Multi-label Pixelwise Classification for Reconstruction of Large-scale Urban Areas

Yuanlie He
School of Computer Science and Technology
Guangdong University of Technology
University-Town, Guangzhou
P. R. China (510006)
Email: heyuanlie@gdut.edu.cn

Sudhir Mudur
Concordia University, CA
Email: sudhir.mudur@concordia.ca

Charalambos Poullis
Immersive and Creative Technologies Lab
Concordia University, CA
Email: charalambos@poullis.org

***Abstract*—Object classification is one of the many holy grails in computer vision and as such has resulted in a very large number of algorithms being proposed already. Specifically in recent years there has been considerable progress in this area primarily due to the increased efficiency and accessibility of deep learning techniques. In fact, for single-label object classification [i.e. only one object present in the image] the state-of-the-art techniques employ deep neural networks and are reporting very close to human-like performance.**

There are specialized applications in which single-label object-level classification will not suffice; for example in cases where the image contains multiple intertwined objects of different labels. In this paper, we address the complex problem of multi-label pixelwise classification. We present our distinct solution based on a convolutional neural network (CNN) for performing multi-label pixelwise classification and its application to large-scale urban reconstruction. A supervised learning approach is followed for training a 13-layer CNN using both LiDAR and satellite images. An empirical study has been conducted to determine the hyperparameters which result in the optimal performance of the CNN. Scale invariance is introduced by training the network on five different scales of the input and labeled data. This results in six pixelwise classifications for each different scale. An SVM is then trained to map the six pixelwise classifications into a single-label. Lastly, we refine boundary pixel labels using graph-cuts for maximum a-posteriori (MAP) estimation with Markov Random Field (MRF) priors. The resulting pixelwise classification is then used to accurately extract and reconstruct the buildings in large-scale urban areas. The proposed approach has been extensively tested and the results are reported.

***Keywords*—urban reconstruction, remote sensing processing**

I. Introduction

Recent advances in the efficiency and accessibility of deep learning techniques have had a significant impact on the progress of many important and at the time dormant problems in computer vision. In particular, object recognition has greatly benefited since for many years the state-of-the-art had been almost restricted to minimal and incremental progress whereas currently human-like performances in object recognition [14], [15] are being reported, albeit for images with a single object.

Many successful applications which rely on single-label object recognition using deep neural networks have already been reported. The assumption with single-label object recognition is that a network can be trained to recognize objects from various categories and identify their general location [in the form of a bounding box] provided that the image contains exactly one object. More recently, it has been shown how this learning pipeline can be extended to handle cases where multiple spatially separable objects are present in an image i.e multi-label object recognition [16]. However, in certain applications the images may contain objects which are intertwined. Moreover, rather than the general location, the precise location of the object is required. One such application is the classification of geospatial objects for reconstruction of large-scale urban areas. The data is in the form of geometry captured with LiDAR and satellite RGB images. The geospatial objects present in the data are buildings, roads, trees, artificial ground, natural ground, cars, etc and all these are perfectly intertwined i.e. a building is surrounded by ground, etc. One can think of the data (LiDAR, images) as being perfectly tesselated by many objects from each of these categories.

In this paper we address the problem of multi-label pixelwise classification for large scale urban reconstruction and propose our distinct solution based on a 13-layer convolutional neural network (CNN). The CNN is trained using both LiDAR and satellite RGB images in multi-scale format, capturing large-scale urban areas and produces an output of six pixelwise values which are interpreted as likelihoods of the pixel in being a building, road, tree, car, or ground [artificial, natural]. An SVM linear classifier takes the likelihoods as inputs and maps them to a single label. In the final step, boundary pixel labels are refined.

Our technical contributions are:

- the design, development and supervized training of a 13-layer convolutional network. The CNN is specifically designed for the classification of geospatial objects appearing in remote sensor data and in particular LiDAR and satellite RGB images into the following six classes: buildings, roads, tree, cars, natural ground and artificial ground.
- a method for introducing scale invariance during the training. Due to the multiple scales the CNN produces a likelihood-per-scale per pixel. These are further processed and combined into a single label by training an SVM. The labels are finally refined using graph cuts for maximum a-posteriori (MAP) estimation with Markov Random Field (MRF) priors which also ad-

dresses the problem of boundary pixels not being assigned labels.

- a complete framework for the geospatial object classification and reconstruction of large-scale urban areas. The multi-label pixelwise classification is used to reconstruct the buildings of large-scale urban areas. Generic objects such as cars and trees are replaced by procedurally generated models to yield realistic 3D visualizations.

A. Paper Organization

The paper is organized as follows: Section II presents an overview of the state-of-the-art in the area of object recognition and large-scale urban reconstruction. In Section III we present a technical overview of our proposed technique and in Section IV we provide a brief description of the dataset used. The architecture of the developed network is described in detail in Section V including the training, refinement, validation and classification results. Section VI presents how these classification results are used in the context of large-scale urban reconstruction. The conclusion and future work are discussed in Section VII.

II. Related Work

Object classification has been an active research topic in computer vision for many years and large-scale urban reconstruction for even more. In fact, object recognition is often employed as the first step in reconstruction for identifying the geospatial objects present in the scene. In this section we provide a brief overview of the state-of-the-art in the areas related to this work in object recognition using neural networks and in large-scale urban reconstruction.

A. Object Classification

The first Convolutional Neural Networks (CNN) was introduced by LeCun [1] for hand writing recognition and could achieve outstanding performance. Yang et al. [2] later extented the CNN with an additional layer for Support Vector Machine (SVM) which could detect and classify traffic signs. They demonstrated excellent performance and reported classification accuracies of 98.24% and 98.77% for the GTSDB and CTSD datasets, respectively. In a different application of object classification, Ijjina and Chalavadi [3] proposed a method for recognizing human action. Instead of random initialization of the network they propose computing the initial weights of the CNN using genetic algorithms which minimize the classification error; using this method they can achieve classification accuracies of $99.98\%$and 96.92% for the UCF50 and MNIST datasets. Hu et al. [4] propose a Single Signal Crowd CNN Model for counting dense crowds, and report outstanding performance for the training on the UCSD dataset and testing on the UCF-CROWD dataset. Liang et al. [5] propose a recurrent CNN (R-CNN) for object recognition by incorporating recurrent connections into each convolutional layer. The R-CNN is shown to outperform the state-of-the-art models on the CIFAR-10, CIFAR-100, MNIST and SVHN datasets.

More recently [20] Fully-Convolutional Networks have been shown to produce the best results for multi-label pixelwise classification by training on overlapping patches, however a significant disadvantage is the fact that the produced output is considerably downsampled compare to the input and further processing is required. When dealing with semantic segmentation of fine structures such as the ones appearing in urban datasets this generates spurious results. Of similar performance but same shortcoming is the CRF-based approach proposed in [21] where again upsampling/interpolation is required on the generated output.

B. 3D reconstruction

The state-of-the-art in urban reconstruction can be better categorized according to the type and scale of the input data. For a comprehensive survey of urban reconstruction of various types and scales we refer the reader to the survey by Musialski et al [6]. In this section, we provide a brief overview of state-of-the-art in large-scale urban reconstruction from remote sensor data, most relevant to our work.

In [19], Zhou et al propose an automated system which given the exact bounding volume of a building can simplify the geometry based on dual contouring while retaining important features. Using this technique the authors were able to simplify the original geometry considerably. A different technique was presented in [18] where pointcloud data was converted automatically to polygonal 3D models. This technique was applicable directly on the raw pointcloud data without requiring any user interaction. Later, in [17] the authors extended the work to include a fast boundary refinement algorithm based on graph-cuts which was used to refine the boundaries and [24] for extracting appearance information.

On a similar line of research, Lafarge et al. [10] proposed a method which produces excellent reconstructed models from aerial LiDAR which can also handle the vegetation and complex grounds. Following a more interactive approach, Arikan et al [8] proposed a system for generating polyhedral models from semi-dense unstructured point-clouds. Planar surfaces were first extracted automatically based on prior semantic information, and later refined manually by an operator.

The Achilles' heel of almost all reported work in this area is the geospatial object classification: if an object is misclassified then subsequent steps will most definitely also go wrong. Furthermore, extracting buildings from LiDAR data often produces jagged boundaries which affects the accuracy and quality of the reconstruction. Hence, it is of imperative importance to have as accurate classification as possible at the pixel-level. The proposed neural network achieves this yielding average accuracy in the high ninety percentile for buildings.

III. System Overview

The training dataset is first converted to the input form expected by the network. Scale invariance is introduced by training the network on composite images containing multiple scales of the original depth map and RGB image captured by airborne LiDAR and satellite imaging. For training data, this also involves creation of multi-scale label data.

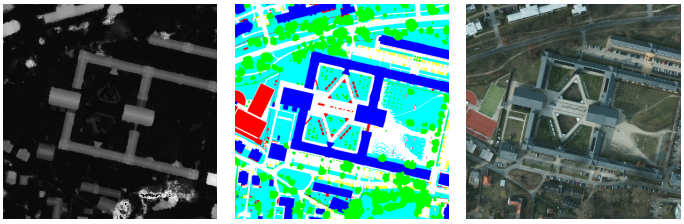
The overall size of each of these composite images is very large. Hence random samples of a fixed patch size taken from

the composite images are used for training the CNN. The CNN's output is six floating-point values per input pixel. These are interpreted as the likelihoods of the pixel to being classified with one of the six labels: building, road, tree, car, artificial ground, or natural ground. These likelihoods are used to train a linear classifier which outputs a single label per pixel. In a final step boundary labels are recovered and all labels are refined using graph-cuts for maximum a-posteriori (MAP) estimation with Markov Random Field (MRF) priors.

The training of the network (CNN, linear classifier) is performed on a large urban dataset described in the following Section IV. Once the network has been trained, data not used during the training is processed and labels are generated. The resulting labels are then used to extract only the buildings, cars and trees which are further processed to produce the 3D models representing the urban area.

IV. Dataset

The data used for the training and testing of the proposed network is provided by the International Society for Photogrammetry and Remote Sensing (ISPRS). The data is available as part of the benchmark on urban object detection and 3D building reconstruction [12] and consists of several datasets. In this work, we have used the Potsdam dataset for 2D semantic labeling [11] because of its higher accuracy. The Potsdam dataset consists of 24 image pairs consisting of three $6K \times 6K$ registered images, namely a depth map captured with airborne LiDAR with a sampling density of $5cms$, a color satellite image, and the ground truth label map. The label map shows the ground-truth per-pixel classification into six classes: buildings (blue), trees (green), roads (white), natural ground (cyan), artificial ground (red) and cars (yellow). The 'artificial ground' label contains all areas on the ground that do not correspond to roads but are covered by materials such as asphalt that are typically used for paving roads. In particular, it contains parking lots, pavements, inner courtyards and driveways (if paved). The 'natural ground' label contains any areas on the ground covered by vegetation other than trees. In particular, it contains lawn and low bushes. The remaining labels are self-explanatory. Figure 1 shows a sample pair available in the Potsdam dataset.



(a) Depth map; contains values ranging from [0,1].

(b) Label map; contains one of six values corresponding to the geospatial feature classes.

(c) Color image; contains an RGB color where each channel ranges from [0,1].

Fig. 1: The Potsdam dataset consists of 24 image pairs. Each pair consists of a $6K \times 6K$ (a) depth map, (b) label map and (c) color image.

V. Network Architecture

The proposed deep neural network consists of a 13-layer Convolutional Neural Network (CNN) and a linear classifier (SVM). A diagram of the network's architecture is shown in Figure 2.

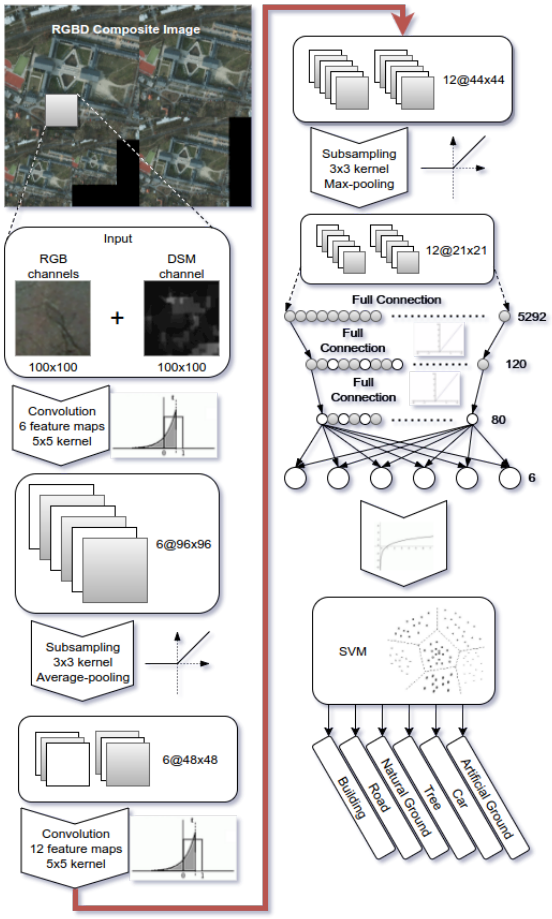


Fig. 2: The proposed deep neural network for geospatial object classification of remote sensor data. A 13-layer CNN followed by a multi-class SVM can perform multi-label pixelwise classification into one of six labels: buildings, roads, trees, cars, ground [natural, artificial].

The input to the network are RGBD values corresponding to pixels contained in a small patch $N \times N$ of the input image. The optimal size of the patch $N \times N$ is determined empirically by varying the size while assessing the performance. Our experiments [the most relevant of which are shown in Figure 9] have shown that the value resulting in optimal performance is $N = 100$. The CNN's filter's kernel size k was also determined in a similar fashion and is set to $k = 5$. An extensive empirical study showing the performance of the CNN with respect to different combinations of 3 patch sizes $[34, 70, 100]$ and 7 kernel sizes $[5, 7, 9, 11, 13, 15, 17]$ was performed. Table I shows the optimal performance achieved by the CNN for which the hyperparameters are set to patch size of 100×100 and kernel size of 5×5.

Based on the above, a patch of pixels with size 100×100 and four channels per pixel (RGBD) becomes the input to the

Urban Area	Ref.	Bldgs (%)	Artif. Gnd.(%)	Trees(%)	Nat. Gnd.(%)	Roads(%)	Cars(%)	Acc. (%)
P3	w/o	93.32	48.04	68.21	83.80	84.93	72.35	81.99
	w	93.44	48.20	68.79	84.16	85.23	74.14	82.34
P4	w/o	94.19	19.99	68.57	74.74	79.02	70.60	79.55
	w	94.41	20.51	69.36	75.14	79.33	72.58	79.99
P6	w/o	95.05	72.43	60.53	58.05	84.08	74.09	83.48
	w	95.30	73.16	61.67	58.41	84.34	75.94	83.88
P7	w/o	94.10	52.80	50.61	70.46	89.58	76.28	84.78
	w	94.26	54.40	50.82	70.65	89.73	77.73	85.01

TABLE I: Optimal network performance. The hyperparameters were empirically derived: patch size is 100×100 and kernel size is 5×5. The table also shows a comparison between before(w/o) and after(w) applying the label refinement process.

network for all reported results. Similarly a kernel size of 5×5 is used for all spatial convolutions.

The first spatial convolution layer maps the input image patch into 6 feature maps [of size 96×96]. The following sub-sampling layer samples the output image of the previous layer with 3×3 kernel and average pooling, and generates 6 images [of size 48×48] output. The next layer of the network applies a spatial convolution and maps the 6 input images to 12 output images [of size 44×44]. Then these images are sub-sampled with 3×3 kernel max pooling, and 12 (21×21) mapping images are generated. All the pixels of the resulting 12 images are fully connected to a linear layer with 5292 nodes i.e. $12 \times 21 \times 21$. Next, the 5292-node linear layer passes through two fully connected linear layers of 120-nodes and 80-nodes respectively. The final linear layer is fully connected with the previous and consists of 6 nodes corresponding to the six labels. Each convolutional operation is followed by the non-linear operation $ReLU(x) = max(x, 0)$. Thus, the CNN models the following operation,

$$\Gamma(\Gamma(\Gamma(\Pi_{max}(ReLU(\Psi \star \Pi_{avg}(ReLU(\Psi \star X)))))))) \mapsto \Phi^p \tag{1}$$

where p is a pixel and Φ^p is a 6-tuple of values corresponding to the six labels. In the above equation X denotes the input data, Ψ denotes a convolution kernel, $\Gamma(.)$ maps the input to a fully connected linear layer, $\Pi_{max}(.)$ is the max-pooling operation, $\Pi_{avg}(.)$ is the average-pooling operation, and $ReLU(.)$ is the rectified linear unit function.

As previously mentioned, the output of the CNN network is a 6-tuple Φ of values for each pixel p contained in the input patch

$$\Phi^p =< \phi_1^p, ..., \phi_6^p > \tag{2}$$

where each component in Φ^p is interpreted as the *unnormalized* likelihood of the pixel p to be classified with any one of the six labels. $\phi_1^p ... \phi_6^p$ represent the unnormalized probabilities of building, tree, road, artificial ground, natural ground and car respectively. We define Λ^p as the 6-tuple of normalized likelihoods given by,

$$\Lambda^p = \frac{e^{\Phi^p}}{\omega} =< \lambda_1^p, ..., \lambda_6^p >=< \frac{e^{\lambda_1^p}}{\omega}, ..., \frac{e^{\lambda_6^p}}{\omega} > \tag{3}$$

where $\omega = \sum_{i=1}^{6} e^{\phi_i^p}$ such that $\sum_{i=0}^{6} \lambda_i^p = 1$.

An example of the output is shown in Figure 3. The components of the per-pixel likelihoods Λ^p are grouped according to the labels and are shown as six images. The range of values of each individual component of Λ is [0,1].

As it is evident from Figure 3, the output at this point is a tuple Λ for each pixel in the input *composite* image. This means that for each pixel in the original [non-composite] image there will be essentially five sets of likelihoods within each composite image; one for each scale. The five sets of likelihoods Λ_i with $1 \leq i \leq 5$ corresponding to each pixel are combined into a single tuple $\bar{\Lambda}$ by first up-scaling the images to the original $6k \times 6k$ resolution and then averaging the per-pixel likelihoods. Pixels lying on the boundaries for which not every scale may output a likelihood are not assigned likelihoods.

The resulting $6k \times 6k$ set of normalized likelihoods $\bar{\Lambda}^p$ corresponding to each pixel p and the original label map with the same resolution $6k \times 6k$ becomes the input to a linear classifier (SVM). After training, the SVM learns weights W and bias b of the mapping function $f(W \times \bar{\Lambda}^p + b) \mapsto l_i$ where $l_i, 1 \leq i \leq 6$ indicates one of the six labels. Figure 3g shows the result of this process on the likelihoods corresponding to the label 'building'. Figure 3h shows the final output of the SVM and Figure 3i shows the ground truth for the labels. It should be noted that at this point, boundary pixels cannot be assigned a label.

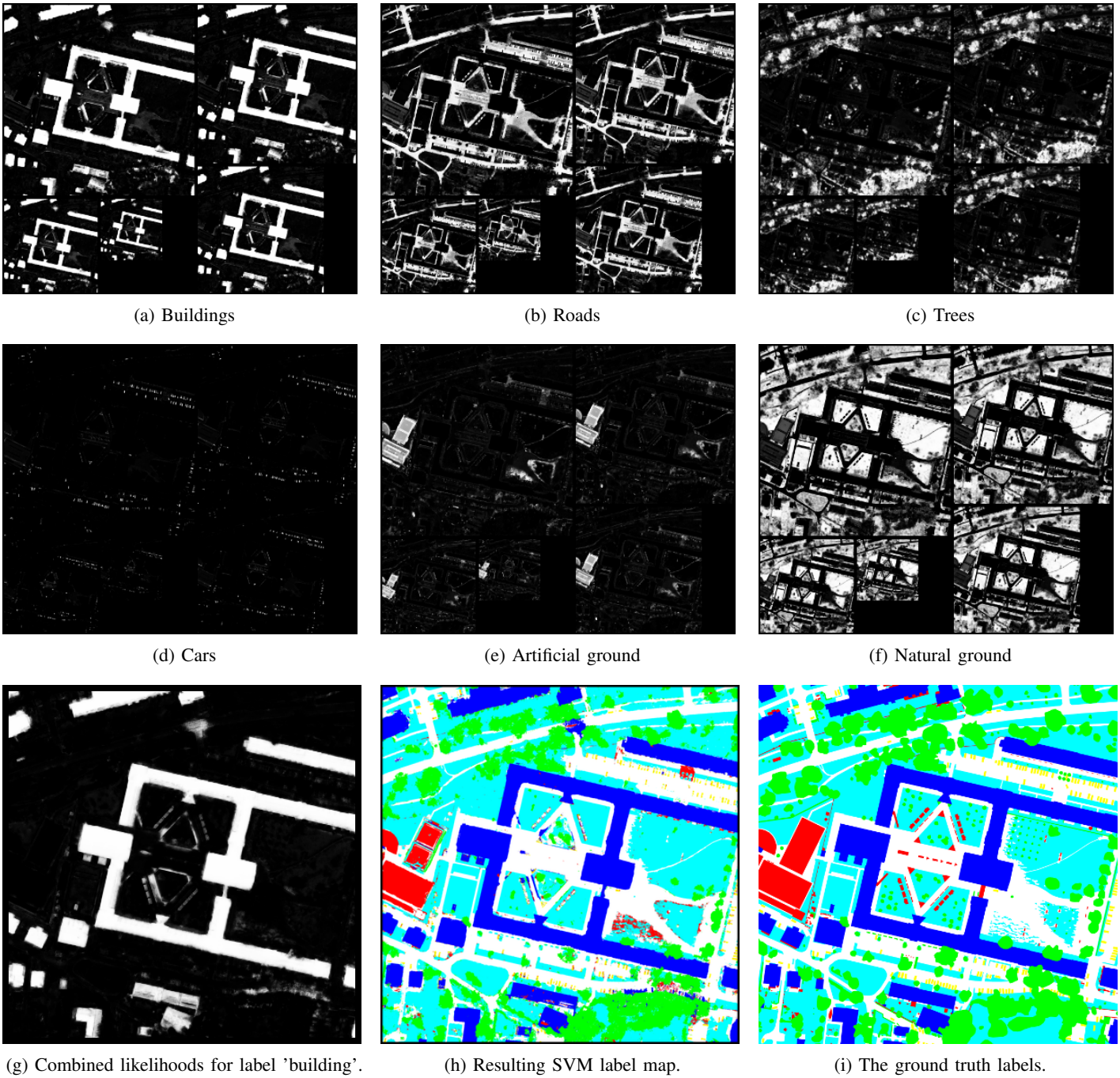
A. Training

The Potsdam dataset consists of 24 pairs of images. The training is performed on 20 randomly selected image pairs and validated against the remaining 4 image pairs. Inspired by DenseNet [13], we incorporate scale invariance into the training by preprocessing the original data to create composite images containing multiple resolutions of the original. These composite image pairs [depth, RGB, labels] become the input to the network. A composite contains the five images I_i where $0 \leq i \leq 4$ each with resolution corresponding to $i \times 16\%$ decrements of the original resolution i.e. $6k \times 6k, 5k \times 5k, 4k \times 4k, 3k \times 3k, 2k \times 2k$. The dataset contains considerable variance in the orientations of the geospatial objects hence rotation invariance is implicitly incorporated in the training.

The CNN was trained for 300 epochs on a single machine with the following specifications: Intel Core i7-6700K CPU @ 4.00GHz 8, 16GB RAM, 12GB NVidia GeForce GTX TITAN X/PCIe/SSE2. The Torch API was used for the development of the CNN and the code will be made available as open source. The duration of the training for 300 epochs was 26 hours however, as it can be seen from Figure 4 after the first few epochs the training error rapidly reduces and almost converges.

Although the available memory on the GPU is 12GB, the Torch API imposes a restriction on the maximum GPU usage to 2GB. Hence, the available training data cannot be used in its entirety. Instead, given the 20 training image pairs we perform random sampling and gather as many training samples [100×100 image patches] as the memory can fit. The uniform random sampling from the 20 image pairs includes patches from various resolutions. We ensure that all pixels within each sampled image patch fall entirely within the same scale. Sampled patches falling on boundaries between different scales are rejected. This results in a total of $400,000$ training samples.

Finally, the likelihoods generated by the CNN are combined as previously described and fed into multi-class SVM,



(a) Buildings (b) Roads (c) Trees

(d) Cars (e) Artificial ground (f) Natural ground

(g) Combined likelihoods for label 'building'. (h) Resulting SVM label map. (i) The ground truth labels.

Fig. 3: (a)-(f): The per-pixel class likelihoods; intuitively, the brighter the value of a pixel in a class' image, the higher the likelihood of the pixel to be classified with that class. Bottom row: (g) The combined likelihoods for label 'building' which is used as part of the input to the SVM, (h) the per-pixel label classification map resulting from SVM; note that boundary pixels are not assigned a label at this point, (i) the ground truth labels.

which generates the pixel's classification after using a one-vs-all learning strategy. The SVM learns how to map the 6-tuple $\bar{\Lambda}^p$ corresponding to each pixel p into a single class which classifies the input with the highest margin. The Torch API was again used for the development of the SVM and the code will be made available as open source. The duration of the training was 27 minutes and was performed on the same machine as above.

B. Maximum-a-posteriori Inference with Markov Random Field Priors for Label Refinement

The linear classifier combines the six pixel-wise likelihoods produced by the CNN into a single label. Pixels along the boundaries of the image cannot be assigned a label because

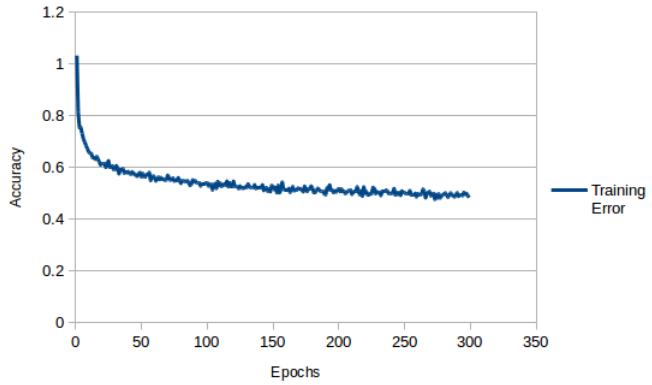


Fig. 4: Training error of 300 iterations

the convolutional filter falls out of bounds. To overcome this problem Overfeat [23] first introduced the shift-and-stitch trick where shifted versions of the input were processed and the results interleaved into a full resolution output. However, the computational efficiency of this approach does not scale to the current large-scale urban datasets we are dealing with.

Instead, we propose the use of graph-cuts for maximum-a-posteriori (MAP) estimation with Markov Random Field (MRF) priors. We reformulate the problem as finding an optimal labeling f_p for every pixel p such that $f(p) \mapsto l$ where l is a new label. In addition to the six labels we include a new label *unknown* to account for the boundary pixels which have not been assigned a label. Hence, the set of labels becomes *[buildings, roads, trees, cars, natural ground, artificial ground, unknown]*.

The energy function which is minimized is then given by,

$$E(f) = E_{unary}(f) + E_{pairwise}(f) \tag{4}$$

The unary energy term $E_{unary}(f)$ provides a measure of the compatibility of the new label under the labeling $f(p_i)$ to the pixel p_i with label l_{p_i} in the observed data and is given by,

$$E_{unary}(f) = \sum_{i=0}^{N} \begin{cases} 10, & \text{if } f(p_i) \neq l_{p_i}. \\ 15, & \text{if } f(p_i) = unknown. \\ 0, & \text{if } f(p_i) = l_{p_i} \end{cases} \tag{5}$$

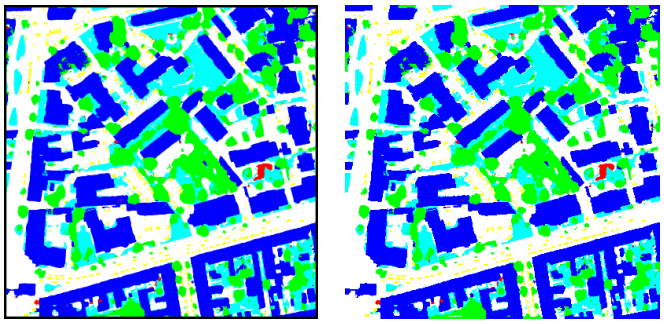
The pairwise energy term $E_{pairwise}(f)$ provides a measure of compatibility of the new labels under the labeling $f(p_i), f(p_j)$ for neighbouring pixels p_i and p_j respectively and is given by,

$$E_{pairwise}(f) = \sum_{i,j=0}^{N} \begin{cases} 0, & \text{if } l_{p_i} = l_{p_j}. \\ 20, & \text{otherwise.} \end{cases} \tag{6}$$

The pairwise measure $V(f_{p_i}, f_{p_j})$ between neighbouring pixels p_i, p_j, and p_k can be trivially shown to be metric since the following conditions are true,

$$\begin{aligned} V(f(p_i), f(p_j)) = 0 \leftrightarrow i = j \\ V(f(p_i), f(p_j)) = V(f(p_j), f(p_i)) > 0 \\ V(f(p_i), f(p_j)) \leq V(f(p_i), f(p_k)) + V(f(p_k), f(p_j)) \end{aligned} \tag{7}$$

This is a multi-label MRF problem with non-submodular energy potentials and as such can only be approximately solved. The alpha-expansion algorithm is used to break the multi-label problem into a series of binary problems. Experiments have shown that after 5 iterations the energy $E(f)$ is reduced on average 12% and the overall accuracy of the classification results increases by $[0.5 - 1\%]$. Table I shows a comparison between the performance before(w/o) and after(w) this process. Image boundary pixels for which no label was generated are now relabeled based on the new labeling resulting from the energy minimization using graph-cuts. Figure 5 shows an example output of this process.



(a) (b)

Fig. 5: Maximum-a-posteriori Inference with Markov Random Field Priors for Label Refinement. (a) The labels generated by the network. Pixels along the image boundaries cannot be assigned a label. (b) Dense and refined labeling resulting from the proposed method. Pixels along the image boundary are assigned a label.

C. Validation

The proposed network was validated against two sets of test images: (a) the 4 image pairs out of the 24 available which were not used in the training and (b) the 14 image pairs available for testing for which ISPRS did not make the ground truth publicly available. The performance is measured in terms of Precision (P), Recall (R), and F_1 score which are defined as,

$$P = \frac{tp}{tp + fp} \qquad R = \frac{tp}{tp + fn} \qquad F_1 = 2 \times \frac{P \times R}{P + R} \tag{8}$$

where tp indicates the true positives, fp indicates the false positives, and fn indicates the false negatives.

1) Classification Results for 4 image pairs: Out of the 24 available image pairs, 20 were used for the training and the remaining 4 were used for validation. The network performance statistics for the 4 validation images were presented in Table I. These statistics were computed from the ground truth labels made available with the Potsdam dataset.

2) Classification Results for 14 image pairs: The ISPRS benchmark also contains 14 image pairs for which ground truth was not made publicly available. The following network performance statistics were computed by and provided by the ISPRS Working Group II/4 organizers as part of their urban classification benchmark. Table II shows the evaluation of the overall classification results for the 14 test images and as it

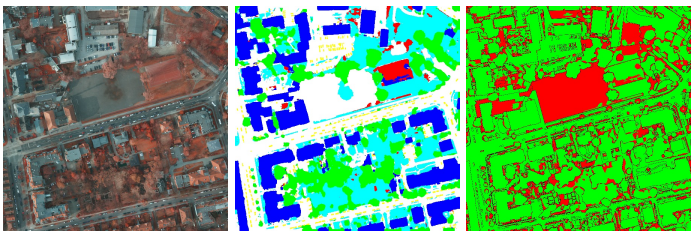
pred./ref.	roads	bldgs	nat. gnd.	tree	car	artif. gnd.
roads	89.9	1.6	4.9	2.9	0.1	0.6
bldgs	2.5	94.40	0.8	2	0	0.3
nat. gnd	6.8	0.90	82.40	9.1	0	0.7
tree	7.6	1	20.3	70.6	0.4	0.1
car	17.60	1.6	0.8	1.80	76.7	1.7
artif. gnd.	38.7	8.3	28.60	4.8	2	17.60
Prec./Corr.	84.40	95	72.5	77.4	86.3	60.5
Rec./Compl.	89.9	94.40	82.40	70.6	76.7	17.60
F1	87.1	94.70	77.10	73.9	81.2	27.3

TABLE II: The overall evaluation of the classification results for the 14 test images for which ground truth was not provided. The network performance statistics were computed by and provided by the ISPRS Working Group II/4 organizers as part of their urban classification benchmark. All shown values are percentages.

pred./ref.	roads	bldgs	nat. gnd.	tree	car	artif. gnd.
roads	89.44	1.17	6.04	1.82	0.11	1.42
bldgs	1.80	96.61	0.38	0.87	0.09	0.24
nat. gnd.	17.64	1.10	70.3	9.82	0.04	1.11
tree	9.72	1.18	16.73	71.75	0.4	0.22
car	18.51	1.7	0.48	1.69	76.02	1.59
artif. gnd.	64.15	6.8	11.96	2.24	0.49	14.36
Prec./Corr.	63.5	93.2	72.5	81.5	88.7	66.5
Rec./Compl.	89.4	96.6	70.3	71.7	76	14.40
F1	74.2	94.90	71.40	76.3	81.90	23.60

TABLE III: Evaluation results for urban area P4-14. All shown values are percentages.

can be seen the overall accuracy for building classification is almost 95%. Figure 6 shows the evaluation results for one of the 14 test images, namely P4-14. The individual evaluation results for P4-14 are shown in Table III.

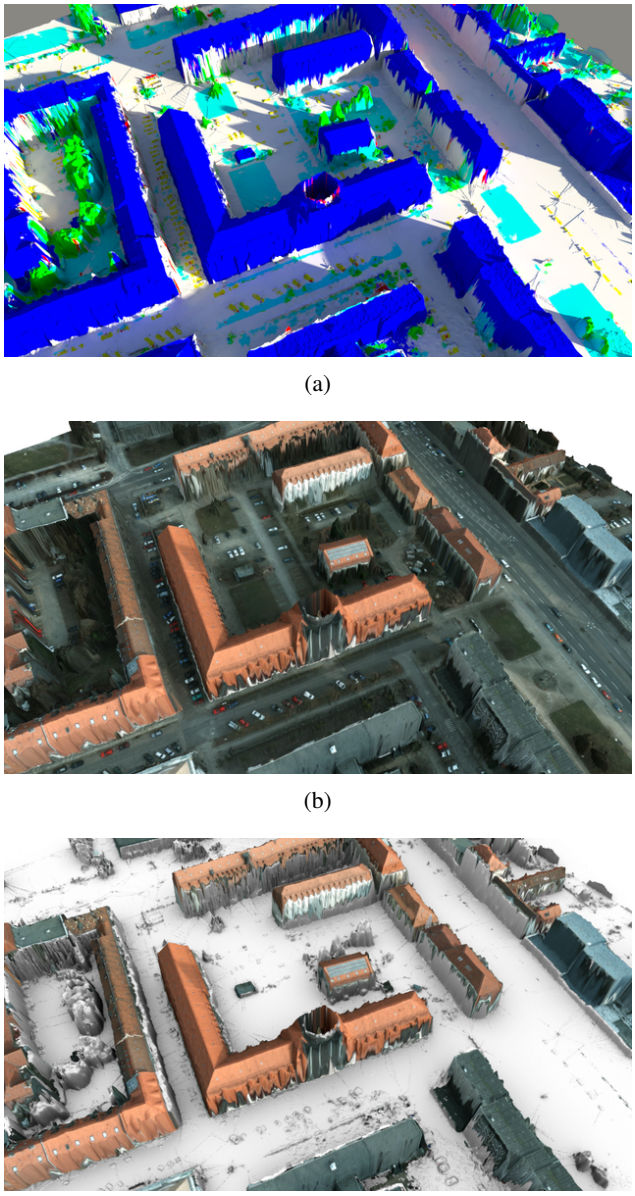


(a) (b) (c)

Fig. 6: The evaluation result for one of the 14 test images. Resolution $6k \times 6k$. (a) Satellite image of the urban area P4-14. (b) Generated label map. (c) Red/green image, indicating wrongly classified pixels.

VI. Urban Reconstruction

The classification results are further processed. The depth map is used to extract boundary points and extrude 3D models to represent the geospatial objects in the scene. In particular the boundaries for the buildings are extracted and extruded to create polygonal models and generic objects such as cars are replaced by CAD models, and trees are replaced by procedural models. Figure 7(a)-(b) shows the generated labels being projected onto the same 3D models and Figure 7(c)-(d) shows textured models only for the classified buildings. Figure 8a shows a closeup of an urban area in which buildings have been replaced by polygonal models, cars by generic CAD models and trees by procedural models. The same scene with textures and from a different viewpoint is shown in Figure 8b.



(a)

(b)

(c)

Fig. 7: (a) The resulting labels being projected onto the 3D models. (b) The satellite image projected onto the scene models. (c) Textured models showing only the classified buildings.

VII. Conclusion

We have presented a novel technique for multi-label pixel-wide classification for reconstruction of large-scale urban areas. Unlike existing methods, the proposed method relies on a relatively small CNN to efficiently process large sets of data. An empirical study was performed and presented to determine the parameters for which the network produces optimal results. Scale invariance is incorporated in the processing by training the network on composite images containing multiple scales of the originals. This results in multiple per-pixel classification

(a)

(b)

Fig. 8: Reconstructed and textured models. Buildings are replaced by polygonal models, cars by CAD models, and trees by procedural models. A mesh resulting from triangulating the depth map is used for the other classes: roads, natural and artificial ground.

labels which are mapped into a single label using a trained linear classifier. Pixels lying on image boundaries are not assigned any label. We reformulate the problem as a labeling problem and propose the use of graph-cuts for maximum-a-posteriori inference of those labels with Markov Random Field priors. The result is a dense set of labels where all pixels in the image are assigned a label according to the minimized energy function. The proposed technique has been extensively tested on large-scale datasets depicting urban areas for which ground truth is available. The achieved accuracy in the classification ranges in the 90th percentile.

ACKNOWLEDGMENT

The authors would like to thank the International Society for Photogrammetry and Remote Sensing and in particular the ISPRS Working Group II/4 organizers and Dr. Markus Gerke for creating the benchmark and making it publicly available. We also thank Microsoft Research for making processing large datasets possible through its Azure for Research Award. This research is based upon work supported by NSERC DG under Grant no N01670.

REFERENCES

[1] LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural Comput. 1(4), 541551, 1989.

[2] Y. Yang, H. Luo, H. Xu, and F. Wu, Towards Real-Time Traffic Sign Detection and Classification, IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 7, pp. 20222031, Jul. 2016.

[3] E. P. Ijjina and K. M. Chalavadi, Human action recognition using genetic algorithms and convolutional neural networks, Pattern Recognition, vol. 59, pp. 199212, Nov. 2016.

[4] Y. Hu, H. Chang, F. Nian, Y. Wang, and T. Li, Dense crowd counting from still images with convolutional neural networks, Journal of Visual Communication and Image Representation, vol. 38, pp. 530, Jul. 2016.

[5] M. Liang and X. Hu, Recurrent Convolutional Neural Network for Object Recognition, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 33673375.

[6] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. van Gool, and W. Purgathofer, A Survey of Urban Reconstruction, Computer Graphics Forum, vol. 32, no. 6, pp. 146177, Sep. 2013.

[7] Wu C., A GARWAL S., C URLESS B., S EITZ S. M.: Multicore bundle adjustment. In 2011 IEEE Conference on Computer Vision and Pattern Recognition (Colorado Springs, CO, USA, 2011), IEEE, pp. 30573064.

[8] A LSISAN S., M ITRA N. J.: Variation-factored encoding of facade images. In EUROGRAPHICS 2012 Short Paper Proceedings (Cagliari, Italy, 2012), pp. 3740.

[9] L IU J., M USIALSKI P., W ONKA P., Y E J.: Tensor completion for estimating missing values in visual data. IEEE Transactions on Pattern Analysis and Machine Intelligence 35, 1 (January 2013), 208220.

[10] L AFARGE F., M ALLET C.: Building large urban environments from unstructured point data. In 2011 International Conference on Computer Vision (Barcelona, Spain, November 2011), IEEE, pp. 10681075.

[11] ISPRS WG III/4. ISPRS 2D Semantic Labeling Contest. http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html

[12] Rottensteiner, F., Sohn, G., Gerke, M., Wegner, J. D., Breitkopf, U., & Jung, J. (2014). Results of the ISPRS benchmark on urban object detection and 3D building reconstruction. ISPRS Journal of Photogrammetry and Remote Sensing, 93, 256-271.

[13] Iandola, F., Moskewicz, M., Karayev, S., Girshick, R., Darrell, T., & Keutzer, K. (2014). Densenet: Implementing efficient convnet descriptor pyramids. arXiv preprint arXiv:1404.1869.

[14] Ouyang, W., Wang, X., Zhang, C., & Yang, X. (2016). Factors in Finetuning Deep Model for Object Detection with Long-tail Distribution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 864-873).

[15] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (pp. 91-99).

[16] Yang, H., Zhou, J. T., Zhang, Y., Gao, B. B., Wu, J., & Cai, J. Exploit Bounding Box Annotations for Multi-label Object Recognition.

[17] Poullis, C. (2013). A framework for automatic modeling from point cloud data. IEEE transactions on pattern analysis and machine intelligence, 35(11), 2563-2575.

[18] Poullis, C., & You, S. (2009, June). Automatic reconstruction of cities from remote sensor data. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on (pp. 2775-2782). IEEE.

[19] Zhou, Q. Y., & Neumann, U. (2012, June). 2.5 D building modeling by discovering global regularities. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on (pp. 326-333). IEEE.

[20] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3431-3440).

[21] Lin, G., Shen, C., & Reid, I. (2015). Efficient piecewise training of deep structured models for semantic segmentation. arXiv preprint arXiv:1504.01013.

[22] Boykov, Y., Veksler, O., & Zabih, R. (2001). Fast approximate energy minimization via graph cuts. IEEE Transactions on pattern analysis and machine intelligence, 23(11), 1222-1239.

[23] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229.

[24] Poullis, Charalambos and You, Suya (2011). 3d reconstruction of urban areas. International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT).

urban area #	kernel size k	patch size N	buildings	artificial ground	trees	natural ground	roads	cars	accuracy
3_11	5x5	100x100	94.44%	45.59%	77.93%	85.02%	86.97%	80.36%	**84.60%**
		70x70	93.49%	47.73%	71.92%	84.23%	85.72%	72.96%	83.00%
		34x34	93.59%	52.49%	64.50%	82.38%	87.05%	71.72%	82.05%
	7x7	100x100	93.11%	44.47%	70.52%	79.80%	82.18%	76.03%	80.20%
		70x70	92.84%	52.95%	64.93%	84.08%	82.41%	71.91%	80.99%
		34x34	92.60%	44.47%	62.10%	81.61%	85.44%	64.27%	80.28%
	9x9	100x100	92.17%	44.03%	73.81%	82.97%	86.18%	76.96%	82.50%
		70x70	94.09%	49.12%	66.88%	83.20%	86.59%	62.71%	82.45%
		34x34	91.50%	51.28%	61.35%	79.54%	85.28%	46.64%	79.37%
	11x11	100x100	93.09%	38.32%	68.50%	82.51%	83.37%	70.16%	80.79%
		70x70	94.35%	42.99%	70.06%	83.26%	85.19%	72.46%	82.12%
		34x34							
	13x13	100x100	93.44%	12.07%	71.21%	83.62%	82.94%	54.14%	80.87%
		70x70	93.32%	48.04%	68.21%	83.80%	84.93%	72.35%	81.99%
		34x34							
	15x15	100x100	92.49%	53.29%	70.96%	83.95%	86.07%	76.72%	82.88%
		70x70	93.85%	43.24%	63.86%	83.12%	84.04%	65.57%	81.04%
		34x34							
	17x17	100x100	91.90%	65.31%	63.42%	81.71%	83.49%	58.83%	80.62%
		70x70	93.10%	43.99%	60.32%	80.42%	83.15%	61.49%	79.31%
		34x34							
4_11	5x5	100x100	94.66%	24.34%	76.68%	78.17%	82.99%	82.46%	**83.12%**
		70x70	93.07%	16.45%	71.31%	76.74%	81.43%	74.68%	80.74%
		34x34	94.70%	11.83%	67.71%	73.40%	78.90%	73.97%	79.39%
	7x7	100x100	93.23%	13.27%	70.15%	74.53%	79.86%	75.87%	79.99%
		70x70	94.48%	22.10%	66.76%	73.87%	78.90%	74.52%	79.36%
		34x34	94.45%	13.67%	64.64%	74.45%	80.87%	67.36%	79.44%
	9x9	100x100	94.38%	14.64%	73.71%	74.01%	79.49%	76.49%	80.62%
		70x70	93.75%	9.13%	68.56%	74.93%	79.40%	64.40%	79.68%
		34x34	92.50%	5.68%	62.40%	71.33%	78.81%	47.58%	77.09%
	11x11	100x100	94.12%	16.74%	70.02%	75.25%	80.68%	73.19%	80.39%
		70x70	94.18%	21.28%	70.80%	75.95%	80.46%	73.74%	80.76%
		34x34							
	13x13	100x100	93.77%	18.97%	71.42%	76.67%	80.25%	62.05%	80.86%
		70x70	94.19%	19.99%	68.57%	74.74%	79.02%	70.60%	79.55%
		34x34							
	15x15	100x100	93.11%	11.69%	70.33%	74.96%	79.34%	74.44%	79.86%
		70x70	94.41%	19.93%	65.63%	73.12%	78.57%	69.72%	78.76%
		34x34							
	17x17	100x100	93.69%	17.98%	65.22%	69.06%	74.15%	59.29%	76.28%
		70x70	93.21%	9.86%	62.76%	74.19%	79.57%	64.25%	78.49%
		34x34							
6_11	5x5	100x100	96.20%	70.98%	74.24%	66.23%	87.89%	84.09%	**87.63%**
		70x70	94.36%	68.78%	65.13%	60.87%	86.78%	79.39%	84.68%
		34x34	95.45%	69.83%	55.16%	55.57%	84.60%	73.14%	82.96%
	7x7	100x100	95.21%	67.63%	64.10%	63.35%	87.41%	80.34%	86.49%
		70x70	95.74%	74.75%	63.95%	59.34%	85.35%	79.63%	85.13%
		34x34	93.93%	64.23%	52.29%	57.73%	85.33%	70.33%	82.24%
	9x9	100x100	95.07%	63.49%	69.88%	57.82%	84.83%	79.23%	84.33%
		70x70	95.03%	76.25%	60.82%	57.92%	85.25%	68.47%	84.09%
		34x34	93.79%	67.76%	52.91%	55.20%	84.34%	53.89%	82.05%
	11x11	100x100	95.61%	62.93%	63.45%	58.02%	85.87%	75.09%	84.44%
		70x70	95.39%	72.63%	62.67%	60.11%	85.51%	76.40%	84.81%
		34x34							
	13x13	100x100	93.86%	16.34%	65.34%	60.99%	83.53%	65.37%	82.46%
		70x70	95.05%	72.43%	60.53%	58.05%	84.08%	74.09%	83.48%
		34x34							
	15x15	100x100	95.43%	75.93%	67.23%	59.41%	85.14%	77.60%	85.28%
		70x70	95.39%	38.33%	56.10%	54.71%	81.73%	73.44%	81.21%
		34x34							
	17x17	100x100	95.48%	61.08%	57.00%	49.06%	80.42%	57.16%	80.45%
		70x70	94.72%	68.51%	51.61%	58.17%	83.96%	67.10%	82.73%
		34x34							
7_11	5x5	100x100	95.43%	42.33%	64.84%	75.74%	91.37%	84.00%	**87.64%**
		70x70	94.95%	31.03%	53.18%	72.01%	90.64%	78.97%	85.36%
		34x34	94.27%	34.88%	48.22%	67.90%	88.36%	77.36%	83.22%
	7x7	100x100	94.01%	30.42%	52.43%	70.77%	90.15%	80.80%	85.51%
		70x70	94.22%	39.37%	51.97%	71.51%	89.42%	78.84%	84.79%
		34x34	94.48%	28.62%	46.24%	70.33%	89.02%	74.84%	83.61%
	9x9	100x100	95.06%	34.77%	59.18%	70.93%	90.15%	80.37%	85.78%
		70x70	94.50%	43.45%	49.46%	69.06%	88.88%	68.16%	83.95%
		34x34	92.52%	36.35%	49.04%	69.39%	88.56%	52.60%	82.88%
	11x11	100x100	95.19%	26.40%	51.84%	70.25%	89.97%	75.42%	84.98%
		70x70	93.91%	42.79%	55.62%	72.37%	89.91%	77.30%	85.37%
		34x34							
	13x13	100x100	93.26%	8.91%	52.88%	72.73%	89.14%	54.60%	84.03%
		70x70	94.10%	52.80%	50.61%	70.46%	89.58%	76.28%	84.78%
		34x34							
	15x15	100x100	93.92%	42.88%	58.91%	71.38%	90.13%	78.79%	85.70%
		70x70	94.97%	17.08%	47.17%	67.92%	88.64%	73.69%	83.53%
		34x34							
	17x17	100x100	93.59%	25.22%	49.74%	62.30%	85.85%	59.00%	80.53%
		70x70	93.55%	25.46%	47.92%	70.77%	88.89%	68.30%	83.51%
		34x34							

Fig. 9: Gray cells: Kernel vs patch size is too small. The resulting image size after the two convolutional layers is too small and can not be processed further without changing the network architecture.