned from training data. To the best of our knowledge, this principle has not yet been applied to the problem of the prediction of land use.

CNN consist of building blocks that combine a convolutional layer, a non-linear mapping and a pooling layer reducing the spatial resolution of the signal (LeCun et al., 1998). Deep CNN consist of a series of such blocks followed by fully connected (FC) layers. Originally, CNN only predicted one label per image, e.g. AlexNet (Krizhevsky et al., 2012). Deeper networks such as VGG-16 and VGG-19 (Simonyan & Zisserman, 2015) could further increase classification accuracy, though at the cost of requiring many parameters. GoogLeNet (Szegedy et al., 2015) is based on even more layers. Using more but smaller convolutional kernels in the convolutional layers, the number of parameters is smaller than in VGG-16 despite the increased depth, because the FC-layers are omitted. ResNet (He et al., 2016) offers an architecture that may have more than 100 layers. This is achieved by shortcut connections bypassing convolutional layers, so that only a residual function needs to be learned by the network. All of these methods only deliver one class label for an entire image (or image patch), but cannot be applied directly to obtain a prediction on a per-pixel level as is required for land cover classification.

To achieve this goal, an image can be divided into many patches, each of them being representative for the label of its central pixel. The pixels near the patch centre provide context information that is converted into a feature vector by the CNN. Längkvist et al. (2016) apply this procedure in a sliding window approach, making each pixel the centre of such a patch. However, processing overlapping patches requires unnecessary computations. Paisitkriangkrai et al. (2016) apply patch-based classification to every n^{th} pixel in the image, using bilinear interpolation of class scores to obtain dense predictions; however, bilinear classification does not preserve the object boundaries well.

An alternative to such patch-based approaches is to switch to network architectures that directly deliver class predictions for each pixel. Fully convolution networks (FCN) (Long et al., 2014) apply the convolutions and the pooling operations to the entire image rather than to a patch, which leads to a map of signals that has a lower spatial resolution. A final upsampling layer delivers the predictions at pixel level. Deconvolution networks (Noh et al., 2015) use an encoder-decoder strategy where the encoder part is similar to a standard CNN, whereas the decoder part is responsible for upsampling the low-resolution signal to the full resolution of the image. The decoder consists of several building blocks that mirror the structure of the encoder part of the network; the difference is that each block starts with an upsampling layer that increases the spatial resolution. A better representation of the object boundaries is achieved by storing the positions that delivered the signal in the max pooling layers and using them to distribute the low-resolution signal to the higher resolution layer. A similar strategy is pursued by SegNet (Badrinarayanan et al., 2017), applying end-to-end learning of all parameters, including those of the decoder part. Variants of such networks have been used for land cover classification, achieving promising results. For example, Marmanis et al. (2018) apply FCN and combine an ensemble of classifiers at different resolutions for that purpose, improving the delineation of boundaries by a network predicting image edges. Sherrah (2016) proposed a FCN without down-sampling to address the problem of the loss of spatial resolution, which could be achieved by *atrous* convolution. However, this method needs more computational effort (at least 40 times more training time) for an increase of 2% in accuracy in the ISPRS labelling challenge. Volpi et al. (2016) propose an encoder-decoder structure with different upsampling strategies. The architecture is more shallow than the one of SegNet, which reduces the reduction in spatial resolution in the encoder part because there are fewer pooling layers. The reduced depth might limit the expressive power at the prediction stage.

The prediction of class labels for land use objects is more closely related to object detection. In the context of CNN, this problem is solved in a procedure based on two stages: first, regions that are likely to contain some object are determined by generic region-proposal methods. After that, a CNN is used to classify these regions (Ren et al., 2015). We do not need region proposals, because we know the location and shape of the objects we want to classify from the geospatial database. However, the object size has large variations, so that rescaling of the bounding box enclosing an object may become infeasible. For instance, roads are very thin objects, yet their bounding boxes may become rather large, so that rescaling them to a window of a fixed size typical for CNN may almost completely remove them from the image. In this paper, we propose a method that allows to use a fixed-size CNN without object-dependent rescaling.

3. CNN-BASED CLASSIFICATION OF LAND COVER

Our land cover classification is based on SegNet (Badrinarayanan et al., 2017), because it delivers a class label for each pixel while outperforming the FCN of Long et al. (2014). According to Audebert et al. (2016), SegNet also provides a good balance between accuracy and computational cost. In section 3.1, we outline the original basic SegNet architecture, referred to as SegNet-B in this paper. Section 3.2 describes our extensions of the architecture. Section 3.3 gives information about the implementation, training and inference.

3.1 SegNet-B

SegNet (Badrinarayanan et al., 2017) applies a symmetric encoder-decoder structure. The encoder is a succession of four blocks, each consisting of one convolutional layer followed by batch normalization (BN; Ioffe et al., 2015), a rectified linear unit (ReLU) introducing non-linearity and max-pooling. The decoder part consists of four blocks that are symmetric to the encoder. Each block starts with an upsampling layer that distributes the lower resolution signals according to the positions of the pixels that were not suppressed by the corresponding max-pooling layer, which is followed by one convolutional layer, batch normalization and a rectified linear unit. Each convolutional layer consists of 64 filters with a size of 7 x 7 and uses zero-padding to keep the spatial dimension of the resultant feature maps. Max-pooling is always applied with a window of 2 x 2 and a stride of 2. At the end of the decoder part, there is a 1x1 convolutional layer that converts the output of the previous layers to a tensor of dimension $L \times H \times W$, where $H \times W$ is the size of the input image and L is the number of classes to be discerned. For each pixel i of the image to be classified, this results in a vector $\boldsymbol{z}_{\boldsymbol{LC}}^{\boldsymbol{i}} = (z_{LC^1}^i, \dots , z_{LC^L}^i)^T$ of class scores, where $\mathbb{C}_{LC} = \{C_{LC^1}, \dots, C_{LC^L}\}$ is the set of land cover classes and $z_{LC^c}^i$ is the class score for class C_{LC^c}. These class scores are normalised by a softmax function, the output of which can be interpreted as the posterior probability $P_i(C_{LC^c}|x)$ for pixel i to take class label C_{LC^c} given the image data *x*:

$$P_i(C_{LC^c}|x) = \text{softmax}\left(z_{LC}^i, C_{LC^c}\right) = \frac{exp(z_{LC^c}^i)}{\sum_{l=1}^{L} exp(z_{LC^l}^i)}. \qquad (1)$$

In training, the parameters of all convolutional layers are determined, including those of the decoder part of the network. Training uses stochastic gradient descent (SGD) based on mini-batches and backpropagation for computing the gradients. The function to be minimized by SGD is the cross-entropy loss:

$$L = -\frac{1}{W \cdot H \cdot N} \sum_{c,i,k} \left[\omega_c \cdot y_{LC^c}^{ik} \cdot log\left(P_i(C_{LC^c}|X_k)\right)\right], \qquad (2)$$

where k is the index of an image, X_k is the k^{th} image in the mini-batch and N is the number of images in a mini-batch. The indicator variable $y_{LC^c}^{ik}$ is 1 if the training label of pixel i in image k is identical to C_{LC^c} and 0 otherwise, and ω_c is a class weight computed according to (Eigen et al., 2015) in order to compensate for an imbalanced class distribution in the training data. The sum in (2) is taken over all potential class labels for all pixels of all images of a mini-batch.

3.2 Network variants

Based on SegNet-B, we developed two additional variants of the network, referred to as SegNet-F and SegNet-O. We also apply combinations of different networks (ensemble methods).

3.2.1 SegNet-F: SegNet-B requires RGB images as input, but it cannot cope with an additional input such as a DSM and an infrared band (IR). This is usually achieved by using different CNN branches for the individual input sources. Unlike existing methods, e.g. (Sherrah, 2016; Hazirbas et al., 2015), we use three branches: the first one corresponds to SegNet-B and is applied to the RGB orthophotos, the other two are also based on SegNet-B, but they use one input band only (DSM and IR, respectively). After the last convolutional layer of the decoder parts of the three branches, their output is concatenated spatially, which is the same as the concatenation in (Sherrah, 2016). Finally, a 1 x 1 convolutional layer is used to convert the concatenated signals into class scores, which are used as arguments for the softmax function.

3.2.2 SegNet-O: SegNet-B uses a filter size of 7 x 7 pixels for all convolutional layers. Following the ideas of Simonyan and Zisserman (2014), we replace these convolutional layers by three successive blocks of convolution, BN and ReLU, where the filter size of the convolution is 3 x 3, keeping the number of filters in each convolutional layer the same as in SegNet-B. This modification adds an implicit regularization to the model due to more BN and non-linearity layers. SegNet-O is deeper than SegNet-B (12 vs. 4 convolutional layers both in the encoder and decoder phases), but achieves a reduction of the number of parameters of 40%. SegNet-O is illustrated in Fig. 1.

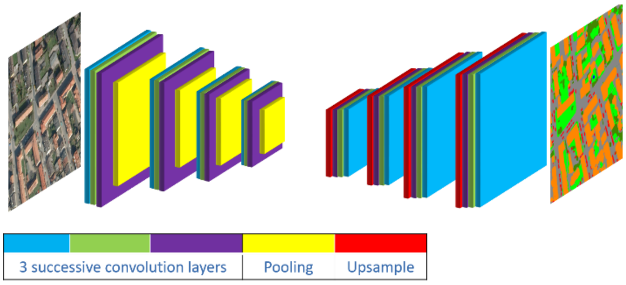

Figure 1: Architecture of SegNet-O.

3.2.3 Ensembles: We also test ensembles of the networks described so far. For that purpose, an image is classified using all networks of the ensemble; we combine the results by multiplying the probabilistic class scores for each pixel.

3.3 Implementation and Training

All networks are implemented based on the Caffe framework (Jia et al., 2014). We use a GPU (Nvidia GTX 1060, 6GB) to accelerate training and inference.

3.3.1 Training: For training of all the networks we employed a SGD optimizer with weight decay 0.00015, momentum 0.9 and step learning policy. The input size for all networks is 256 x 256 pixels. Due to the limitations of our GPU, the mini-batch size and base learning rate are set to different values for different architectures. For SegNet-B, the learning rate is set to 0.1 with a mini-batch size of 18; for SegNet-F and SegNet-O, the corresponding values are 0.01 and 4, respectively.

3.3.2 Transfer learning: For SegNet-B we also compare a variant based on training from scratch (cf. section 3.3.1) with another one in which we initialize the weights from the original SegNet trained on the CamVid dataset (Brostow et al., 2009). Starting from this initialisation, we fine-tune the network using SGD with a learning rate of 0.01 and a mini-batch size of 18.

4. CNN-BASED CLASSIFICATION OF LAND USE

The classification of land use is based on a CNN taking an image patch of 256 x 256 pixels and returning a land use label. Again, the networks are based on Caffe with GPU support. Before applying the CNN, the patch is prepared so that it represents the land use object from the geospatial database for which the class label is derived. In this context, large land use objects are split into several patches. Patch preparation is described in section 4.1, whereas section 4.2 outlines the CNN architectures used for classification.

4.1 Patch preparation

The image patch should represent the image data inside the polygon corresponding to an object from the geospatial database. As the images may have a very high resolution, a patch of 256 x 256 pixels may correspond to a very small area in object space. Thus, we start by rescaling the images using a uniform scale factor so that the resultant patches correspond to an area of about 100 m x 100 m in object space, a value we found to be appropriate in preliminary experiments. If the polygon fits into a 256 x 256 patch after rescaling, patch preparation is straightforward. We initialise the patch by a background colour (black) and place the polygon in the centre of the patch. After that, for each pixel inside the polygon we replace the background colour by the RGB values from the rescaled image. In this way, the polygon shape is represented by the transition between RGB data and the background, which we expect to be beneficial for the classification.

For polygons that do not fit into a single patch, we define a rectangle that is aligned with the row and column directions of the image and split it into a series of tiles of 256 x 256 pixels with 25% overlap between neighbouring tiles. For each tile we check the proportion of its area that is inside the object from the database; we exclude tiles having an overlap smaller than a threshold (set to 99.995% of its area) from further processing. As this may still result in a large number of tiles, we randomly select 30% of the remaining tiles for further processing. Each tile results in a patch to be classified. A tile is initialised by the background colour; after that, the RGB values of pixels inside the land use objects are copied from the rescaled image. Patch generation for a land cover image, which serves as additional input, works in a similar way.

4.2 Network Variants

4.2.1 LiteNet-B: This architecture is based on (Paisitkriangkrai et al., 2016) and serves as our baseline. We use it because it requires much fewer parameters than other networks (2.7 million, compared e.g. to 140 million of VGG-16) while still promising a similar performance. The input size for all networks is 256 x 256 pixels. The network consists of four convolution layers, each followed by a ReLU and a max-pooling layer with a window of 2 x 2 and stride 2. The first three convolutional layers have 32, 64, and 96 filters of size 5 x 5, respectively. The fourth convolutional layer has 128 filters with a size of 3 x 3. No padding is applied in any convolutional layer. After the last pooling layer there are two FC layers having 128 neurons each, each of them followed by a dropout layer with dropout ratio 0.5. The last FC layer converts the resultant feature vectors into a vector of class scores $\boldsymbol{z_{LU}} = (z_{LU^1}, \dots, z_{LU^M})^T$, where $\mathbb{C}_{LU} = \{C_{LU^1}, \dots, C_{LU^M}\}$ is a set of land use classes and z_{LU^c} is the score of for class C_{LU^c}. To get a probabilistic class score, the softmax function (eq. 1) is applied to the class scores, thus $P(C_{LU^c}|\mathrm{X}) = \mathrm{softmax}(\boldsymbol{z}_{LU}, C_{LU^c})$. Training is based on SGD; the function to be optimised is the cross-entropy loss

$$L = -\frac{1}{N} \cdot \sum_{c,k} \left[y^k_{LU^c} \cdot log\left(P(C_{LU^c}|X_k)\right)\right], \qquad (3)$$

where X_k is the k^{th} image in the mini-batch, N is the number of images in a mini-batch, and $y^k_{LU^c}$ is 1 if the training label of X_k is C_{LU^c} and 0 otherwise.

4.2.2 LiteNet-O: We extend LiteNet-B by replacing all convolutional layers by two successive 3 x 3 convolutional layers, each followed by a ReLU, for the reasons already given in section 3.3.2. The last convolution in LiteNet-B has a filter size of 3 x 3, but we also replace it by two successive 3x3 convolutional layers, so the receptive field of the combined convolution is larger than the one of the last LiteNet-B layer. Figure 2 shows the resultant network architecture.

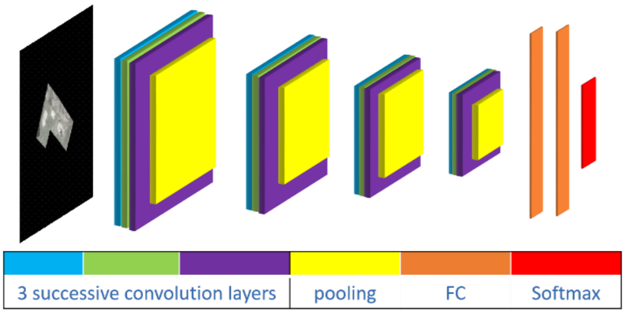

Figure 2: Architecture of LiteNet-O.

4.2.2 Ensembles: We also test a combination of the results of different classifiers, similarly to section 3.2.3.

4.2.3 Training and inference: For all variants, training uses SGD with weight decay 0.00015, momentum 0.9 and step learning policy. The base learning rate is set to 0.001 with a mini-batch size of 32. In the classification, the CNN delivers a prediction for each patch. For polygons that had to be split into multiple patches, the probabilistic class scores of all patches are multiplied to obtain a combined score for the compound object.

5. EXPERIMENTS

5.1 Test Data und Test Setup

The approach is evaluated using a test site located in the city of Hameln (Germany). It is characterised by residential areas with detached houses as well as by densely built-up areas in the centre of the city, but there are also industrial areas, rural areas and rivers. The test site covers an area of 2 km x 6 km. The input consists of digital orthophotos (DOP), DSM, DTM and land use objects corresponding to cadastral parcels of the German

Authoritative Real Estate Cadastre Information System (ALKIS) (Albert et al., 2017). The DOP are multispectral images (RGB + infrared / IR) with a ground sampling distance of 20 cm. We generated the normalised DSM (nDSM) by subtracting the DTM from DSM; the nDSM is used to represent the height information in the experiments. A pixel-based reference for land cover was generated by manual labelling. We could use 37 manually labelled images for land cover classification, each having a size of 768 x 768 pixels (153.6 m x 153.6 m in object space). The reference for land use classification was derived from the geospatial database. We distinguish eight land cover classes: *building (build.), sealed area (seal.), bare soil (soil), grass, tree, water, car* and *others*. The definition of land use classes has to comply with the specifications of the German geospatial land use database; we distinguish ten land use classes: *residential (res.), non-residential (non-res.), urban green (green), traffic (traf.), square, cropland (cropl.), grassland (grassl.), forest, water body (water)* and *others*. The class structures of land cover and land use are the same as in (Albert et al., 2017).

The evaluation is based on cross validation. We split the data into subsets and use some subsets for training and the others for testing, repeating this process multiple times so that in the end each object is used for testing once. After each test run, we compare the results to the reference and determine the confusion matrix as well as derived metrics. Here, we report average metrics over all test runs. We focus on the overall accuracy (OA), i.e. the percentage of entities (pixels for land cover, objects for land use) that are assigned the correct class label by the classification process, and the average F1 score, i.e., the average of the harmonic means of the completeness and the correctness per class.

5.1.1 Test setup for land cover classification: For evaluating the land cover classification, we split each of the 37 images with reference data into nine non-overlapping tiles of size 256 x 256, which results in 333 tiles. The tile size corresponds with the input size required by our CNN variants. These tiles were divided into three subsets of equal size, and in each test run we use one subset for training and one for testing, so that there are altogether three test runs. For training, we applied data augmentation by flipping the training samples in horizontal and vertical directions and applying rotations of 90°, 180° and 270°. We tested Segnet-B in two scenarios: SegNet-B0 is based on training from scratch, whereas SegNet-B1 used a pre-trained model for initialisation (cf. section 3.3.2); in both cases, the RGB bands of the DOP were the only input. The latter statement is also true for the experiments using SegNet-O, whereas for SegNet-F we used the RGB and IR bands as well as the nDSM. We also compared different ensembles (EN), where we drop the term SegNet to denote the classifiers that were combined. For instance, EN(O, F) refers to an ensemble that combines the output of SegNet-O and SegNet-F.

5.1.2 Test setup for land use classification: The test data set is split into twelve blocks of 5000 x 5000 pixels (1 km^2) each. In each test run, we used 11 blocks for training and one for testing. Land use objects crossing the block boundaries were split at these boundaries; after splitting these polygons, the reference consisted of 4155 land use objects. For patch generation, a scale of 1:2 was used which corresponds to a ground sampling distance of 40 cm. For training we applied data augmentation by horizontal and vertical flipping and by rotations of 45°, 90°, 135°, 180°, 225°, 270° and 335°. Patch generation and data augmentation resulted in 154570 patches. We carried out two tests for each of the two architectures: in the first test, RGB images were used as the basis of classification; these variants are referred to LiteNet-B0 and LiteNet-O0, respectively. In the second set of tests (LiteNet-B1, LiteNet-O1), we used a label image encoding the land cover as the single input. This means that the CNN had to be adapted to use a single image band as input, i.e., the filters in the first convolution layers have different numbers. In the tests involving land cover as input, we used an ensemble of all land cover classifiers to derive the land cover information. For both CNN architectures, we also tested an ensemble of the nets based on RGB and land cover data, referred to as EN(B0, B1) and EN(O0, O1), respectively. In the classification process, the set of geometrical transformations used for data augmentation was also applied to each test patch, so that each patch was classified 10 times by the CNN; the combined class scores were obtained by multiplying the scores from the individual results.

5.2 Evaluation of land cover classification

Table 1 presents the land cover classification results for all variants described in section 5.1.1. In general, we consider the results to be quite good, with OA being larger than 81% in all cases. The F1 scores show that the correct classification of classes having few training samples (*car* and *others*, covering 1.0% and 0.5% of the test area, respectively) is more challenging than the classification of other classes. In the subsequent sections, we analyse the variants in more detail.

5.2.1 SegNet-B: SegNet-B trained from scratch (SegNet-B0) yields a mean overall accuracy of about 81.7% and an average F1 score of about 72.1%, with particularly low F1 scores for *car* and *others*. When applying transfer learning (SegNet-B1), the OA is improved by a small margin (0.2%), the most obvious effect being an improvement of the F1 scores for *water* and *car* (2.3% / 2.2%, respectively). This is in accordance with the findings of Yosinski et al. (2014), who show that results after transfer learning may be slightly better than results achieved when training from scratch because the classifier seems to "remember" previously seen samples even if taken from a different domain. It is to be expected that training from scratch and fine-tuning will arrive at different minima in parameter space, which makes the two classifiers complementary. This is confirmed by the results achieved by a combination of the two classifiers, EN(B0, B1), which results in an improvement of 1.9% in OA and 3% in the average F1 score compared to SegNet-B. The F1 scores of all classes are improved, most obviously for *car* and *others* (6.1% / 5.4%).

5.2.2 SegNet-O: Our extension achieves similar results as the baseline while only requiring 40% of the parameters. Despite the network being deeper, there is only a small improvement in OA and average F1 score of 0.1% and 0.4%, respectively. An improvement in the F1 score for class *car* of 6.7% is contrasted by a decrease in class *soil* of 4%. By combining Segnet-O with SegNet-B (variant EN(B0, O)) the results are improved to a level similar to EN(B0, B1), with advantages for *car* and *others* and slight disadvantages for *water*.

5.2.3 SegNet-F: This network integrates RGB, IR and nDSM data in one model. We expected the additional channels of IR and nDSM to improve the results, but this is not the case: SegNet-F performs slightly worse than the baseline in terms of OA and average F1 score. One reason for that behaviour could be that the SegNet-F requires almost three times as many parameters as SegNet-B. Consequently, it may be necessary to use more training data, but in our experiments, the same number of samples was used for training. There are also some problems with the quality of the DSM at building boundaries and with trees. However, the benefits of the additional information sources are indicated by the results achieved by the ensembles EN(B0, F) and EN(O, F) of SegNet-F with SegNet-B0 and SegNet-O, respectively. Compared to the baseline, OA is improved by about 3% in both cases, and there is also an improvement of 1% compared to the best ensemble methods without SegNet-F. It would seem that SegNet-F indeed covers cases that are not represented well by the nets not relying on IR and height data. However, the improvement is not distributed equally over the classes: *build.*, *seal.* and *water* show the largest improvement, whereas the F1 scores of *others* and *soil* are considerably lower than for ensembles without SegNet-F.

5.2.4 EN(B0, B1, O, F) - Ensemble of all classifiers: Finally, we tested an ensemble of all the classifiers, which clearly outperformed all other variants with an OA of 85.7% and an average F1 score of 76.6%. This is an improvement of 4% compared to the baseline and of 1% compared to ensembles of two nets and involving SegNet-F (section 5.2.3). Nearly all F1 scores are better than for all other variants; the only F1 score that is notably lower is the one for *others*.

5.3 Evaluation of land use classification

Table 2 presents the results of the land use classification for different networks. In general, the quality indices are at a lower level than the ones for land cover, which may be partly attributed to the fact that despite data augmentation, the number of available training samples is still very low for some classes.

5.3.1 LiteNet-B: LiteNet-B0, only based on RGB images, achieved an OA of 76.1% and an average F1 score of 61.2%. The CNN has problems in classifying objects of types *square, grassland* and *others*, which we attribute to the small amount of training samples available for these data (only 1.2%, 1.3% and 5% of all training samples belong to *grassland*, *square*, and *others*, respectively; in addition, *others* is a class of very heterogeneous appearance). Although the OA is slightly lower when using the semantic classification results as input (LiteNet-B1), the average F1 score is almost the same. For some of the classes, the F1 score is improved by a margin of up to 15.6%. (*square*) and for others it is reduced by up to 6.1% (*others*). Using the ensemble EN(B0, B1) results in an improvement of OA of 1.3% and the average F1 score of 1.9%, compared to LiteNet-B0. Except for the classes *grassland* and *others*, the F1 scores are improved by adding the land cover data to the RGB images. This shows that the two networks (based on different input data) complement each other, resulting in a better overall performance. Compared to the results of LiteNet-B1, there is a loss in F1 score in the classes *square* and *grassland* by adding the RGB information.

5.3.2 LiteNet-O: In general, our extensions LiteNet-O0 and LiteNet-O1 do not outperform the corresponding LiteNet-B variants. While delivering OAs of a similar level, the average F1 scores are smaller. Compared to LiteNet-B0, LiteNet-O0 achieves slightly better F1 scores for the classes *residential, non-residential, traffic, forest, water* and *others* (improvement between 0.2% and 3.8%). This is contrasted by a drop in F1 scores for the other classes, in particular for *square* (16.5%). Similarly to the LiteNet-B variants, using land cover rather than RGB data as input leads to a slightly lower OA, yet improves the average F1 score. The latter is due to an increased F1 score of most classes (up to 17.3% in case of *square*), though not for all of them; most notably, there is a decrease of 13.2% for *water*. As with the LiteNet-B variants, the ensemble EN(O0, O1) delivers better results than both LiteNet-O0 and LiteNet-O1, which again confirms that the land cover data and RGB are complementary. However, in this case the improvement for the ensemble is smaller than for the LiteNet-B variants. Whereas the OA of EN(B0, B1) is slightly larger than the one achieved for EN(O0, O1) (0.4%), the difference in the average F1 score is 2.4%. The LiteNet-B variants achieve more homogeneous results for all classes than LiteNet-O. In the light of these results, the ensemble of the LiteNet-B variants using both RGB and land cover data is the best of the compared methods.

5.3.3 Influence of the patch generation strategy: In our patch generation strategy we set the area outside of the object to zero, thus clearly separating the object from its background. In this way, the CNN can implicitly learn a model of the shape of the object boundaries, which we consider important because in (Albert et al., 2017) we found that shape features were among the most important ones for land use classification. Without setting the background to zero, the object shape, related to property boundaries, would not be reflected clearly in the patches to be classified. This is illustrated by Figure 3, which shows two representative feature maps from the first convolutional layer of LiteNet-B0 for a land use object. In the top part of the figure, the

Network	Input	F1 [%]								avg. F1	OA
variant		*build.*	*seal.*	*soil*	*grass*	*tree*	*water*	*car*	*others*	[%]	[%]
SegNet-B0	RGB	85.5	74.7	78.7	83.6	84.2	86.8	57.4	25.5	72.1	81.7
SegNet-B1	RGB	85.4	75.4	78.6	83.4	83.7	89.1	59.6	25.9	72.6	81.9
EN(B0, B1)	RGB	87.2	77.4	81.1	84.8	85.3	90.1	63.5	30.9	75.1	83.6
SegNet-O (O)	RGB	85.7	76.0	74.7	82.4	84.4	86.3	64.1	26.1	72.5	81.8
EN (B0, O)	RGB	87.6	78.0	80.9	84.4	85.6	88.1	65.9	**31.3**	75.3	83.7
SegNet-F	DOP + nDSM	87.7	77.2	75.3	80.1	80.7	88.6	50.8	22.1	68.8	81.3
EN (B0, F)	DOP + nDSM	90.8	80.3	77.7	84.7	85.5	**91.6**	58.9	26.7	74.5	84.5
EN (O, F)	DOP + nDSM	90.7	80.6	76.4	84.6	86.0	91.3	63.9	25.8	74.9	84.7
EN (B0, B1, O, F)	DOP + nDSM	**90.8**	**81.4**	**81.2**	**85.7**	**86.8**	91.5	**66.1**	29.6	**76.6**	**85.7**

Table 1. Results of land cover classification. Network variant: cf. section 5.1.1. F1: F1 score, OA: Overall Accuracy, both evaluated on the basis of pixels. Best scores are printed in bold font.

Network	Input	F1 [%]										avg.	OA
variant		*res.*	*non-res.*	*green*	*traf.*	*square*	*cropl.*	*grassl.*	*forest*	*water*	*others*	F1[%]	[%]
LiteNet-B0	RGB	80.7	64.6	68.8	88.0	31.8	78.2	34.4	73.8	60.5	31.5	61.2	76.1
LiteNet-B1	LC	81.1	68.2	64.4	87.9	**47.4**	78.2	**37.8**	76.3	57.3	25.4	61.4	75.7
EN (B0, B1)	RGB + LC	82.0	**70.8**	**69.2**	88.2	40.4	**80.6**	33.3	**77.0**	**63.4**	25.9	**63.1**	**77.4**
LiteNet-O0	RGB	81.4	66.0	67.3	88.2	15.3	77.4	25.6	75.3	60.8	**35.3**	59.2	76.1
LiteNet-O1	LC	80.8	68.2	64.3	**88.4**	32.6	78.9	36.1	76.6	47.2	26.9	60.0	75.7
EN (O0, O1)	RGB + LC	**83.1**	69.4	67.8	88.1	23.3	79.6	27.9	**77.0**	60.0	30.6	60.7	77.0

Table 2: Results of land use classification. Network variant: cf. section 5.1.2. LC: Land cover. F1: F1 score, OA: Overall Accuracy, both evaluated on the basis of objects. Best scores are printed in bold font.

patch is extracted from the RGB image without setting the background to zero; the resultant feature maps are related to image edges and corners that correspond to land cover changes or shadows. Applying our patch generation strategy yields feature maps that clearly highlight the object boundaries, while still responding to edges inside the object (bottom part of Figure 3).

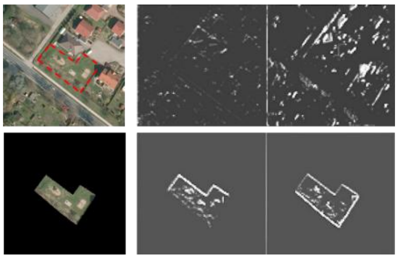

Figure 3: Feature maps from the first convolutional layer for a land use object (right) and corresponding image patch (left). Top / bottom: without / with setting the grey values outside the object to zero.

5.3.4 Influence of object size: In section 5.3.1, we have seen that the performance of land use classification is related to the number of training samples per class. Here we analyse how the polygon size affects the classification accuracy. Table 3 shows the OA achieved by EN(B0, B1) for three different sets of land use objects. The set *small* consists of all objects that were represented by a single patch in the classification process, whereas the set *large* consists of all objects that were split in the patch generation phase (cf. section 4.1). The table also contains the combined results (*all*), which are identical to the ones shown in Table 2. The results show that the large objects are classified more reliably (OA 85.3%) than the smaller ones (OA 69.5%).

object set	*large*	*small*	*all*
objects	2076	2079	4155
OA [%]	85.3	69.5	77.4

Table 3: OA for three different sets of objects based on EN(B0, B1); objects: number of objects in the set.

We also evaluate the OA of the objects from the *small* set as a function of the ratio of the object area and the tile size (*occupation ratio*; Table 4). In nearly 75% of all cases, the object covers less than 10% of the tile, and the corresponding OA is low (66.9%). In general, the OA increases with the object size, achieving 79.1% for objects between 10% and 20% of the patch size; the number for the largest objects, based on very few instances, is not representative. However, even for the largest objects, the OA remains smaller than the one for the objects in the *large* set. This evaluation clearly gives us a direction of conducting further research, focussing on a better classification of small objects. This could for instance be achieved by involving multiple scales for resizing the image patches, so that also for small objects a larger proportion of the patch to be classified is different from the background.

6. CONCLUSION

In this work, we have investigated the use of different encoder-decoder structures of CNN based on SegNet for the pixel-wise classification of land cover based on aerial images and derived data. We compared different variants of the CNN architecture. Our experiments have shown that an ensemble of CNN having different architectures and using different input data achieves the best performance with an overall accuracy of almost 86% for eight land cover classes, whereas it was 83.7% in Albert et al., 2017. However, there are still some misclassifications; for instance, the *asphalt* on bridges is sometimes misclassified as *building*, or *bare soil* is misclassified as *grass*. At the same time, the boundaries between objects, e.g. between *building* and *street*, are not very precise. These problems will be addressed in our future work on land cover classification, where we will investigate deeper network structures and architectures that place more focus on a precise delineation of the object boundaries.

We have also proposed a new method for the classification of land use objects based on CNN and a task-specific patch generation strategy. Although the overall accuracy is about 1% lower than the ones in Albert et al., 2017, the results are still very promising, in particular for large objects. A small number of training samples and small object size could be identified as the major limiting factors. We have shown that integrating the results of land cover classification improves the classification of land use. We compared networks of different depth, but did not achieve better results for deeper networks than for more shallow ones. Our future work will focus on improving the method by concentrating on its current limitations. Firstly, a multi-scale analysis could help to improve the classification accuracy for small objects. The problems related to the number of training samples could be tackled by label noise robust methods for training (Mnih & Hinton, 2012) that can leverage the (existing but potentially outdated) land use labels from the geospatial database for training. Post-processing by a CRF (Albert et al., 2017) could also improve the results. Finally, we want to develop an improved framework for the integration of land cover and land use classification.

OR [%]	<10	10 - 20	20 - 30	>30
objects	1557	373	129	20
OA [%]	66.9	76.7	79.1	70.0

Table 4: OA as a function of the occupation ratio (OR); objects: number of objects in the respective set.

ACKNOWLEDGEMENT

We thank the Landesamt für Geoinformation und Landesvermessung Niedersachsen(LGLN), the Landesamt für Vermessung und Geoinformation Schleswig Holstein (LVermGeo) and Landesamt für innere Verwaltung Mecklenburg-Vorpommern (LaiV-MV) for providing the test data and for their support of this project.
The first author is an associate member of the Research Training Group i.c.sens (GRK 2159), funded by the German Research Foundation (DFG).

REFERENCES

Albert, L., Rottensteiner, F., Heipke, C., 2017. A higher order conditional random field model for simultaneous classification of land cover and land use. ISPRS Journal of Photogrammetry and Remote Sensing 130: 63-80.

Audebert, N., Saux, B. L., Lefevre, S., 2016. Semantic segmentation of earth observation data using multimodal and multi-scale deep networks. In: Asian Conference on Computer Vision, pp. 180-196.

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.

Barnsley, M. J. & Barr, S. L., 1996. Inferring urban land use from satellite sensor images using kernel-based spatial reclassification. Photogrammetric Engineering & Remote Sensing 62(8): 949–958.

Barnsley, M. J. & Barr, S. L., 2000. Monitoring urban land use by earth observation. Surveys in Geophysics 21(2): 269-289.

Brostow, G. J., Fauqueur, J., Cipolla, R., 2009. Semantic object classes in video: A high-definition ground truth database. Pattern Recognition Letters 30(2): 88-97.

Eigen, D., Fergus, R., 2015. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: IEEE International Conference on Computer Vision, pp. 2650-2658.

Girshick, R. B., Donahue, J., Darrell, T., Malik, J., 2013. Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 580-587.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770-778.

Helmholz, P., Rottensteiner, F., Heipke, C., 2014. Semi-automatic verification of cropland and grassland using very high resolution mono-temporal satellite images. ISPRS Journal of Photogrammetry and Remote Sensing 97: 204-218.

Hermosilla, T., Ruiz, L. A., Recio, J. A., Cambra-López, M., 2012. Assessing contextual descriptive features for plot-based classification of urban areas. Landscape and Urban Planning, 106(1): 124-137.

Ioffe, S., Szegedy, C., 2015. Batch Normalization: accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 448-456.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T., 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. In: ACM International Conference on Multimedia, pp. 675-678.

Krizhevsky, A., Sutskever, I., Hinton, G. E., 2012. ImageNet classification with deep convolutional neural networks. In: International Conference on Neural Information Processing Systems (NIPS'12) 25 Vol. 1, pp. 1097-1105.

Längkvist, M., Kiselev, A., Alirezaie, M., Loutfi, A., 2016. Classification and segmentation of satellite orthoimagery using convolutional neural networks. Remote Sensing 8(4): 329-350.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11): 2278–2324.

Long, J., Shelhamer, E., Darrell, T., 2014. Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431-3440.

Marmanis, D., Schindler, K., Wegner, J. D., Galliani, S., Datcu, M., Stilla, U., 2018. Classification with an edge: Improving semantic image segmentation with boundary detection. ISPRS Journal of Photogrammetry and Remote Sensing 135: 158–172.

Mnih, V., Hinton, G., 2012. Learning to label aerial images from noisy data. In: International Conference on Machine Learning, pp. 567-574.

Montanges, A.P., Moser, G., Taubenböck, H., Wurm, M., Tuia, D., 2015. Classification of urban structural types with multisource data and structured models. In: IEEE Joint Urban Remote Sensing Event (JURSE), pp. 1–4.

Novack, T., Stilla, U., 2015. Discrimination of urban settlement types based on space-borne SAR datasets and a conditional random fields model. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences II-3/W4, pp. 143–148.

Noh, H., Hong, S., Han, B., 2015. Learning deconvolution network for semantic segmentation. In: IEEE International Conference on Computer Vision, pp. 1520-1528.

Paisitkriangkrai, S., Sherrah, J., Janney, P., Hengel, A., 2016. Semantic labeling of aerial and satellite imagery. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 7(9): 1-14.

Ren S., He, K., Girshick, R., Sun, J. 2015. Faster R-CNN: towards real-time object detection with region proposal networks. In: International Conference on Neural Information Processing Systems (NIPS'15) 28 Vol. 1, pp. 91-99

Sherrah, J., 2016. Fully convolutional networks for dense semantic labelling of high-resolution aerial imagery. ArXiv:1606.02585.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: International Conference for Learning Representations.

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(1): 1929-1958.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. In: IEEE Conference on Computer Vision and Pattern Recognition.

Volpi, M., Tuia, D., 2017. Dense semantic labeling of sub-decimeter resolution images with convolutional neural networks. IEEE Transactions on Geoscience and Remote Sensing 55(2): 881-893.

Walde, I., Hese, S., Berger, C., Schmullius, C., 2014. From land cover-graphs to urban structure types. International Journal of Geographical Information Science 28(3): 584–609.

Wegner, J.D., Rottensteiner, F., Gerke, M., Sohn, Gunho, 2017. The ISPRS labelling challenge. Available in the WWW: http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html (accessed 11/01/2018).

Yosinski, J., Clune, J., Bengio, Y., Lipson, H., 2014. How transferable are features in deep neural networks? In: International Conference on Neural Information Processing Systems (NIPS'14) 27, Vol. 2, pp. 3320-3328.

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, Vol. 5, pp. 8-36