Semantic Labeling for improved Vehicle Detection in Aerial Imagery

Lars Sommer\textsuperscript{2,1} Kun Nie\textsuperscript{1} Arne Schumann\textsuperscript{1} Tobias Schuchert\textsuperscript{1} Jürgen Beyerer\textsuperscript{1,2}

\textsuperscript{1}Fraunhofer IOSB
Fraunhoferstrasse 1
76131 Karlsruhe, Germany

\textsuperscript{2}Vision and Fusion Lab
Karlsruhe Institute of Technology KIT
Adenauerring 4, 76131 Karlsruhe, Germany

firstname.lastname@iosb.fraunhofer.de

Abstract

Growing cities and increasing traffic densities result in an increased demand for applications such as traffic monitoring, traffic analysis, and support of rescue work. These applications share the need for accurate detection of relevant vehicles, e.g. in aerial imagery. Recently, the application of deep learning based detection frameworks like Faster R-CNN clearly outperformed conventional detection methods for vehicle detection in aerial images. In this paper, we propose a detection framework that fuses Faster R-CNN and semantic labeling to integrate contextual information. We achieve an improved detection performance by decreasing the number of false positive detections while the number of candidate regions to classify is reduced. To demonstrate the generalization of our approach, we evaluate our detection framework for various ground sampling distances on a publicly available dataset.

1. Introduction

With growing cities and increasing traffic densities, the demand for applications like traffic monitoring, traffic analysis, and support of rescue work has strongly increased. For this, images or videos captured by surveillance cameras or airborne systems are commonly used. To reduce the work load of image analysts, automatic detection systems are required.

In this work, we focus on vehicle detection in aerial imagery, which provides information of large areas at once. Recently, detection frameworks based on convolutional neural networks (CNNs) clearly outperformed conventional detection methods for vehicle detection in aerial imagery [19, 22]. In case of low resolution imagery, the best detection performance is achieved for small network architectures. However, the lack of contextual information due to the small network size results in a high number of false alarms caused by objects with shapes similar to vehicles, e.g. windows or solar panels on buildings [22].

To address this issue, we propose a detection framework that fuses semantic label information with the established CNN object detection pipeline based on region proposals (e.g. [17]). An overview is depicted in Figure 1. This gives us two main points at which to include the segmentation information: after the proposal stage or after the final classification step. Our main contribution lies in demonstrating the usefulness of incorporating semantic segmentation information to decrease the number of false alarms while the number of region proposals to classify is reduced. We show the generalization of our approach by evaluating our detection framework for various ground sampling distances (GSDs) on the publicly available ISPRS 2D Semantic Labelling Challenge Potsdam dataset [2].

2. Related Work

Many recent works focus on semantic segmentation with deep learning methods [9]. We will limit our discussion to such works that perform semantic segmentation in aerial imagery. A popular type of deep net for semantic segmentation are fully convolutional networks (FCNs) [12]. Sherrah [20] uses FCNs with a no- downsampling architecture for semantic segmentation of aerial imagery with the help of infrared and depth information. Marmanis et al. [15] use FCNs as well and train an ensemble of CNNs. In [16], Paisitkriangkrai et al. propose a combination of hand-crafted and CNN features for semantic segmentation of aerial images with conditional random fields as a post-processing stage. Maggiori et al. [13] learn features at multiple resolutions and combine them to leverage local and global information for the segmentation. Volpi et al. [23] train an encoder-decoder architecture by use of convolutional and deconvolutional layers. Lio et al. [11] also use encoder-decoder architecture and integrate inception layers to provide richer context through the varying receptive field.
Figure 1. Overview of our approach. The semantic segmentation is used to filter out unlikely proposals which, for example, contain only pixels segmented as vegetation (teal) or building (blue). Proposals with car pixels (yellow) or impervious surface, e.g. street, pixels (white) are relevant and can be classified. Alternatively, the fusion can be applied at a later stage on the classified detections.

Several detection methods have been proposed to detect vehicles in aerial imagery [6]. In general, these methods consist of a combination of hand-crafted descriptors and a classifier within a sliding window approach. Recently, these methods were outperformed by applying detection frameworks based on CNNs [19, 22]. In [22], the authors systematically investigate the applicability of Fast R-CNN and Faster R-CNN on two publicly available aerial imagery datasets. In [19], the authors propose Faster R-CNN for vehicle detection in multimodal imagery. Audebert et al. [4] propose an alternative detection method based on initial semantic labeling. They extract connected components labeled as car followed by CNN-based vehicle classification. To improve object detection, several authors recently combine detection frameworks and semantic labeling [7, 5]. Du et al. [7] reduce the number of false alarms in case of pedestrian detection by removing all detections with less than 20% pixels assigned to class pedestrian. In [5], the authors apply multi-task learning and multi-feature classification by sharing the convolutional layers of the detection and semantic labeling network.

3. Method

In this section we first describe the proposed segmentation net, then the main detection pipeline, and finally our proposed fusion methods.

3.1. Semantic Labeling

Following the recent insights from semantic segmentation literature, we choose a fully convolutional architecture for our aerial segmentation approach (AFCN). We base our network on the VGG-16 network [21] and initialize it with weights learned for classification on ImageNet [18]. We create three principal architecture variations in order to investigate the impact of popular design choices on the segmentation result in aerial images.

**AFCN-32** In a straightforward fully convolutional adaptation of the VGG16 network we base our segmentation on the feature map of the convolutional layer initialized with weights of $fc7$. Due to the prior five pooling layers this corresponds to a 32 factor downsampling. Upsampling back to the original input size is performed in a single deconvolutional layer.

**AFCN-MS** Following the original FCN architecture [12] we employ a multi-scale architecture which performs segmentation based on feature maps at layers $pool3$, $pool4$, and $fc7$. This corresponds to downsampling factors of 8, 16, and 32, respectively. The resulting segmentations are upsampled to the original input resolution through consecutive deconvolutional layers and combined through elementwise addition at the earliest possible stages.

**AFCN-D16** Finally, we use an architecture variation employing dilated convolutions. This increases the receptive field, which has recently shown to result in improved performance [24]. For this we set the stride of the $pool5$ layer to 1, decreasing the downsampling factor to 16 and instead apply a dilated convolution with dilation 2 at $fc6$ and $fc7$. The reduced downsampling allows us to retain more detail (i.e. local information) while at the same time increasing the receptive field through dilation and thus including more context (i.e. global information).

We generate training samples from the training data by cropping $256 \times 256$ dimensional patches with an overlap of 50%. The data is augmented by a factor of 24 through applying vertical or horizontal flipping as well as rotation in steps of 45 degrees. At test time we apply our model on image patches of size $512 \times 512$ with 50% overlap between patches. For each patch we select the central $384 \times 384$ pixels for the final segmentation. We found empirically that this methods results in slightly improved segmentation accuracy compared to averaging the overlapping regions and qualitatively leads to fewer visible edges.

We train our model for 100,000 iterations at a batch size of 6. We use the Adam solver with an initial learning rate of $1e-8$. 

fields. Marmanis et al. [14] combine the encoder-decoder architecture with a semantic edge detection to achieve better segmentation results at boundaries. Audebert et al. [3] leverage OpenStreetMap data to improve semantic segmentation.
3.2. Faster R-CNN

We use the Faster R-CNN approach (FRCN) proposed in [17] as our base detection system. The Faster R-CNN detector is composed of two modules: a region proposal network (RPN) followed by the Fast R-CNN detector [10]. The RPN takes an image as input and generates a set of candidate regions. For this, a small network is applied at each location of the output of the last convolution layer, which is used as feature map. The output of the network is a confidence value regarding the presence of an object, which is used to rank the generated proposals and the corresponding coordinates. The Fast R-CNN detector projects each region proposal onto the feature map and the corresponding features are extracted by a region of interest pooling layer to a feature vector of fixed length. Each feature vector is passed through a sequence of fully connected layers to a classification layer and a bounding box regression layer to refine the coordinates.

For our experiments, we use the VGG-16 network [21] pre-trained on ImageNet [18] as our base network. We removed the last group of convolutional layers, which exhibits better results for small objects [19] as in case of higher GSDs. We perform end-to-end training using the parameter settings of [1]. To account for the size of present objects, we set the anchor base size to 4 and the anchor scales to 8, 16, and 24. Due to the limited memory, we cropped the original images into subimages of size 600×600 for training and testing.

3.3. Fusion

We consider two methods of fusing the segmentation information into the detection process. The first is an early fusion at the stage of object proposals, as depicted in Figure 1, and the second is a later fusion after the proposals have been classified into vehicle detections.

As fusion method we propose a filtering of the proposals or detections using the segmentation information. This is accomplished by computing the distribution of segmentation classes within each bounding box and accepting or rejecting proposals and detections based on the characteristics of this distribution. A simple example would be to only accept proposals or detections which contain at least 10% car pixels. We investigate a number of filters based on such characteristics in Section 4.2 and propose a final, more complex filter which combines our findings.

Note that a filtering of detections is only able to reduce the number of false positives in order to improve the detection result. A filtering of the proposals additionally reduces runtime, because a smaller set of proposals will need to pass the classification stage of the detection network.

4. Evaluation

We evaluate our detection framework on the ISPRS 2D Semantic Labelling Challenge Potsdam dataset [2], which comprises 38 patches with a resolution of 6000×6000 pixels and a GSD of 5 cm. The dataset is split into 24 training patches with labeled ground truth (GT) for six categories (impervious surfaces, building, low vegetation, tree, car and clutter) and 14 test patches. As proposed in [20], we split the labeled training data into two subsets: one for training and one for validation. For our experiments, we use only the RGB imagery as additional channels (IR) or DSMs are not available for all aerial imagery. We adapted the GT annotations for class car as required for Faster R-CNN. For this, bounding boxes are generated around each segment labeled as car. Split and merged GT annotations are manually adjusted and missed cars (see Figure 2) are added.

To evaluate the semantic labeling results, we use overall accuracy and F1 score [2]. To evaluate the detection performance, we use Average Precision (AP) computed as defined in [8].

4.1. Semantic Labeling

In Table 1 we show the segmentation results of our described architectures. The single scale architecture (AFCN-32) achieves an overall accuracy across all classes of 87.68%. All classes are equally well segmented, with the exception of the class clutter which is by far the least accurate, because of the large variance of objects and concepts aggregated in this class. When we apply the multi-scale architecture (AFCN-MS) in combination with full data augmentation and overlapping patches, we observe a drop in overall accuracy.

Finally, our dilatation architecture is able to combine both aspects of the previous architecture. Through the reduced downsampling, compared to AFCN-32, we are able to achieve a high accuracy on the car class of 93.34%. Through the increased receptive field, we are able to better integrate this local detail with global information and achieve increased accuracy on the larger structures as well, resulting in an overall accuracy of 88.19%. Thus, for our further experiments we rely on the AFCN-D16 architecture. This result strikes a good balance between high accuracy on the car class and that of the other classes and slightly out-

---

1. Our extended GT is available at s.fhg.de/semseg-avss2017
Table 1. Semantic labeling results (pixelwise accuracy in %).

<table>
<thead>
<tr>
<th>Approach</th>
<th>Imp. surf.</th>
<th>Building</th>
<th>Low veg.</th>
<th>Tree</th>
<th>Car</th>
<th>Clutter</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>[20] RGB only</td>
<td>88.96</td>
<td>92.49</td>
<td>83.84</td>
<td>82.11</td>
<td>86.13</td>
<td>73.09</td>
<td>86.05</td>
</tr>
<tr>
<td>[20] RGB+IR+DSM</td>
<td>90.01</td>
<td>93.83</td>
<td>86.15</td>
<td>83.59</td>
<td>92.97</td>
<td>75.87</td>
<td>87.84</td>
</tr>
<tr>
<td>AFCN-32</td>
<td>89.42</td>
<td>92.14</td>
<td>83.80</td>
<td>83.61</td>
<td>90.76</td>
<td>54.95</td>
<td>87.68</td>
</tr>
<tr>
<td>AFCN-MS</td>
<td>86.36</td>
<td>86.44</td>
<td>79.14</td>
<td>80.87</td>
<td>89.29</td>
<td>47.29</td>
<td>83.92</td>
</tr>
<tr>
<td>AFCN-D16</td>
<td>89.69</td>
<td>92.93</td>
<td>84.54</td>
<td>84.54</td>
<td>93.34</td>
<td>57.35</td>
<td>88.19</td>
</tr>
</tbody>
</table>

Table 2. Detection performance for various criteria applied to filter proposals and detections (*).

<table>
<thead>
<tr>
<th>Filter Criterion</th>
<th>Semantic Labeling</th>
<th>AP (%)</th>
<th># Proposals/Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>-</td>
<td>92.9</td>
<td>100</td>
</tr>
<tr>
<td>50% Building</td>
<td>GT</td>
<td>93.5</td>
<td>84.8</td>
</tr>
<tr>
<td>50% Low veg.</td>
<td>GT</td>
<td>93.1</td>
<td>77.7</td>
</tr>
<tr>
<td>50% Tree</td>
<td>GT</td>
<td>87.9</td>
<td>89.8</td>
</tr>
<tr>
<td>50% Clutter</td>
<td>GT</td>
<td>93.6</td>
<td>95.1</td>
</tr>
<tr>
<td>Eq. (1)</td>
<td>GT</td>
<td>94.3</td>
<td>50.3</td>
</tr>
<tr>
<td>Eq. (1)</td>
<td>AFCN-D16</td>
<td>93.9</td>
<td>51.1</td>
</tr>
<tr>
<td>Eq. (1)*</td>
<td>GT</td>
<td>93.7</td>
<td>100</td>
</tr>
</tbody>
</table>

performs the results achieved in [20] which additionally rely on IR and DSM information.

4.2. Detection

The detection performance of our detection framework compared to the baseline Faster R-CNN is given in Table 2. To evaluate the impact of each segmentation category which is unlikely to surround or contain a vehicle on filtering of the proposals, we separately remove proposals whose pixels are labeled at least 50% as building, low vegetation, tree, or clutter, respectively. For this, we use the provided semantic labeling GT. Removing proposals that are mainly labeled as building or clutter clearly improves the detection performance as false alarms due to objects with shapes similar to vehicles, like windows on roofs, are filtered out. Instead, removing proposals mainly labeled as tree causes a drop in performance due to the semantic labeling GT. As depicted in Figure 2 (left) several vehicles, which are clearly visible, are parked beneath a tree and consequently labeled as tree. Removing proposals mainly labeled as low vegetation shows only minor improvements in AP. However, the number of proposals per image that are classified is clearly reduced.

To remove as many false alarms as possible and to minimize the number of proposals to classify, we use in the further course of this work only proposals that fulfill following equation:

$$\max\left(\frac{N_{\text{car}}}{N_{\text{bg}}}, \frac{N_{\text{imp. surf.}}}{N_{\text{bg}}}, \frac{N_{\text{tree}}}{N_{\text{bg}}} \right) > 1,$$  (1)

where $N_{i}$ is the number of pixels corresponding to class $i$ within a proposal. $N_{\text{bg}}$ is the sum of all pixels labeled as building or low vegetation. Thus, only proposals that are mainly labeled as car, impervious surface, or tree are considered for classification. The AP is increased by 1.4% compared to the baseline Faster R-CNN while the number of proposals, which are classified, is more than halved. Reason for the better AP is the clearly reduced number of false positive detections, i.e. for a classification confidence threshold of 0.5 the number of false positive detections is reduced by a factor of 0.67.

Using the AFCN-D16 semantic labeling results instead of the GT shows a slightly worse AP due to incorrect predicted labels, e.g. confusion between clutter and car. However, the detection performance is still improved compared to the Faster R-CNN baseline. Using the detection method of [4] results in worse detection performance compared to our proposed approach. We achieve precision and recall values of 96.6% and 91.8%, respectively, for a classification confidence threshold of 0.5, while the precision value of 90.9% and recall value of 82.7% for [4] are considerably less. Reasons for the worse performance are split and merged detections (see Figure 2 (right)) as well as missed detections as many cars are incorrectly labeled as tree as discussed above.

Filtering the final detections instead of the proposals exhibits similar results while the number of proposals that are classified is not reduced. Hence, we use the proposals filtering approach for the following experiments.

4.3. Different Ground Sampling Distances

In the following, we evaluate the impact of applying semantic labeling to filter proposals for various GSDs. For this, we down-scaled the images by factor 2, 3, 4, and 5, respectively. To generate the semantic labeling masks, we use AFCN-D16 trained on the original image resolution instead of training a net for each resolution separately. During testing, we up-scale the down-scaled images to the original image resolution. The corresponding semantic labeling results are given in Table 3. The performance for all classes decreases with higher GSDs and lower ground resolution. The drop is only minor for all classes and the overall accuracy for a GSD of 10 cm. Category tree exhibits the strongest decrease for higher GSDs. As depicted in Figure 2, class tree is due to the season only represented by thin tree branches. However, such fine structures are eliminated during down-
Table 3. Semantic labelings results for different GSDs using AFCN-D16 (in %).

<table>
<thead>
<tr>
<th>GSD (in cm)</th>
<th>Imp. surf.</th>
<th>Building</th>
<th>Low veg.</th>
<th>Tree</th>
<th>Car</th>
<th>Clutter</th>
<th>Overall Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>89.69</td>
<td>92.93</td>
<td>84.54</td>
<td>84.54</td>
<td>93.34</td>
<td>57.35</td>
<td>88.19</td>
</tr>
<tr>
<td>10</td>
<td>89.35</td>
<td>92.30</td>
<td>83.67</td>
<td>83.08</td>
<td>93.04</td>
<td>55.09</td>
<td>87.45</td>
</tr>
<tr>
<td>15</td>
<td>84.43</td>
<td>89.65</td>
<td>75.54</td>
<td>44.99</td>
<td>91.24</td>
<td>37.36</td>
<td>77.03</td>
</tr>
<tr>
<td>20</td>
<td>76.08</td>
<td>84.93</td>
<td>70.86</td>
<td>18.25</td>
<td>87.79</td>
<td>28.40</td>
<td>68.60</td>
</tr>
<tr>
<td>25</td>
<td>70.10</td>
<td>75.75</td>
<td>69.76</td>
<td>11.60</td>
<td>78.01</td>
<td>19.29</td>
<td>63.04</td>
</tr>
</tbody>
</table>

Table 4. Average Precision for different GSDs (in %).

<table>
<thead>
<tr>
<th>GSD (in cm)</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRCN</td>
<td>92.9</td>
<td>92.2</td>
<td>90.1</td>
<td>82.7</td>
<td>62.9</td>
</tr>
<tr>
<td>FRCN + Eq. (1)</td>
<td>93.9</td>
<td>93.3</td>
<td>91.7</td>
<td>85.8</td>
<td>70.7</td>
</tr>
</tbody>
</table>

scaling and consequently assigned to wrong classes.

The best results for all GSDs is achieved for category car, which shows a F1 score above 78% even for a GSD of 25 cm. Good F1 scores around 70% are achieved for the categories impervious surface, building and low vegetation, which are essential to distinguish regions of interest or not.

Table 4 shows the detection performance for various GSDs. For this, we trained a Faster R-CNN as described in Section 3.2 for each GSD. The detection performance of Faster R-CNN with and without filtering decreases with higher GSDs. To filter proposals, we use the semantic labeling mask generated for the corresponding resolution and Equation 1. The detection performance is improved for all GSDs. For GSDs up to 15 cm, the AP is improved by around 1% whereas the AP considerably increases for higher GSDs even though the semantic labeling accuracy get worse for higher GSDs (see Table 3).

4.4. Qualitative Results

Figure 3 shows proposals (first column) and detection examples before (top row) and after (bottom row) filtering the proposals with a semantic labeling mask (middle row). For this, we use AFCN-D16 and Equation 1. The number of proposals to classify is clearly reduced as proposals that mainly overlap with category low vegetation or category building are filtered out. Furthermore, less false alarms are produced as objects with shapes similar to vehicles, e.g. windows on buildings, are removed.

5. Conclusion

In this paper, we presented a detection framework that fuses Faster R-CNN and semantic labeling. We demonstrated that the number of false alarms is clearly reduced by integrating semantic information while the number of candidate regions to classify is halved. For this, we evaluated different semantic labeling architectures and analyzed various criteria to filter proposals. In future work, we will merge the semantic labeling net and the detection net to enable end-to-end training and reduce the computational effort during runtime.

Figure 3. Proposals (first column) and detection examples before (top row) and after (bottom row) filtering the proposals with the semantic labeling mask achieved for AFCN-D16. Filtering the proposals reduces the number of proposals to classify and results in less false alarms due to objects with shapes similar to vehicles, e.g. windows on buildings.
References


