

**EXPLORANDO REDES NEURAS
CONVOLUCIONAIS PARA DESCRIÇÃO
CONTEXTUAL BASEADA EM SUPERPIXELS**

EDUARDO DE ARAÚJO TAVARES

**EXPLORANDO REDES NEURAIIS
CONVOLUCIONAIS PARA DESCRIÇÃO
CONTEXTUAL BASEADA EM SUPERPIXELS**

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR: JEFERSSON ALEX DOS SANTOS

Belo Horizonte
Dezembro de 2018

EDUARDO DE ARAÚJO TAVARES

**EXPLOITING CONVOLUTIONAL NEURAL
NETWORKS FOR SUPERPIXEL BASED
CONTEXTUAL DESCRIPTION**

Dissertation presented to the Graduate Program in Computer Science of the Universidade Federal de Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

ADVISOR: JEFERSSON ALEX DOS SANTOS

Belo Horizonte

December 2018

© 2018, Eduardo de Araújo Tavares.
Todos os direitos reservados.

T231e Tavares, Eduardo de Araújo
Exploiting Convolutional Neural Networks for
superpixel based contextual description / Eduardo de
Araújo Tavares. — Belo Horizonte, 2018
xxiv, 63 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de
Minas Gerais

Orientador: Jefersson Alex dos Santos

1. Computação — Teses. 2. Detecção de objetos.
3. Sensoriamento remoto. 4. Rede convolucional.
I. Orientador. II. Título.

CDU 519.6*82.10(043)



UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO


FOLHA DE APROVAÇÃO

Exploiting Convolutional Neural Networks for superpixel based contextual
description


EDUARDO DE ARAUJO TAVARES

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:


PROF. JEFERSSON ALEX DOS SANTOS - Orientador
Departamento de Ciência da Computação - UFMG


PROF. ADRIANO ALONSO VELOSO
Departamento de Ciência da Computação - UFMG


PROF. GUILLERMO CÁMARA CHÁVEZ
Departamento de Computação - UFOP


DR. THIAGO VALLIN SPINA
Laboratório Nacional de Luz Síncrotron - CNPEM

Belo Horizonte, 12 de dezembro de 2018.

A Alberto Campolina.
Pour la joie, le bonheur, la croissance
et surtout l'avenir

*“Eheu fugaces labuntur anni.
Vincit qui se vincit.
Incepto ne desistam.
Igne natura renovatur integra.
Flectere si nequeo superos, Acheronta movebo.
Fac fortia et patere.”*
()

Resumo

Sensoriamento remoto é uma importante técnica para a obtenção de observações consistentes de fenômenos e processos em larga escala. Entretanto, o aumento exponencial da quantidade de dados causado pela melhoria contínua da frequência e taxa de cobertura torna a interpretação manual de dados brutos impraticável em muitos casos. A obtenção de procedimentos automatizados para a extração de informações semânticas úteis é crucial para possibilitar as mais recentes aplicações baseadas em sensoriamento remoto.

Métodos automáticos oferecem uma ampla gama de benefícios e são o foco de muitas áreas de pesquisa. Aplicações potenciais incluem desde a análise de mudanças na vegetação e clima, derretimento das calotas polares, etc. ao mapeamento de desastres naturais como terremotos, tsunamis, deslizamentos de terra ou avalanches, ou ainda o rastreamento de objetos móveis, por exemplo, para monitoramento de tráfego ou vigilância e planejamento urbano. Dada a complexidade desses fenômenos, um processamento completamente automático geralmente ainda não é possível. Entretanto, o processamento objetivo, rápido e confiável dos dados obtidos em todos esses casos é de grande importância e requer métodos avançados.

O crescimento da resolução espacial dos satélites aumentou o escopo para a extração de *features*, levando a um aumento no número de classes de utilização do solo, entretanto a semântica inerente a essas classes geralmente não é explorada. Métodos de classificação a nível de pixel perderam então sua eficácia, dado que a relação entre o tamanho dos pixels e a dimensão dos objetos observados na superfície da Terra foi alterada significativamente. Portanto, a classificação baseada em objetos tornou-se cada vez mais popular ao longo desta década. Ela combina segmentação e classificação baseada em contexto. Segmentação divide a imagem em grupos de pixels homogêneos (segmentos), que são agrupados em classes com base em suas características espectrais, geométricas e de textura. Entretanto essa tarefa é bastante desafiadora e descritores de baixo nível usualmente utilizados ignoram a informação semântica que pode ser extraída a partir da configuração espacial das classes presentes numa imagem. *Deep*

learning, um avanço recente no campo do Aprendizado de Máquina, vem possibilitando novas abordagens para esse problema.

O objetivo deste trabalho é de avaliar a aplicabilidade de um método de descrição contextual baseado em superpixels usado para classificação de imagens ao cenário de detecção de objetos em imagens de sensoriamento remoto. Este método explora redes convolucionais para extrair características de diferentes níveis contextuais em torno dos superpixels obtidos. O método foi modificado com o intuito de avaliar diferentes métodos de *pooling* e foi testado em bases de dados disponíveis publicamente assim como em bases customizadas criadas para testar a aptidão do método nos cenários de localização e detecção de objetos. Níveis de acurácia acima de 99% foram obtidos na tarefa de localização de árvores e carros.

Palavras-chave: detecção de objetos, sensoriamento remoto, contexto.

Abstract

Remote sensing is an important technique for acquiring consistent, repeated high resolution observations of large-scale phenomena and processes. However, the ever increasing amount of data provided by the continuing improvement in coverage and repetition rates in recent years makes the manual interpretation of the raw data impracticable in many cases. Devising automatic procedures to extract useful semantic information is then crucial for enabling the latest remote sensing based applications.

Automatic methods offer a wide gamut of benefits and are the focus of many research fields. Potential applications range from detecting changes in vegetation and climate, ice melt, etc. to the mapping of natural disasters such as earthquakes, tsunamis, landslides or avalanches, to tracking of moving objects, e.g. for traffic monitoring or surveillance and urban planning. Due to the complexity of these phenomena, fully automatic processing is often not yet possible. However, reliable, fast and objective processing of the recorded data in all these cases is of great importance and requires advanced methods.

The evolution in spatial resolution of satellites has increased the scope for feature extraction leading to a rising number of land cover classes, while the underlying semantics of these classes were not explored. Pixel-based classification methods became then less effective, since the relationship between the pixel size and the dimension of the observed objects on the Earth's surface has changed significantly. Therefore object-oriented classification has become increasingly popular over the past decade. This combines segmentation and contextual classification. Segmentation divides the image into homogeneous pixel groups (segments), which are arranged into classes based on their spectral, geometric, textural and other features. However, this task is notoriously challenging and low-level descriptors usually employed ignore semantic information that may be provided by the spatial configuration of the classes present in an image. Deep learning, a recent breakthrough in machine learning, has shed light on this problem.

The aim of this work is to evaluate the applicability of a superpixel based contextual description method used for image classification to the scenario of object detection

in remote sensing imagery. Such method exploits convolutional networks to compute deep contextual features from different context levels surrounding the superpixels obtained. The method is modified in order to assess the use of different pooling methods and is tested on publicly available datasets and custom datasets created to evaluate the method's suitability to different scenarios. Accuracy levels above 99% were obtained in the tasks of car and tree localisation.

Palavras-chave: Object detection, Remote Sensing, Context.

List of Figures

2.1	Taxonomy of methods for object detection in optical RSIs [Cheng and Han, 2016].	7
2.2	Object shown with no context information.	9
2.3	Object shown with spatial context information being provided.	9
2.4	Identification of the object when shown with its semantic, spatial and scale context becomes a much easier task.	10
2.5	Flowchart of machine learning-based object detection [Cheng and Han, 2016].	12
2.6	Detecting horizontal edges from an image using convolution filtering [Stenroos et al., 2017].	13
2.7	An example of a convolutional network [Stenroos et al., 2017].	14
3.1	The proposed representation to exploit all contextual levels ranging from the superpixel itself to an entire image patch containing it. Given a target superpixel s_i , its final representation is the concatenation of the ℓ_2 -normalized features extracted from s_i , a small rectangular contextual area surrounding s_i and a large contextual area [Santana et al., 2017].	20
3.2	Examples of patches containing cars after the performance of the second segmentation of the proposed method.	21
3.3	Example of the approach proposed by Mostajabi et al. [2015] to extract deep features from the superpixel s_i . It consists in keeping a mapping between each pixel inside s_i and the corresponding points of the feature maps as the image I is forwarded across the network. So it is possible to generate just one k -dimensional feature vector after each convolutional layer by average pooling the k feature maps over s_i . When the resolution of the feature maps is reduced by the stride in pooling and convolutional layers, an upsampling is employed in order to restore their original size and consequently keep the mapping [Santana et al., 2017].	23

3.4	The proposed approach for deep contextual feature extraction with AlexNet. Given a superpixel s_i , an image patch I is created centering the superpixel and used as input for the AlexNet. The features are computed considering three levels of context (or three layers): $\phi_1(s_i)$, $\phi_5(s_b)$ and $\phi_{fc2}(I)$. Adapted from [Santana et al., 2017].	24
3.5	Toy example illustrating the drawbacks of max pooling and average pooling [Yu et al., 2014].	25
4.1	Images from grss_dfc_2014 used in the experiments.	28
4.2	Samples from the dataset created for the task of cars detection. Top row: positive samples. Bottom row: negative samples.	29
4.3	Samples from the dataset created for the task of trees detection. Top row: positive samples. Bottom row: negative samples.	29
4.4	Images from the Munich Vehicle Dataset.	30
4.5	Groundtruth of the test image along with the map generated by the proposed method with XGBoost classifier using layers Conv1, Conv5 and fc8.	34
4.6	Maps generated for image 0110 from the test subset from the Munich Vehicle Detection dataset, with the number of samples used for training the classifier ranging from 5K to 50K.	39
4.7	Image 0110 from the test subset of the Munich Vehicle Detection dataset.	40
4.8	Groundtruth map of image 0110 from the test subset of the Munich Vehicle Detection dataset.	41
4.9	Map generated by the proposed method for image 0110 from the test subset from the Munich Vehicle Detection dataset, with the number of superpixels used for segmenting the entire image ranging from 20K to 30K and each selected frame segmented in 36, 48 and 60 superpixels.	42
4.10	Map generated by the proposed method for image 0110 from the test subset from the Munich Vehicle Detection dataset, with the number of superpixels used for segmenting the entire image ranging from 40K to 50K and each selected frame segmented in 36, 48 and 60 superpixels.	43
4.11	Map generated by the proposed method for image 0110 from the test subset from the Munich Vehicle Detection dataset, with the number of superpixels used for segmenting the entire image ranging from 60K to 70K and each selected frame segmented in 36, 48 and 60 superpixels.	44

4.12	Visual comparison of the results obtained for image 0110 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	45
4.13	Visual comparison of the results obtained for image 0120 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	46
4.14	Visual comparison of the results obtained for image 0120 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	47
4.15	Visual comparison of the results obtained for image 0140 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	48
4.16	Visual comparison of the results obtained for image 0150 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	49
4.17	Visual comparison of the results obtained for image 0160 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	50
4.18	Visual comparison of the results obtained for image 0250 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	51

4.19	Visual comparison of the results obtained for image 0265 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	52
4.20	Visual comparison of the results obtained for image 0278 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	53
4.21	Visual comparison of the results obtained for image 0120 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.	54

List of Tables

4.1	Configurations of decision trees used as classifiers.	31
4.2	Configurations of SVMs used as classifiers.	31
4.3	Configurations of KNNs used as classifiers.	31
4.4	Configurations of Ensembles used as classifiers.	31
4.5	Layer importance analysis in grss_dfc_2014 dataset using the XGBoost classifier performed in [Santana et al., 2017].	33
4.6	Layer importance analysis in grss_dfc_2014 dataset by using different pooling strategies.	33
4.7	Average accuracy obtained for cars detection on 4 different classifiers with the layers of AlexNet used as features and the pooling methods employed.	35
4.8	Average accuracy obtained for trees detection on 3 different classifiers with the layers of AlexNet used as features and the pooling methods employed.	36
4.9	Results obtained on the Munich Vehicle Dataset.	37
4.10	Performance comparison between different methods as reported in [Deng et al., 2017].	37

Contents

Resumo	xiii
Abstract	xv
List of Figures	xvii
List of Tables	xxi
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	2
1.3 Outline	3
2 Background and Related Work	5
2.1 Object Detection in Remote Sensing	5
2.2 Superpixel representation	10
2.2.1 Segmentation	10
2.2.2 Context representation	11
2.3 Convolutional Neural Networks	13
2.3.1 Pooling	14
2.3.2 Additional layers	15
2.3.3 Regularization	16
2.3.4 Development	16
3 Methodology	19
3.1 Overview	19
3.2 Segmentation	20
3.3 Multi-context feature extraction	21
3.4 Pooling strategies	24

4	Experimental Analysis	27
4.1	Setup	27
4.1.1	Datasets	27
4.1.2	Classifiers and Training Protocol	30
4.1.3	Metrics	32
4.2	Evaluation in semantic segmentation task	32
4.2.1	Results on grss_dfc_2014 Dataset	32
4.3	Evaluation in detection task	34
4.3.1	Results on Potsdam dataset	34
4.3.2	Results on Munich Vehicle Dataset	36
5	Conclusion	55
	Bibliography	57

Chapter 1

Introduction

The ability to identify the objects present in an image or scene is one of the most basic requirements when it comes to interacting with one's environment. While it seems completely effortless for humans, and in fact for most animals, trying to teach computers to see and also to “understand” what they are seeing has been proved to be extremely difficult [Mohammed, 2014].

The key to understanding visual scenes are three closely related sub-problems. The easiest one consists in classification, in which the one dominant object in a given image should be determined and labelled. The next more demanding task is object localization: in addition to labelling the dominant object, it also needs to be localized in the image, usually by determining a bounding box around the image region that is occupied by the object. The difficulty of this task increases if not only one but all objects in an image need to be labelled and multiple objects of the same category can appear in an image. This task is called object detection [Kloss, 2015].

1.1 Motivation

Detecting and identifying the different objects in an image quickly and reliably is an important skill for interacting with one's environment and there is a vast number of applications where object detection is an essential component. All systems that rely on visual input for reasoning about the environment use object detection in some form. The main problem is that in theory, all parts of an image have to be searched for objects on many different scales to make sure that no object instance is missed. However, it takes considerable time and effort to actually classify the content of a given image region and both time and computational capacities that an agent can spend on

classification are limited. Since the current state-of-the-art for most objects is well below human capabilities, the research in the field is active and important.

One of the tasks for which object detection also plays an important role is the analysis of Remote Sensing imagery. The usual pipeline for such analysis involves the extraction of low-level descriptors from few image samples that are annotated by the user, and used to train a classifier. The generated classifier should be able to annotate the remaining samples in the image and its accuracy depends on the quality of the descriptors and the training samples selected [Santana et al., 2017].

Remote Sensing Imagery (RSI) classification has been usually based on pixel statistics analysis, but as the spatial resolution of images increased, the information from neighboring pixels (either texture or context) was used to improve results. Traditional low-level appearance features, such as color or shape, are limited for capturing the appearance variability of real-world objects represented in images. Noise and changes in lighting conditions are factors that cause an increase in intra-class variance, leading to classification errors. The coherent arrangement of the elements expected to be found in real world scenes has been shown to improve classification results through a contextual description of the image [Santana et al., 2017].

1.2 Objectives

The objective of this work is the evaluation of the applicability of a superpixel-based contextual description method used for image classification to the scenario of object detection in remote sensing images. Such method exploits convolutional networks to compute deep contextual features from different context levels surrounding the obtained superpixels. A deep feature is the consistent response of a node or layer within a hierarchical model to an input that gives a response that is relevant to the model's final output. One feature is considered "deeper" than another depending on how early in the decision tree or other framework the response is activated. Convolutional Neural Networks (ConvNets or CNNs) are a category of Neural Networks that uses perceptrons, a machine learning unit algorithm to analyze data, that have proven very effective in areas such as image recognition and classification. It is comprised of one or more convolutional layers (often with a subsampling step) and then followed by one or more fully connected layers as in a standard multilayer neural network. The architecture of a CNN is designed to take advantage of the 2D structure of an input image.

The research questions this work aims to answer are:

- Does the use of contextual features improve the accuracy of object detection in

remote sensing imagery?

- How does the tested methodology compare with the baselines in the literature?
- What is the impact of changing the pooling strategy employed in CNNs used as feature extractors for the task of object detection in remote sensing imagery?

The method is modified in order to assess the use of different pooling strategies and an improved segmentation approach is proposed in this work. The proposed modifications are validated on car and tree localization and car detection scenarios.

1.3 Outline

The remainder of this work is organized as follows: **Chapter 2** provides general background on object detection, superpixel-based segmentation and how convolutional neural networks can help in object detection by encoding semantic context.

Chapter 3 details the proposed method, while the settings used in the validation of the method and the results obtained are described in **Chapter 4**.

Finally **Chapter 5** brings conclusions, recommendations for future research, and final remarks.

Chapter 2

Background and Related Work

This chapter presents background concepts on the task of object detection in remote sensing images as well as related works to such task.

2.1 Object Detection in Remote Sensing

Remote sensing is defined by [Lillesand et al. \[2014\]](#) as the science of obtaining information about an object, area, or phenomenon through the analysis of data acquired by a device that is not in contact with the object, area, or phenomenon under investigation. Object detection in remote sensing images is a fundamental yet challenging problem in the field of aerial and satellite image analysis, as it plays an important role for a wide range of applications such as geological hazard detection, geographic information system (GIS) update, environmental monitoring, LULC mapping, precision agriculture, urban planning, etc [[Cheng and Han, 2016](#)]. It consists in determining whether a given aerial or satellite image contains one or more objects belonging to a class of interest and to locate the position of each predicted object in the image [[Cheng and Han, 2016](#)].

The process of automatic object detection in optical remote sensing imagery (RSI) is comprised of the following steps: data acquisition, data preparation, image segmentation (it is optional depending on whether a pixel-based or a region-based approach is used), feature extraction, model training, and object detection. The data acquisition step consists in the detection and storage of the electromagnetic radiation reflected or emitted by objects or phenomena on the Earth surface. The sensors used may be either passive sensors, which only register the reflected radiation, or active sensors, which emit radiation and register it after its interaction with one or more targets [[Lillesand et al., 2014](#)].

The second step is needed due to the fact that RSIs usually contain noise and errors due to the atmospheric interference and imaging geometry, so pre-processing techniques are applied in order to soften noise and correct radiometric and geometric distortions [Meneses et al., 2012].

In the next step, objects are delineated in the image by grouping their pixels so that the entire image is composed of many disjoint regions. Segmentation is required when the image analysis approach chosen is based on regions or objects. In order to avoid ambiguity, object is defined as follows:

Definition 1. An object is any meaningful and distinguishable entity depicted in an image.

The resolution of the images considered is of fundamental importance to the applications that will be based on them. It is then defined by Gonzalez et al. [2004] as:

Definition 2. The spatial resolution is the linear measurement of the distance imaged on the ground per pictorial element (pixel) of the image.

The next step produces a representation of image samples that describes them in terms of some kind of visual cue, such as color, texture and shape, or even a combination of them. These representations are regarded as low-level, once they are based on computations over the pixels themselves. More complex representations also encode the context of the objects depicted in the images or apply some transformations over the low-level representations in order to generate a higher-order one, named middle level representation [Perronnin and Dance, 2007]. Therefore, all the mentioned representations depend somehow on the low-level ones, which in turn are computed through image descriptors [Dalal and Triggs, 2005].

Model training and label prediction are two closely related steps. Problems are solved through the use of a statistical model defined in terms of some parameters. Learning is the process of running an algorithm to optimize the parameters of the model according to the example data available (also known as training data or training samples). Once the model is trained, it may be used to make predictions or inferences.

Object detection in optical RSIs faces several challenges like the large variations in the visual appearance of objects caused by viewpoint variation, occlusion, background clutter, illumination, shadow, etc., the growth of RSIs in quantity and quality, and the various requirements of new application areas. To address these challenges, the topic

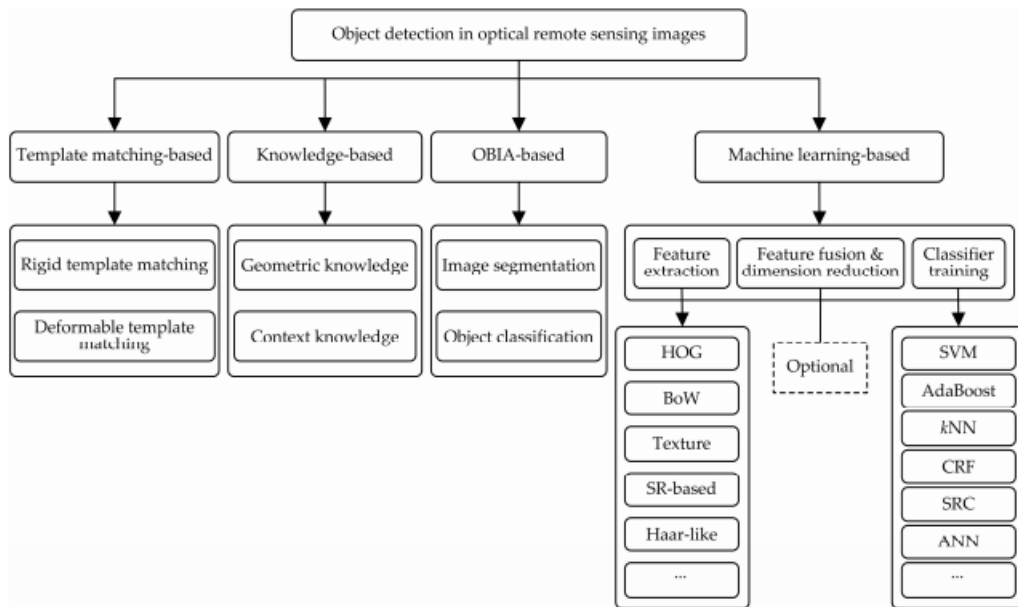


Figure 2.1. Taxonomy of methods for object detection in optical RSIs [Cheng and Han, 2016].

of geospatial object detection has been extensively studied since the 1980s [Cheng and Han, 2016].

Considerable efforts have been made to develop methods for the detection of different types of objects in satellite and aerial images, such as roads, buildings, trees, vehicles, etc. We can generally divide them into four main categories: template matching, knowledge-based, OBIA (Object-based image analysis)-based, and machine learning-based methods [Cheng and Han, 2016]. Fig. 2.1 shows a taxonomy of geospatial object detection methods.

Template matching-based methods are one of the simplest and earliest approaches for object detection. There are two main steps in template matching-based object detection framework. 1) Template generation: a template T for each to-be-detected object class should be firstly generated by hand-crafting or learning from the training set. 2) Similarity measure: given a source image, the stored template T is used to match the image at each possible position to find the best matches, according to the minimum distortion or maximum correlation measures, while taking into account all allowable translation, rotation, and scale changes. The most popular similarity measures are the sum of absolute differences (SAD), the sum of squared differences (SSD), the normalized cross correlation (NCC), and the Euclidean distance (ED) [Cheng and Han, 2016].

Knowledge-based object detection methods are another type of popular approaches for object detection in optical RSIs. An extensive collection of papers on

knowledge-based object detection have been published for buildings, roads and other more general object extraction applications like landslide, bridges, vehicles, urban land changes, crops, drainage channels, and forests. This type of approaches generally translates object detection problem into hypotheses testing problem by establishing various knowledge and rules. The establishment of knowledge and rules is the most important step. Two kinds of widely used knowledge on target objects are geometric knowledge.

The object geometric information is the most important and widely used knowledge for object detection, which encodes prior knowledge by taking parametric specific or generic shape models. The context knowledge is another crucial cue for knowledge-based object detection and the most widely used context knowledge is the spatial constraints or relationships between objects and background, or the information regarding how the object interacts with its neighboring regions. The core of knowledge-based object detection methods is how to effectively transform the implicit knowledge understanding on target objects into the explicit detection rules. If the defined rules are too strict, some target objects will be missed; conversely, too loose rules will cause false positives [Cheng and Han, 2016].

According to [Galleguillos and Belongie, 2010], contextual knowledge is any information which was not produced by the appearance of the own object, but by nearby image data or metadata related to the image, such as tags or image annotations. The authors divide existing approaches for contextual description into three categories: semantic, scale and spatial, each of them being regarded as either local or global context. The semantic context of an object O is the likelihood of O is in a given image I with a set of objects S , while the spatial context of an object O is the orientation and localization of O relative to each other object in image I ; and finally, the scale context of an object O is the size of O relative to all other objects in an given image I .

A toy example that illustrates the usefulness of context information is depicted in Fig. 2.2. When asked what is the object shown in the image, a person might have difficulty to identify it. Once spatial and semantic context is provided (as depicted in Fig. 2.3 and Fig. 2.4), such task becomes much easier to be performed.

With the increasing availability and wide utilization of sub-meter imagery, object-based image analysis (OBIA or GEOBIA for geospatial object based image analysis) has become a new methodology or paradigm to classify or map VHR imagery into meaningful objects (or rather, grouping of relatively local homogeneous pixels). OBIA involves two steps: image segmentation and object classification. Since OBIA offers the potential to exploit geographical information system (GIS) functionality, such as the incorporation of the spatial context or object shape in the classification, it provides a framework for overcoming the limitations of conventional pixel-based image classifi-



Figure 2.2. Object shown with no context information.



Figure 2.3. Object shown with spatial context information being provided.



Figure 2.4. Identification of the object when shown with its semantic, spatial and scale context becomes a much easier task.

cation methods and has been successfully applied to landslide mapping, land cover and land use mapping and change detection.

2.2 Superpixel representation

2.2.1 Segmentation

Recently, superpixel has been researched and applied in computer vision tasks, e.g. object location and tracking, and class segmentation [Achanta et al., 2012; Sun and Chi, 2015]. The segmentation approach is now preferred over the sliding-window technique for object categorization and recognition as it decreases the amount of analysis of candidate locations for object categorization and recognition, by reducing candidate locations from tens of thousands of windows to thousands or even hundreds of windows. Segmentation has attracted increased attention in the computer vision community, and a wide range of segmentation algorithms have been developed [Ammour et al., 2017].

Superpixels provide clusters of perceptually similar pixels throughout an image, thus capturing the redundancy inherent in most natural images [Pappas et al., 2017], being an effective manner to obtain spatial structure information [Jia et al., 2017]. Each superpixel is a meaningful region, which can represent the spatial structure of an image with adaptive shapes and sizes. Moving to superpixels allows us to measure feature statistics on a naturally adaptive domain rather than on a fixed window [Fulkerson et al., 2009].

The results of superpixel algorithms may vary in quality, uniformity, size and number, partly due to the lack of a consistent, rigorous definition of what constitutes

a superpixel [Pappas et al., 2017]. One of the most widely used algorithms, proposed by Achanta et al. [2012] is *Simple Linear Iterative Clustering* (SLIC). SLIC segments an image according to a 5-dimensional distance metric comprised of spatial (x, y coordinates) and colour information (L, a, b , of the CIELAB colorspace) as shown in the following equations:

$$d_{lab} = \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2} \quad (2.1)$$

$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2} \quad (2.2)$$

$$D_s = d_{lab} + \frac{m}{S}d_{xy} \quad (2.3)$$

The number of generated superpixels k (and indirectly, their size) is specified by the user; $\frac{m}{S}$ is a scaling factor where S is the initial cluster seed grid interval (dependent on image size and desired k) and m allows the user to control superpixel compactness and shape regularity [Achanta et al., 2012].

2.2.2 Context representation

With the advance of machine learning techniques, especially the powerful feature representations and classifiers, many recent approaches regarded object detection as a classification problem and have achieved significant improvements. Fig. 2.5 gives the flowchart of machine learning-based object detection, in which object detection can be performed by learning a classifier that captures the variation in object appearances and views from a set of training data in a supervised, semi-supervised or weakly supervised framework. The input of the classifier is a set of regions (sliding windows or object proposals) with their corresponding feature representations and the output is their respective predicted labels, i.e., object or not.

Compared with object detection in natural scene images, detection of targets in RSIs is a more challenging task [Deng et al., 2017]. Handcrafted features have recently been significantly outperformed by deep learning based methods, which are now perceived as the most effective methods for image classification. Recently, many CNN-based object detection pipelines have been proposed with impressive performance [Krizhevsky et al., 2012a; He et al., 2016a].

Long et al. [2015] employed visual saliency to generate a small number of bounding boxes, and then extracted features using deep belief networks, a method suitable for simple environments. Hariharan et al. [2015] proposed a rotation-invariant CNN model

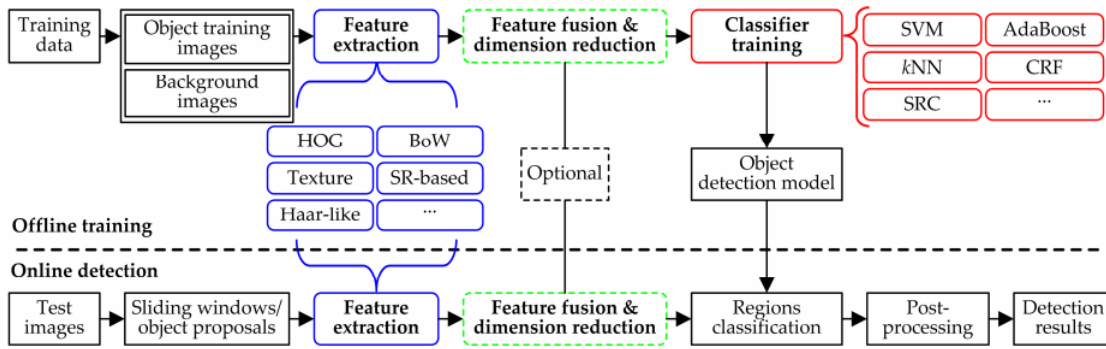


Figure 2.5. Flowchart of machine learning-based object detection [Cheng and Han, 2016].

for object detection in RSIs, while Diao et al. [2016] proposed an aircraft detection method based on coupled CNNs. In the case of vehicle detection, Chen et al. [2014] proposed a method based on sliding windows and deep CNN called the hybrid deep neural network (HDNN).

The effectiveness and sampling simplicity of sliding-windows made it the most popular approach used in the past few years for object detection, recognition, and localization [Lampert et al., 2008; Vedaldi et al., 2009]. But it is a time-consuming process and tends to obtain a possible location for entire non-rigid or non-canonical posed objects. Furthermore, window bounding the object may also cover much of the background area, which may corrupt the evaluation [Ammour et al., 2017]. Recent advances in computer vision indicate that state-of-the-art methods for object detection are moving away from the use of sliding-window to search for possible object locations [Girshick et al., 2014a; Wang et al., 2013; Uijlings et al., 2013; Ren et al., 2015], instead, they rely on segmentation in a pre-processing step for region proposal.

A number of works utilize one or more segmentations as a starting point for their task. Li et al. [2016] proposed a pixel-level and superpixel-level probabilistic fusion method for HSI (Hyperspectral Images) classification. Superpixel-level discriminative sparse model and multitask sparse representation methods are proposed by Fang et al. [2015b] and Li et al. [2015]. Fang et al. [2015a] proposed a novel HSI classification framework to exploit the spectral-spatial information of superpixel via multiple kernels (SC-MK). He et al. [2016b] studied a superpixel-based group sparse and low-rank model for HSI classification. These superpixel-based methods present smoother classification maps and obtain more precise results [Shi and Pun, 2018]. The two winning algorithms for object detection in the ImageNet 2013 detection challenge also relied on segmentation [Ammour et al., 2017].

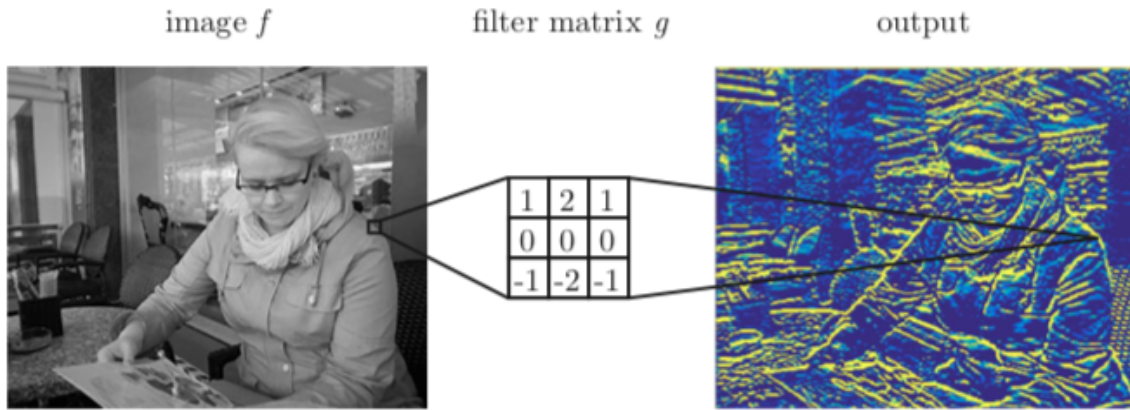


Figure 2.6. Detecting horizontal edges from an image using convolution filtering [Stenroos et al., 2017].

For these reasons, we use the contextual feature extraction from superpixels approach proposed in Mostajabi et al. [2015] for object detection. We adapted it in order to explore different pooling methods employed after convolutional layers of deep networks. Such method is evaluated on the scenarios of vehicle and trees detection.

2.3 Convolutional Neural Networks

Convolutional neural networks (CNNs) are multi-layer feed-forward networks designed to recognize features in 2-dimensional image data and whose architecture is inspired by a study of neurobiological signal processing in cats' visual cortex performed by Hubel and Wiesel [1968]. The basic idea of the CNN was inspired by the biological concept of receptive fields that act as detectors that are sensitive to certain types of stimuli, for example, edges. They are found across the visual field and overlap each other.

This biological function can be approximated in computers using the convolution operation [Marr and Hildreth, 1980]. In image processing images can be filtered using convolution to produce different visible effects. Fig. 2.6 shows a convolutional filter detecting horizontal edges, functioning in a similar fashion to a receptive field.

The discrete convolution operation between an image f and a filter matrix g is defined as:

$$h[x,y] = f[x,y] * g[x,y] = \sum_n \sum_m f[n,m]g[x-n,y-m] \quad (2.4)$$

The dot product of the filter g and a sub-image of f (with same dimensions as g) centered on coordinates x, y produces the pixel value of h at coordinates x, y

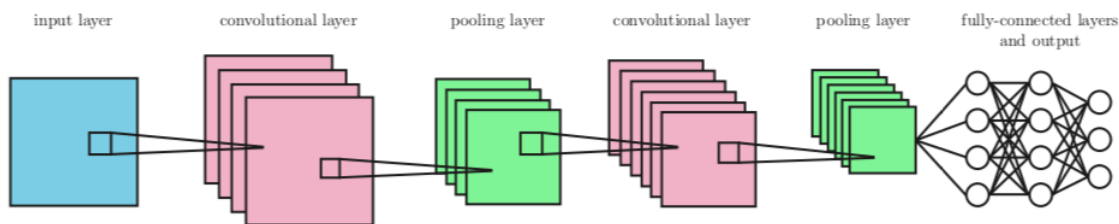


Figure 2.7. An example of a convolutional network [Stenroos et al., 2017].

[Goodfellow et al., 2016]. The size of the receptive field is adjusted by the size of the filter matrix. Aligning the filter successively with every sub-image of f produces the output pixel matrix h . In the case of neural networks, the output matrix is also called a feature map (or an activation map after computing the activation function). Edges need to be treated as a special case. If image f is not padded, the output size decreases slightly with every convolution [Goodfellow et al., 2016].

Convolutional filters are combined to form a convolutional layer of a neural network [Fukushima, 1988]. The matrix values of the filters are treated as neuron parameters and trained using machine learning. The convolution operation replaces the multiplication operation of a regular neural network layer. Output of the layer is usually described as a volume and the height and width of the volume depend on the dimensions of the activation map. The depth of the volume depends on the number of filters [Fukushima, 1988].

Since the same filters are used for all parts of the image, the number of free parameters is reduced drastically compared to a fully-connected neural layer [LeCun et al., 1989]. The neurons of the convolutional layer mostly share the same parameters and are only connected to a local region of the input. Parameter sharing resulting from convolution ensures translation invariance [Fukushima, 1988].

Successive convolutional layers (often combined with other types of layers, such as pooling described below) form a *convolutional neural network* (CNN). An example of CNN is shown in Fig. 2.7. Layers closer to the input are reported to learn to recognize low-level features of the image, such as edges and corners, and the layers closer to the output learn to combine these features to recognize more meaningful shapes [Fukushima, 1988].

2.3.1 Pooling

In order to make the network less complex for classification, the activation map size is decreased in the deep end of the network. The deep layers of the network require

less information about exact spatial locations of features, but require more filters to recognize multiple high-level patterns [Goodfellow et al., 2016]. With the reduction of the height and width of the data volume, the depth of the data volume can be increased while the computation time is kept at a reasonable level.

Reducing the data volume size can be achieved in two manners. One way is to include a pooling layer after a convolutional layer [Nasrabadi, 2007]. The layer effectively downsamples the activation maps. Pooling has the added effect of making the resulting network more translation invariant by forcing the detectors to be less precise, however, it can destroy information about spatial relationships between subparts of patterns. A common example of pooling method is max-pooling. Max-pooling simply outputs the maximum value within a rectangular neighbourhood of the activation map [Goodfellow et al., 2016].

2.3.2 Additional layers

Convolutional layers typically include a non-linear activation function, such as a rectified linear activation function. Activations are sometimes described as a separate layer between the convolutional layer and the pooling layer.

Some systems, such as the one presented by Simonyan and Zisserman [2014], also implement a layer called local response normalization, which is used as a regularization technique. Local response normalization mimics a function of biological neurons called lateral inhibition, which causes excited neurons to decrease the activity of neighbouring neurons. However, other regularization techniques are currently more popular and are discussed below.

The final hidden layers of a CNN are typically fully-connected layers [Nasrabadi, 2007]. A fully-connected layer can capture relationships that parameter-sharing convolutional layers cannot. However, a fully connected layer requires a sufficiently small data volume size in order to be practical. Pooling and stride settings can be used to reduce the size of the data volume that reaches the fully-connected layers. A convolutional network that does not include any fully-connected layers, is called a *fully convolutional network* (FCN) [Ren et al., 2015].

If the network is used for classification, it usually includes a softmax output layer [Nasrabadi, 2007]. The activations of the top most layers can be used as a feature representation of an image with the convolutional network being used as a large feature detector [LeCun et al., 1989].

2.3.3 Regularization

Regularization refers to methods that are used to reduce overfitting by introducing additional constraints or information to the machine learning system [Goodfellow et al., 2016]. A classical way of using regularization in neural networks is adding a penalty term to the objective/loss function that penalizes certain types of weights. The parameter sharing feature of convolutional networks is another example of regularization.

Among the several regularization techniques for deep neural networks, one of the most popular ones is the dropout method [Srivastava et al., 2014], which attempts to reduce the co-adaptation of neurons by randomly dropping out neurons during training, meaning that a slightly different neural network is used for each training sample or mini-batch. The basic idea is that each neuron in the network has certain probability to be deactivated during one iteration. This potential for deactivation is evaluated in every iteration, to ensure that network has different architecture every time. Deactivated means that it will not propagate any signal through. This forces individual neurons to learn features that are less dependent on its surrounding and causes the system not to depend too much on any single neuron or connection and provides an effective yet computationally inexpensive way of implementing regularization [Goodfellow et al., 2016]. In convolutional networks, dropout is typically used in the final fully-connected layers [Simonyan and Zisserman, 2014].

Overfitting may also be reduced with an increase in the amount of training data. When it is not possible to acquire more actual samples, data augmentation is used to generate more samples from the existing data. For classification using convolutional networks, this can be achieved by computing transformations on the input images that do not alter the perceived object classes, yet provide additional challenge to the system. The images can be, for example, flipped, rotated or subsampled with different crops and scales or noise can be added [Goodfellow et al., 2016].

2.3.4 Development

Convolutional neural networks were one of the first successful deep neural networks. The Neocognitron, developed by Fukushima in 1980s, provided a neural network model for translation-invariant object recognition, inspired by biology [Fukushima, 1988]. LeCun et al. [1989] combined this method with a learning algorithm, i.e. back-propagation. These early solutions were mostly used for handwritten character recognition.

After providing some promising results, the neural network methods faded in prominence and were mostly replaced by support vector machines [Girshick et al.,

2014b]. Then, in 2012, Krizhevsky et al. [2012b] achieved excellent results on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset by combining Le Cun's method with recent fine-tuning methods for deep learning. These results popularized CNNs and led to the development of new powerful object detection methods [Girshick et al., 2014b].

Chapter 3

Methodology

This chapter details how the proposed method aggregates contextual information into the representation generated through image superpixels. It exploits several layers in convolutional networks to compute deep contextual features from superpixels by keeping the mapping between the pixels within each of them and the feature maps across the network.

3.1 Overview

Most methods in the literature exploit either only a single level of context or a combination of local and global cues. In order to exploit a range of contextual levels, from the superpixel itself to the entire image patch containing it, the method proposed in [Santana et al. \[2017\]](#) is used. The first step consists in segmenting an image into superpixels in order to obtain homogeneous units. Given a superpixel, in the second step features concerning three contextual areas defined by different groups of pixels are separately extracted: (1) inside the superpixel; (2) a small contextual area around it; and (3) a larger contextual area. Small and larger contextual areas are limited by boxes centered in the superpixel and consider both internal and external pixels from it. As a third step we decided to add to the method, different pooling strategies are tested to assess the impact that such change may cause. The final representation consists in the concatenation of the extracted features.

Despite the simplicity of the proposed composition of contextual features, its strength lies on the exploitation of intermediate contextual levels, which are left out by most methods. A general overview of this representation is shown in Fig. 3.1.

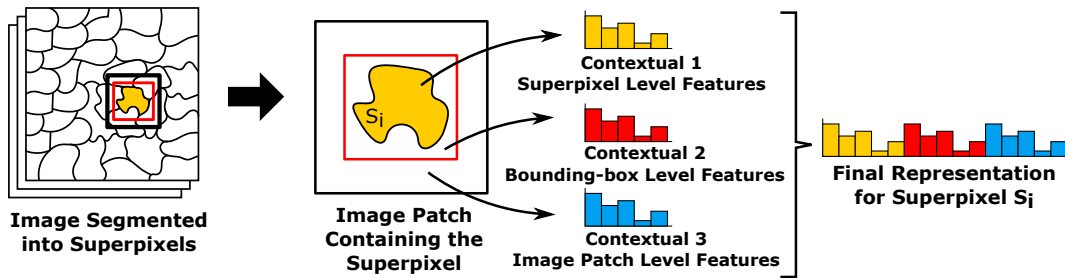


Figure 3.1. The proposed representation to exploit all contextual levels ranging from the superpixel itself to an entire image patch containing it. Given a target superpixel s_i , its final representation is the concatenation of the ℓ_2 -normalized features extracted from s_i , a small rectangular contextual area surrounding s_i and a large contextual area [Santana et al., 2017].

3.2 Segmentation

In the proposed method instead of using the sliding-window technique for region proposal for object detection, images are segmented into superpixels using SLICO, an adaptive version of the Simple Linear Iterative Clustering (SLIC) algorithm. It can be considered an adaptation of k -means for superpixel generation and provides several distinct advantages over previous methods. SLIC generates superpixels with high perceptual homogeneity, uniformity in size and shape, good accuracy and boundary recall properties [Achanta et al., 2012], has a high time and memory efficiency, can cope with both colour and grayscale imagery and easily generalizes to multiple spectral bands [Jia et al., 2017] and works as described in Section 2.2.

After the segmentation of the whole image is obtained, each superpixel is taken as a candidate region for object detection. A 227×227 patch is extracted around the center of such superpixel and the next steps of the detection process are carried on in this patch, as it is also segmented with SLICO and the feature extraction described in the next section is performed, as depicted in Fig. 3.2. It is worth noticing that these two segmentations are performed in images with extremely different scales, as the first segmentation occurs over the whole image and the second occurs on small patches, and naturally the ideal number of superpixels for each of these cases will radically differ. The use of two segmentations differs from the original single segmentation approach, and such change was proposed based on the hypothesis that segmenting a patch comprising a local region would yield better results.

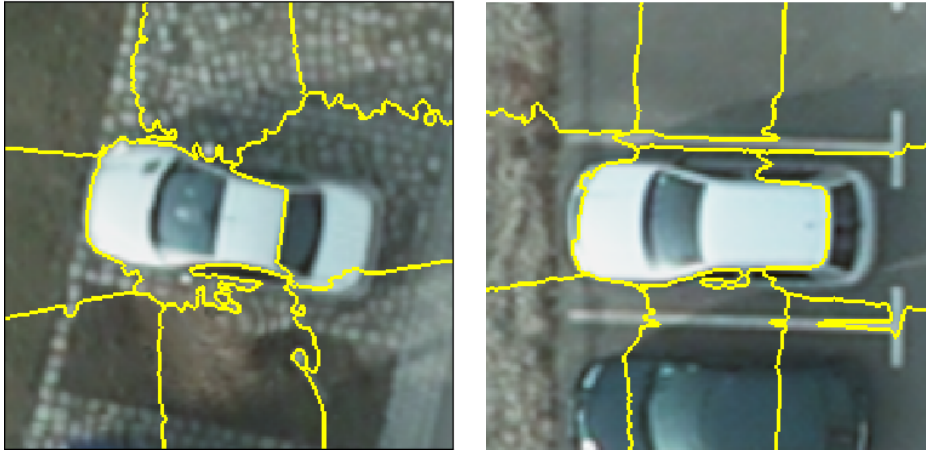


Figure 3.2. Examples of patches containing cars after the performance of the second segmentation of the proposed method.

3.3 Multi-context feature extraction

CNNs are used to extract the proposed contextual features as deep CNN architectures trained on huge datasets of numerous categories can be transferred to new domains by employing them as feature extractors on other tasks including recognition and retrieval, providing better performance than handcrafted features [Penatti et al., 2015].

The strength of such composition of contextual features lies on the fact that it exploits the way in which CNNs encode contextual representation: each layer learns filters that enhance different visual semantic levels [Mostajabi et al., 2015]. The first convolutional layers emphasize low-level properties like borders, color, texture, patterns and other properties from a small set of pixels, which may represent parts of roofs, streets, cars, and small objects. The last layers are able to incorporate entire objects and its spatial relationship.

Nevertheless, an existing challenge concerning the feature extraction from arbitrarily shaped regions is that the main algorithms are designed for rectangular or squared image patches [dos Santos et al., 2012] and convolutional networks also require square images/patches as input due to their characteristic architecture based on convolutions.

In order to overcome the aforementioned issue, the approach proposed by Mostajabi et al. [2015] for computer vision applications was applied. For creating a feature representation for a given region s_i , a square box around it called image patch I is initially defined. The next step is to use a pre-trained convolutional network to create feature maps in different layers. Regardless of the chosen network, a convolutional layer with k filters produces k feature maps stacked so that there exists a k -dimensional

feature vector associated with each point of the feature maps stack along the width and height dimensions. The major novelty introduced by the authors is that such maps are average pooled over the superpixel or region s_i , generating just one k -dimensional feature vector to represent s_i .

Since the pooling step outputs a single vector for the whole superpixel, it allows the proposed method to make use of deep features which are intrinsically rich in semantic context, besides the spatial context already aggregated by the method. Therefore, the final representation generated encodes both semantic and spatial context for each superpixel s_i . However, due to the pooling process and strides larger than one that may be used for the convolutions, the first layers usually produce feature maps of lower resolution compared to the original image patch I . Since I and the segmentation which delineated s_i remain with the same initial resolution, the mapping between each pixel of I (and consequently the pixels within s_i) and each point of the stack of feature maps output by each layer is lost as I is forwarded across the network. Thus, bilinear interpolation is necessary to rescale the feature maps and allow for feature extraction from the original patch I , as shown in Fig. 3.3.

The proposed approach was validated using the AlexNet [Krizhevsky et al., 2012b] convolutional network. Fig. 3.4 illustrates the proposed strategy for deep contextual feature extraction using AlexNet. The first level $\phi_1(s_i)$ is responsible for encoding the features of the superpixel s_i itself. These features are extracted from the first convolutional layer of the ConvNet and are mainly responsible for capturing color, texture, patterns and other properties from a small set of pixels, which may represent parts of roofs, streets, cars, and small objects. The second level $\phi_5(s_b)$ uses a larger context that should yield more information about the region s_b around the superpixel, giving cues about its neighborhood and helping in its classification. Intuitively, the features computed at this level (which are not available in the previous ones) tend to be more complex and give more information since they may represent whole buildings, streets, cars as well as interactions among them. The final level of context $\phi_{fc2}(I)$ represents the entire input image patch I . These features encode an even larger area that represents the whole scene of the input image patch I , including relationships between buildings, cars, streets, etc. Features from this layer are useful for global support of local labeling decisions, e.g., lots of green in an image supports labeling a forest or a park. This layer then helps to determine the presence of categories in the scene, as it imposes a constraint in the label space by eliminating classes which have no elements within the image patch I . In the proposed method by Santana et al. [2017], features are extracted from a final layer (the last fully connected one). The last level of context $\phi_{fc2}(I)$ is comprised of only features values and not feature maps. At the end

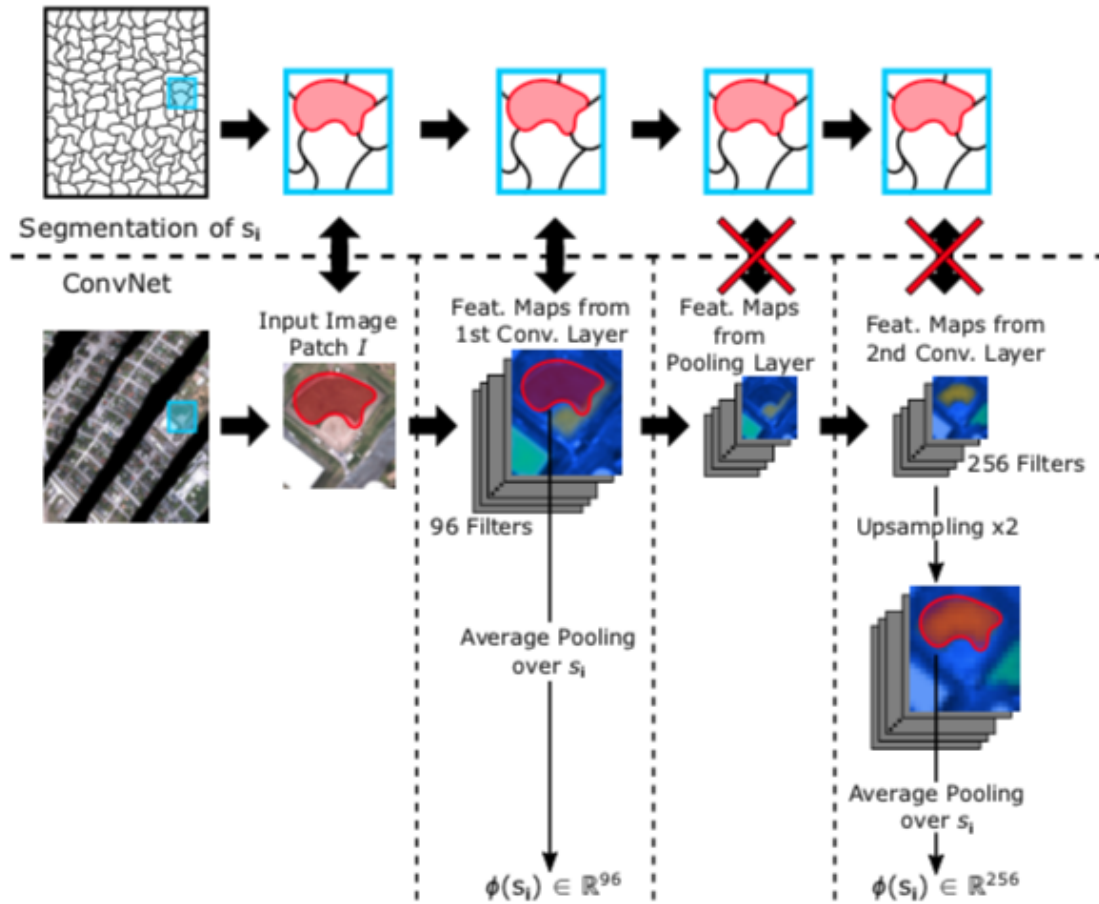


Figure 3.3. Example of the approach proposed by Mostajabi et al. [2015] to extract deep features from the superpixel s_i . It consists in keeping a mapping between each pixel inside s_i and the corresponding points of the feature maps as the image I is forwarded across the network. So it is possible to generate just one k -dimensional feature vector after each convolutional layer by average pooling the k feature maps over s_i . When the resolution of the feature maps is reduced by the stride in pooling and convolutional layers, an upsampling is employed in order to restore their original size and consequently keep the mapping [Santana et al., 2017].

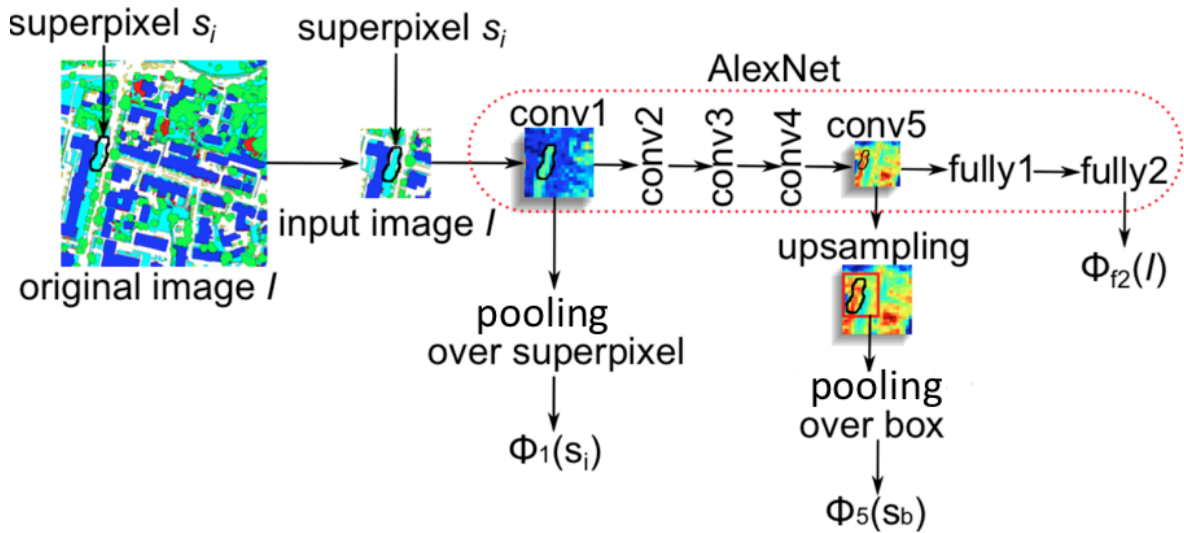


Figure 3.4. The proposed approach for deep contextual feature extraction with AlexNet. Given a superpixel s_i , an image patch I is created centering the superpixel and used as input for the AlexNet. The features are computed considering three levels of context (or three layers): $\phi_1(s_i)$, $\phi_5(s_b)$ and $\phi_{fc2}(I)$. Adapted from [Santana et al., 2017].

of the process, the extracted feature vectors $\phi_1(s_i)$, $\phi_5(s_b)$ and $\phi_{fc2}(I)$ are concatenated for the final representation of the superpixel s_i .

3.4 Pooling strategies

Deep CNNs are composed of several layers of processing — each containing linear as well as non-linear operators — which are jointly learnt in an end-to-end way to solve specific tasks [Farabet et al., 2013].

They are commonly made up of convolutional, normalization, pooling, and fully connected layers. The convolutional layer is the main building block of the CNN, and its parameters consist of a set of learnable filters.

The pooling layer takes small rectangular blocks from the convolutional layer and subsamples them to produce a single output from each block. The literature conveys several ways to perform pooling, such as taking the average, the maximum, or a learned linear combination of the values in the block. This layer allows control of over-fitting and reduces the number of parameters and computation in the network [Ammour et al., 2017].

Usually average or max pooling are employed in this layer, but they both have some drawbacks. As max pooling only considers the maximum element in the pooling

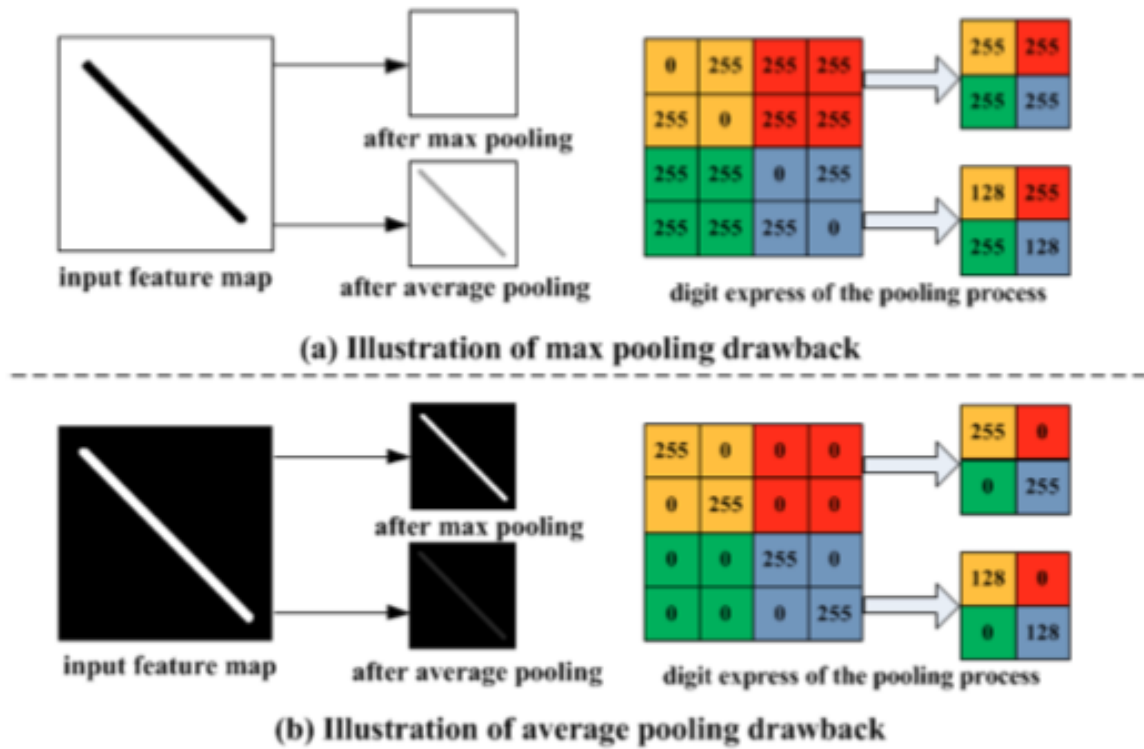


Figure 3.5. Toy example illustrating the drawbacks of max pooling and average pooling [Yu et al., 2014].

region, if most of the elements are of high magnitudes, the distinguishing feature vanishes after max pooling. Average pooling on the other hand, calculates the mean of all the elements within the pooling region taking all low magnitudes into consideration. This causes a reduction in the contrast of the new feature map as show in Fig. 3.5, with a possible worst case scenario happening when many zero elements are present in which the characteristic of the feature map will be largely reduced [Yu et al., 2014]. We investigated the effect of using more than one of such pooling methods when extracting features through CNNs, concatenating the average value, max value and standard deviation from the values contained in the superpixel taken into account in the method described in the previous subsection.

Chapter 4

Experimental Analysis

The first section of this chapter brings a description of the setup of the tested scenario, starting with the datasets and the classifiers employed in the experiments performed, along with the metrics used to analyze the results obtained. The second and third sections display these results and discuss them.

The first results regard the use of different pooling strategies when applying the method to its original scenario of semantic segmentation. The method is then assessed on the task of object detection in two remote sensing images datasets. One of the datasets used was custom built and will be publicly released as of the time when this research will be published.

4.1 Setup

4.1.1 Datasets

4.1.1.1 `grss_dfc_2014`

This dataset is comprised of HSI data covering an urban area near Thetford Mines in Québec, Canada, and it was used in the 2014 IEEE GRSS Data Fusion Contest. It was acquired on May 21st, 2013 by an airborne long-wave infrared hyperspectral imager with 84 channels ranging between 7.8 to 11.5 μm wavelengths. It consists of two different sets of imagery data: 1) a long-wave infrared (LWIR, thermal infrared) hyperspectral image composed of 84 channels with nearly 1 m spatial resolution; and 2) a Very High Resolution (VHR) color image with 3769×4386 pixels in the visible spectrum, composed of many RGB sub-images with spatial resolution of 20 cm and associated with distinct flight-lines [Liao et al., 2015]. Both sets were radiometrically

and geometrically corrected posteriorly. Fig. 4.1 shows samples from the described dataset.

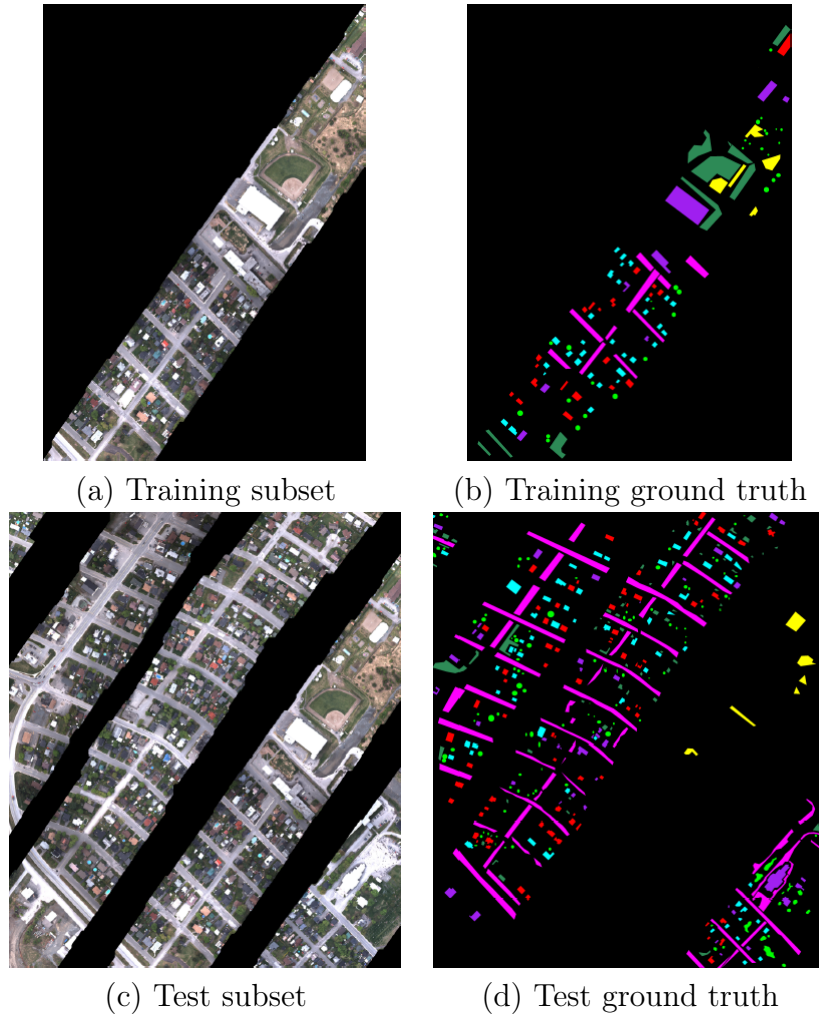


Figure 4.1. Images from `grss_dfc_2014` used in the experiments.

4.1.1.2 ISPRS Potsdam

Two datasets with focus on trees and cars detection were created based on image blocks cropped from the original large-scale aerial images of the ISPRS Potsdam dataset from the 2D Semantic Labeling competition with the aim of assessing the feasibility of the proposed method for object detection.

The ISPRS Potsdam dataset consists of 38 VHR true orthophoto (TOP) image patches of 6000×6000 pixels and corresponding digital surface models (DSMs) obtained through dense image matching. Both types of data were acquired using a ground sampling distance of 5 cm over Potsdam, Germany by BSF Swissphoto, that made the

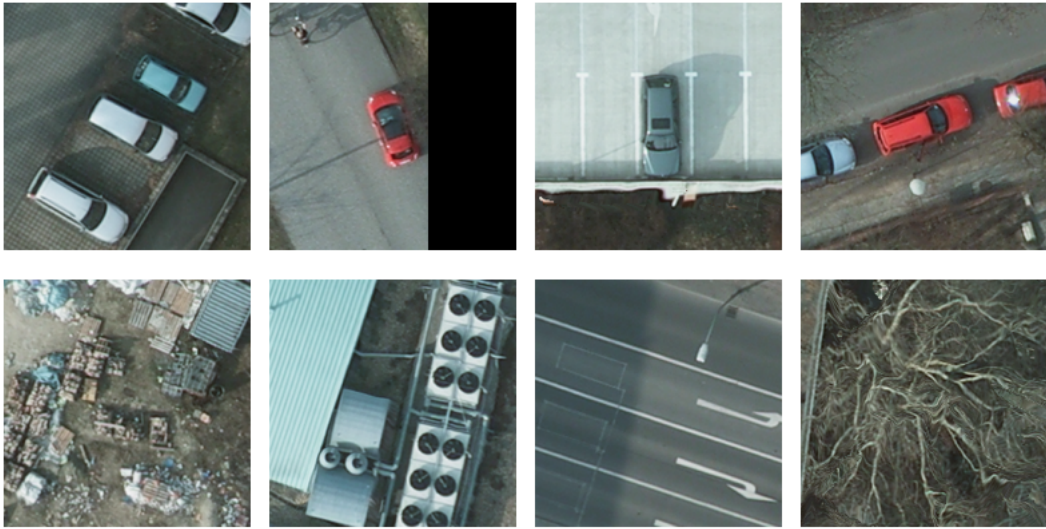


Figure 4.2. Samples from the dataset created for the task of cars detection. Top row: positive samples. Bottom row: negative samples.



Figure 4.3. Samples from the dataset created for the task of trees detection. Top row: positive samples. Bottom row: negative samples.

data available for the Semantic Labeling Contest of the ISPRS. Although DSMs may be useful to improve classification results, only RGB images were used in this work.

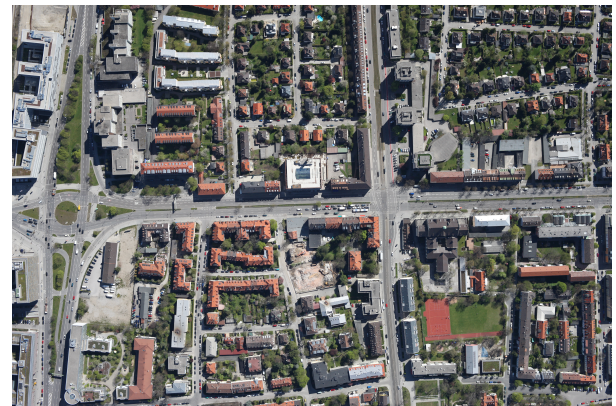
Patches with size 227×227 of positive and negative samples were extracted from all the images in the original ISPRS Potsdam dataset. The tree detection dataset created contains 5818 positive and 5818 negative samples, while the cars dataset contains 5693 positive samples and 5697 negative samples. Fig. 4.2 depict samples of patches extracted for the creation of the cars detection dataset, showing positive and negative samples; while Fig. 4.3 shows samples for the tree detection scenario.

4.1.1.3 Munich Vehicle Dataset

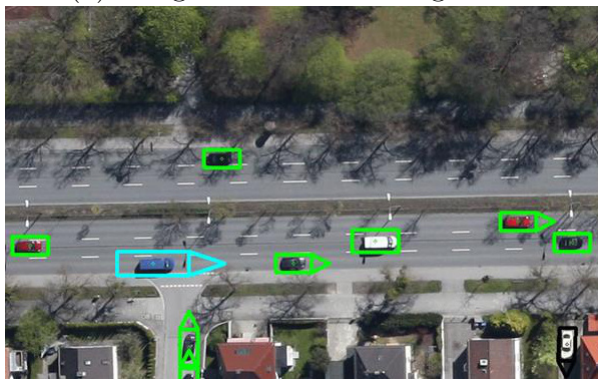
After promising preliminary results (that are described in Subsection 4.2.1) the proposed method is further evaluated on a publicly available vehicle dataset collected over the city of Munich, Germany. It is comprised of 20 aerial images captured from an airplane by a Canon Eos 1Ds Mark III camera with a resolution of 5616 x 3744 pixels, 50 mm focal length and stored in JPEG format [Liu and Mattyus, 2015]. The images were taken from a height of 1000 m and the ground sampling distance is of approximately 13 cm [Leitloff et al., 2014]. The first ten images compose a training subset while the test subset is composed of the remaining ten images. Fig. 4.4 shows samples from the described dataset.



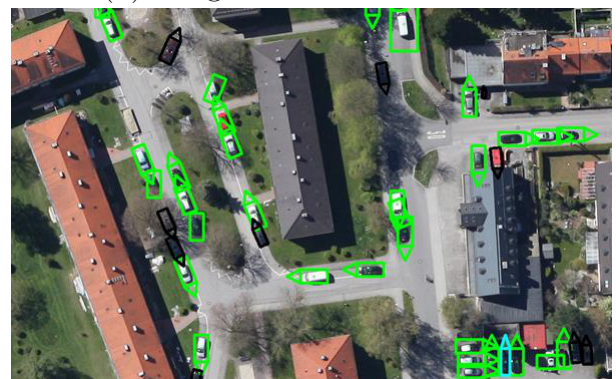
(a) Image from the training subset



(b) Image from the test subset



(c) Zoom-in in showing labeled samples



(d) Zoom-in in showing labeled samples

Figure 4.4. Images from the Munich Vehicle Dataset.

4.1.2 Classifiers and Training Protocol

SVMs, KNNs, decision trees and ensembles [Alpaydin, 2009] were employed to perform the tests reported in the next chapter. Table 4.1, Table 4.2, Table 4.3 and Table 4.4 detail the set of classifiers used and their respective parameters.

Tree type	Maximum number of splits	Split criterion	Surrogate decision splits
Coarse Tree	4	Gini's diversity index	Off
Medium Tree	20	Gini's diversity index	Off
Fine Tree	100	Gini's diversity index	Off

Table 4.1. Configurations of decision trees used as classifiers.

SVM Kernel	Box constraint level	Kernel scale mode	Multiclass method	Standardize data
Linear	1	Auto	One-vs-One	Yes
Quadratic	1	Auto	One-vs-One	Yes
Cubic	1	Auto	One-vs-One	Yes
Fine Gaussian	1	Manual (0.5)	One-vs-One	Yes
Medium Gaussian	1	Manual (2)	One-vs-One	Yes
Coarse Gaussian	1	Manual (8)	One-vs-One	Yes

Table 4.2. Configurations of SVMs used as classifiers.

KNN type	Number of neighbors	Distance metric	Distance weight	Standardize data
Fine KNN	1	Euclidean	Equal	Yes
Medium KNN	10	Euclidean	Equal	Yes
Coarse KNN	100	Euclidean	Equal	Yes
Cosine KNN	10	Cosine	Equal	Yes
Cubic KNN	10	Minkowski	Equal	Yes
Weighted KNN	10	Euclidean	Squared inverse	Yes

Table 4.3. Configurations of KNNs used as classifiers.

Type	Ensemble method	Learner type	Maximum # of splits	# of learners	Learning rate
Boosted Trees	Adaboost	Decision tree	20	30	0.1
Bagged Trees	Bag	Decision tree	149	30	0.1
Subspace Discriminant	Subspace	Discriminant	-	30	0.1
Subspace KNN	Subspace	Nearest neighbor	-	30	0.1
RUSBoosted Trees	RUSBoost	Decision tree	20	30	0.1

Table 4.4. Configurations of Ensembles used as classifiers.

Five-fold cross validation was employed for the datasets that do not contain a test subset. The average accuracy is calculated and the confidence intervals are calculated using the Student's t -distribution with 4 d.f. and 95% of confidence.

4.1.3 Metrics

For the validation of the evaluated methods, the following metrics [Michalski et al., 2013] shall be used:

1. *Recall*: number of retrieved relevant items divided by the total number of relevant items;
2. *Precision*: number of retrieved and relevant items divided by the total number of retrieved items;
3. *F-Measure*: a measure of the accuracy of a test. This metric takes into account both precision and recall in order to compute a score p , that is the number of correct positive results divided by the number of all positive results; and r , the number of correct positive results divided by the number of positive results that should have been returned. It may be interpreted as a weighted average of the precision and recall, where an F-Measure reaches its best value at 1 and worst at 0.

4.2 Evaluation in semantic segmentation task

4.2.1 Results on grss_dfc_2014 Dataset

In this section, the experiments carried out and the results achieved on the grss_dfc_2014 dataset are presented. A brief discussion follows the description of the experiments.

The first experiment performed is an analysis of the contribution of the convolutional network layers to the final result. To assess the importance of features from each layer, Santana et al. [2017] trained classifiers using all possible concatenations of the three layers of the network used in the proposed method (first, fifth and last fully connected), which were also used by Mostajabi et al. [2015].

Table 4.5 shows the results of the concatenations that yielded the best results using the XGBoost classifier while Table 4.6 shows the accuracy obtained with the modifications proposed in this work, that is, employing different pooling strategies besides the usual average pooling.

Layers	Method	Accuracy
Conv1	[Santana et al., 2017]	93.84%
	[Mostajabi et al., 2015]	93.84%
Conv 5	[Santana et al., 2017]	91.17%
	[Mostajabi et al., 2015]	93.52%
fc8	[Santana et al., 2017]	88.76%
	[Mostajabi et al., 2015]	88.76%
Conv1 & 5	[Santana et al., 2017]	94.77%
	[Mostajabi et al., 2015]	95.51%
Conv5 and fc8	[Santana et al., 2017]	90.45%
	[Mostajabi et al., 2015]	94.05%
Conv1, 5 and fc8	[Santana et al., 2017]	95.63%
	[Mostajabi et al., 2015]	95.99%

Table 4.5. Layer importance analysis in grss_dfc_2014 dataset using the XG-Boost classifier performed in [Santana et al., 2017].

Layers	Original Accuracy	Modified Accuracy
Conv1	93.84% [Santana et al., 2017]	Max pooling 95.68%
	93.84% [Mostajabi et al., 2015]	Standard deviation 95.78%
Conv 1 & 5	94.77% [Santana et al., 2017]	Max pooling 95.72%
	95.51% [Mostajabi et al., 2015]	Standard deviation 96.42%
Conv 1,5 and fc8	95.63% [Santana et al., 2017]	Max pooling 96.15%
	95.99% [Mostajabi et al., 2015]	Standard deviation 96.22%

Table 4.6. Layer importance analysis in grss_dfc_2014 dataset by using different pooling strategies.

The observation made by Santana et al. [2017] that the composition of the first and last layers presents the highest accuracy levels holds, followed by Conv1 and Conv5 and then by Conv5 and fc8. Such behavior is expected, once Conv1 and fc8 are complementary in terms of abstraction level (low and high, respectively) and contextual area (local and global, respectively).

Employing max pooling and standard deviation as pooling methods yielded achievements in the accuracy of all tested scenarios, which shows that the three pooling strategies combined might complement each other’s drawbacks as described in Section 3.4.

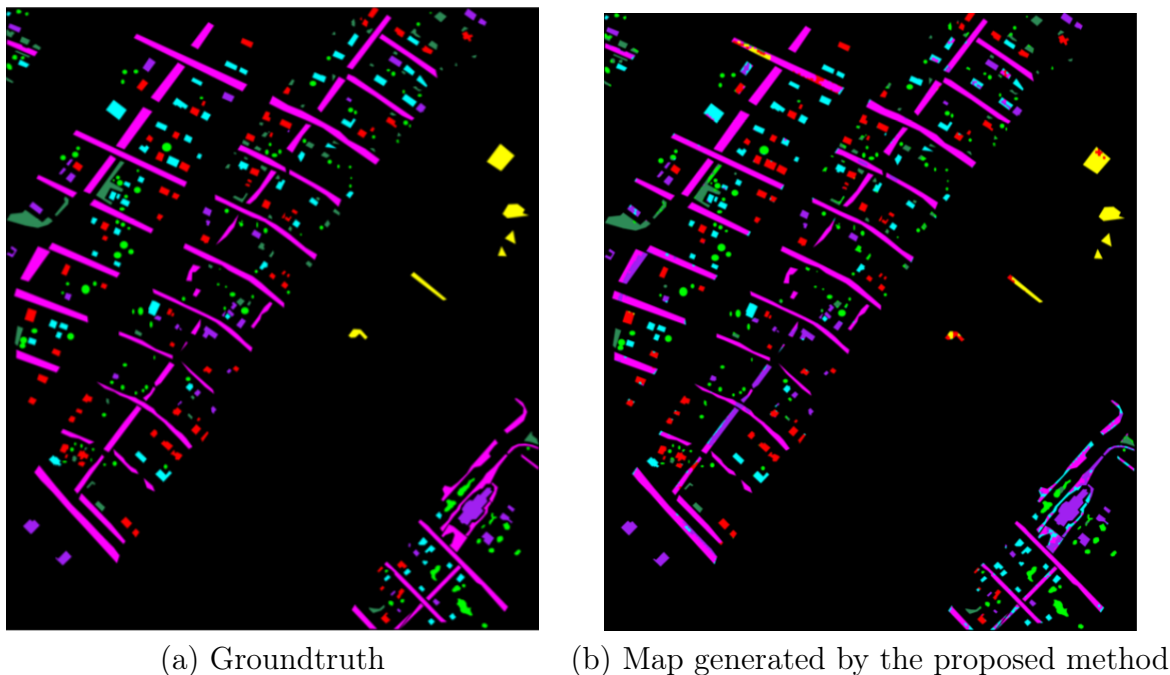


Figure 4.5. Groundtruth of the test image along with the map generated by the proposed method with XGBoost classifier using layers Conv1, Conv5 and fc8.

4.3 Evaluation in detection task

4.3.1 Results on Potsdam dataset

Table 4.7 and Table 4.8 detail the average accuracy levels obtained in the tests performed on the datasets created for cars and trees detection from the Potsdam dataset. Different combinations of convolutional layers employed were tested, as well as three pooling methods: average pooling, max pooling and standard deviation. Since this dataset was specifically built with image patches that are candidate regions for object detection with the exact size used by the proposed method, the first segmentation of the method is not performed, therefore the aforementioned tables only show the number of superpixels evaluated for the second segmentation. Due to space constraints, only the results for the best classifiers are displayed.

The concatenation of the activations from different pooling methods lead to accuracy levels of more than 99%. Concatenating more layers slowly improved this accuracy towards the maximum levels attained. As such improvement is not so substantial after a certain threshold, it may be the case that some layers might be ignored for speeding up computation time if the application scenario has time restrictions.

The results concerning tree detection were slightly inferior to those of car detection and that may be mainly due to the fact that the images of this dataset were

Classifier/ Layers	# Superpixels 2nd segmentation	Fine KNN	Logistic Regression	Quad SVM	Sub Disc
fc8	-	97.85%	97.36%	97.67%	97.68%
avg conv1	6	71.73%	83.64%	82.97%	71.65%
	12	75.62%	83.61%	82.98%	75.51%
	18	94.44%	95.49%	95.25%	94.45%
	24	81.97%	92.39%	92.10%	82.42%
avg conv5	6	97.62%	98.42%	98.49%	97.00%
	12	95.69%	97.50%	97.69%	94.88%
	18	96.16%	97.67%	97.57%	95.69%
	24	97.83%	98.68%	98.56%	97.60%
avg max std conv 1	6	95.54%	97.40%	97.69%	95.77%
	12	93.71%	94.79%	95.26%	93.55%
	18	97.42%	98.18%	98.03%	97.56%
	24	96.08%	97.28%	97.32%	95.95%
avg max std conv 5	6	99.19%	99.16%	99.19%	99.03%
	12	97.47%	98.18%	98.06%	97.54%
	18	97.42%	98.18%	98.03%	97.56%
	24	98.68%	99.13%	99.06%	98.68%
avg max conv1 fc8	6	99.40%	98.86%	99.20%	99.31%
	12	98.44%	98.28%	98.54%	98.38%
	18	98.49%	98.43%	98.61%	98.54%
	24	98.93%	98.96%	99.04%	98.93%
avg max std conv5 fc8	6	99.46%	99.29%	99.49%	99.53%
	12	98.49%	98.41%	98.40%	98.44%
	18	98.42%	98.59%	98.72%	98.52%
	24	99.13%	99.17%	99.21%	99.14%
avg max conv1,5 fc8	6	99.50%	99.22%	99.47%	99.58%
	12	98.33%	98.47%	98.61%	98.61%
	18	98.65%	98.68%	98.75%	98.73%
	24	99.11%	99.27%	99.29%	99.08%
avg max std conv1,5 fc8	6	99.49%	99.30%	99.54%	99.59%
	12	98.61%	98.50%	98.65%	98.68%
	18	98.64%	98.71%	98.82%	98.85%
	24	99.24%	99.30%	99.25%	99.25%

Table 4.7. Average accuracy obtained for cars detection on 4 different classifiers with the layers of AlexNet used as features and the pooling methods employed.

obtained in the fall season over a city located in Germany, so most trees had lost all its leaves at the time. That makes their detection harder, as the contrast between the leafless branches and the ground covered by dried leaves decreases, making it hard even for humans to discern whether an object is a leafless tree or a bush.

Classifier/ Layers	# Superpixels 2nd segmentation	Coarse Tree	Logistic Regression	Quad SVM
fc8	-	95.34%	95.18%	95.17%
avg conv1	6	90.04%	92.16%	91.60%
	12	87.63%	89.22%	88.82%
	18	88.46%	89.30%	89.04%
	24	89.46%	89.95%	90.12%
avg conv 5	6	97.98%	98.04%	98.02%
	12	94.67%	94.60%	94.65%
	18	94.59%	94.98%	95.05%
	24	96.54%	96.18%	96.22%
avg max std conv 1	6	94.67%	95.27%	95.56%
	12	91.17%	91.49%	91.83%
	18	91.87%	91.92%	91.99%
	24	95.05%	94.92%	95.47%
avg max std conv 5	6	98.81%	98.71%	98.81%
	12	95.34%	95.30%	95.26%
	18	95.15%	95.21%	94.87%
	24	97.74%	97.46%	97.70%
avg max conv1 fc8	6	98.26%	98.11%	98.43%
	12	96.12%	95.92%	96.14%
	18	96.41%	96.30%	96.33%
	24	97.42%	97.01%	97.34%
avg max std conv5 fc8	6	99.02%	98.85%	99.09%
	12	96.27%	96.10%	96.33%
	18	96.51%	96.19%	96.34%
	24	98.30%	98.05%	98.25%
avg max conv1,5 fc8	6	98.97%	98.81%	99.01%
	12	96.38%	96.18%	96.32%
	18	96.54%	96.53%	96.4%
	24	98.38%	98.26%	98.44%
avg max std conv1,5 fc8	6	99.07%	98.89%	99.03%
	12	96.52%	96.36%	96.64%
	18	96.59%	96.45%	96.45%
	24	98.39%	98.14%	98.34%

Table 4.8. Average accuracy obtained for trees detection on 3 different classifiers with the layers of AlexNet used as features and the pooling methods employed.

4.3.2 Results on Munich Vehicle Dataset

The proposed method was further tested on the publicly available Munich dataset [Leitloff et al., 2014]. We chose to use the concatenation of the three pooling methods and all three layers employed in the preliminary experiments whose results were de-

scribed above. We also empirically varied two parameters: the number of superpixels employed in the first and second segmentations performed in the proposed method. The results obtained are detailed in Table 4.9 and are comparable to results reported in the literature which are shown in Table 4.10.

#Superpixels 2nd segmentation	#Superpixels 1st segmentation	TP	FP	Recall	Precision	F-measure
36SP	20K	4064	5241	68.97%	43.68%	0.53
	30K	4256	8620	72.23%	33.05%	0.43
	40K	4445	12632	75.44%	26.03%	0.38
	50K	4508	16561	76.51%	21.40%	0.33
	60K	4602	20249	78.11%	18.52%	0.29
	70K	4657	23942	79.04%	16.28%	0.27
	80K	4722	27911	80.14%	14.47%	0.24
48SP	20K	4392	4136	74.54%	51.50%	0.60
	30K	4632	7386	78.62%	38.54%	0.51
	40K	4822	11132	81.84%	30.22%	0.44
	50K	4949	14713	84.00%	25.17%	0.38
	60K	5003	18300	84.91%	21.47%	0.34
	70K	5061	21503	85.90%	19.05%	0.31
	80K	5098	25254	86.52%	16.80%	0.28
60SP	20K	4419	3949	75.00%	52.81%	0.61
	30K	4839	8315	82.13%	36.79%	0.50
	40K	4985	10977	84.61%	31.23%	0.45
	50K	5051	14522	85.73%	25.81%	0.39
	60K	5152	22397	87.44%	18.70%	0.30
	70K	5061	21503	85.90%	19.05%	0.31
	80K	5196	24940	88.19%	17.24%	0.28

Table 4.9. Results obtained on the Munich Vehicle Dataset.

Method	TP	FP	Recall	Precision	F-measure
[Liu and Mattyus, 2015]	4085	619	69.63%	86.8%	0.77
ACF detector	3078	4062	52.24%	43.31%	0.47
ACF+fast R-CNN	2583	1540	43.84%	62.65%	0.52
SS+fast R-CNN	3287	15012	55.79%	17.96%	0.27
Faster R-CNN	4050	503	68.74%	88.95%	0.78
AVPN_basic	4454	729	75.59%	85.93%	0.80
AVPN_basic+fast R-CNN	4403	384	74.73%	91.98%	0.82
AVPN_large	4538	630	77.02%	87.81%	0.82

Table 4.10. Performance comparison between different methods as reported in [Deng et al., 2017].

The effects of the number of samples used to train the classifiers is depicted in Fig. 4.6, which shows the maps generated by a Cubic SVM for the test image 0110 from the Munich Vehicle dataset (the original image and its groundtruth are shown in Fig. 4.8 and Fig. 4.7). The number of superpixels used for the two segmentations performed in the proposed method were empirically tested. The best results were achieved when the range of the second segmentation varied from 36 to 60 superpixels. We believe that this parameter is directly related to the resolution and focal length of the pictures analyzed, as the size of the car in the patch extracted depends on such settings, for a small number of superpixels will fail to correctly delineate a car from the background, while too many superpixels will end up dividing a car in more than one part.

The number of superpixels of the first segmentation varied in our experiments from 20 K to 80 K. This parameter is also related to the image resolution as it will determine the number of individual units of the image to be analyzed. From the obtained results we could notice that the greater the number of superpixels used in the first segmentation, the greater the number of true positives obtained by the proposed method. However, the increase in true positives comes along with a great spike in the number of false positives, which leads to a decrease in the precision levels and also the F-measure value.

The visual effects of varying these parameters can be visualized in Fig. 4.9, Fig. 4.10 and Fig. 4.11, which depict the resulting maps generated by the proposed method for one of the images from the test portion of the Munich Vehicle Dataset. Despite the decrease in the precision and F-measure values as the number of superpixels increases, the resulting maps look sharper as this parameter increases, with more well delineated units and less merged blobs.

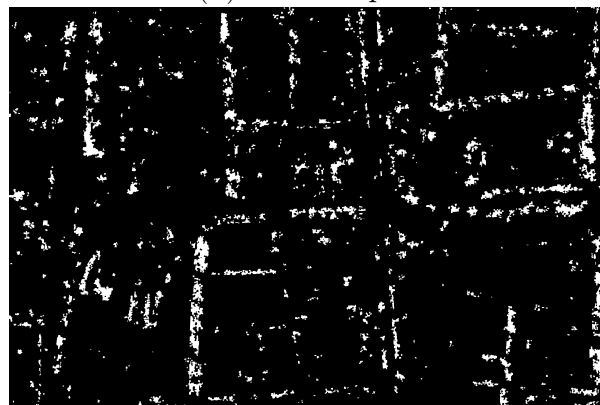
The maps generated with 20 K and 80 K superpixels for all 10 images of the test subset are shown in the next ten Figures along with the respective groundtruths.



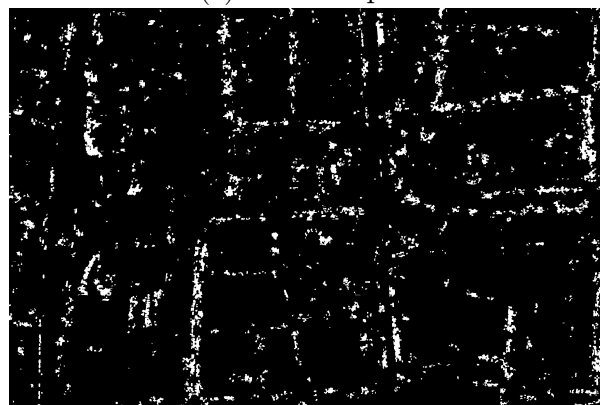
(a) 5K samples



(b) 10K samples



(c) 30K samples



(d) 50K samples

Figure 4.6. Maps generated for image 0110 from the test subset from the Munich Vehicle Detection dataset, with the number of samples used for training the classifier ranging from 5K to 50K.



Figure 4.7. Image 0110 from the test subset of the Munich Vehicle Detection dataset.



Figure 4.8. Groundtruth map of image 0110 from the test subset of the Munich Vehicle Detection dataset.

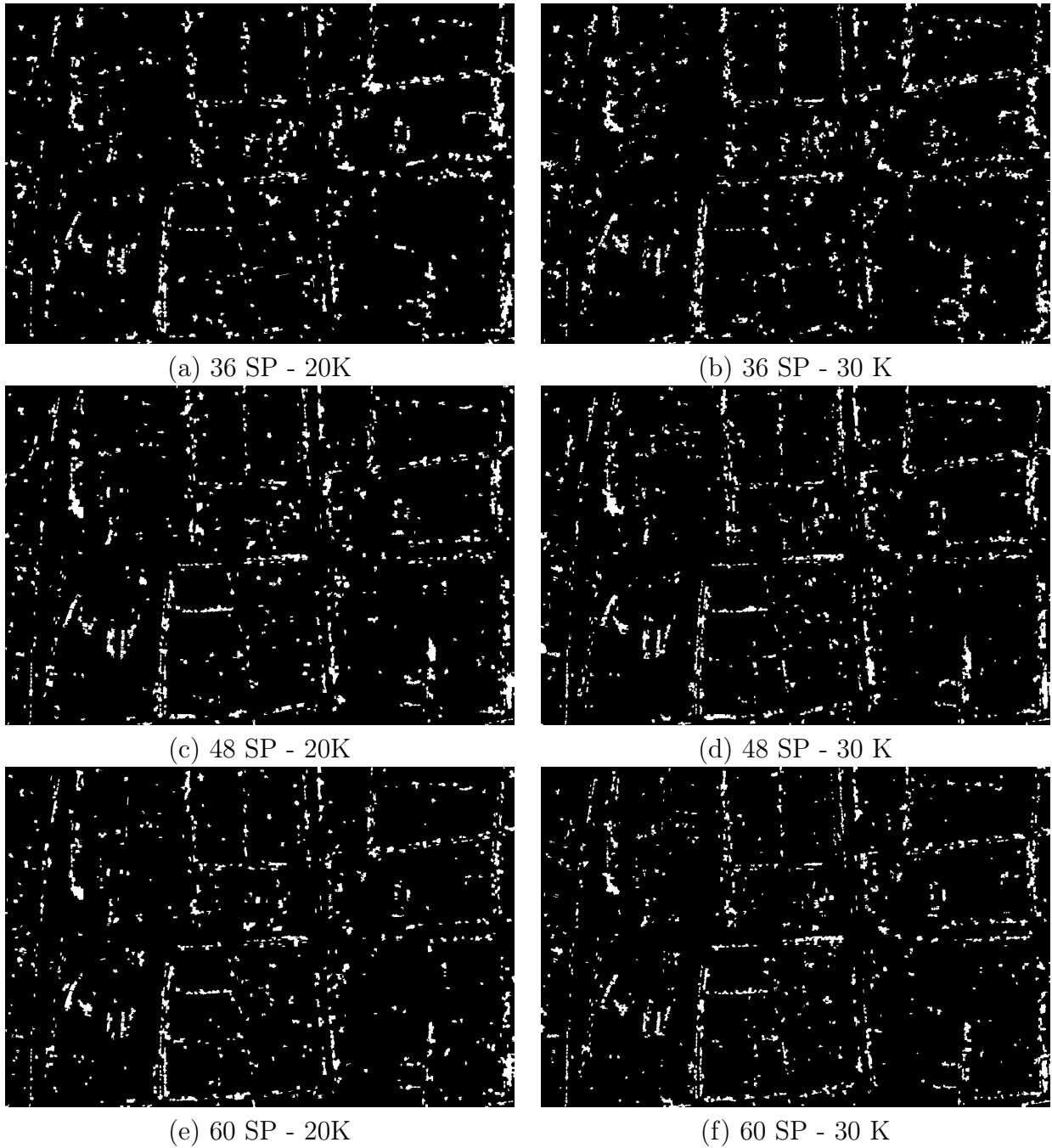


Figure 4.9. Map generated by the proposed method for image 0110 from the test subset from the Munich Vehicle Detection dataset, with the number of superpixels used for segmenting the entire image ranging from 20K to 30K and each selected frame segmented in 36, 48 and 60 superpixels.

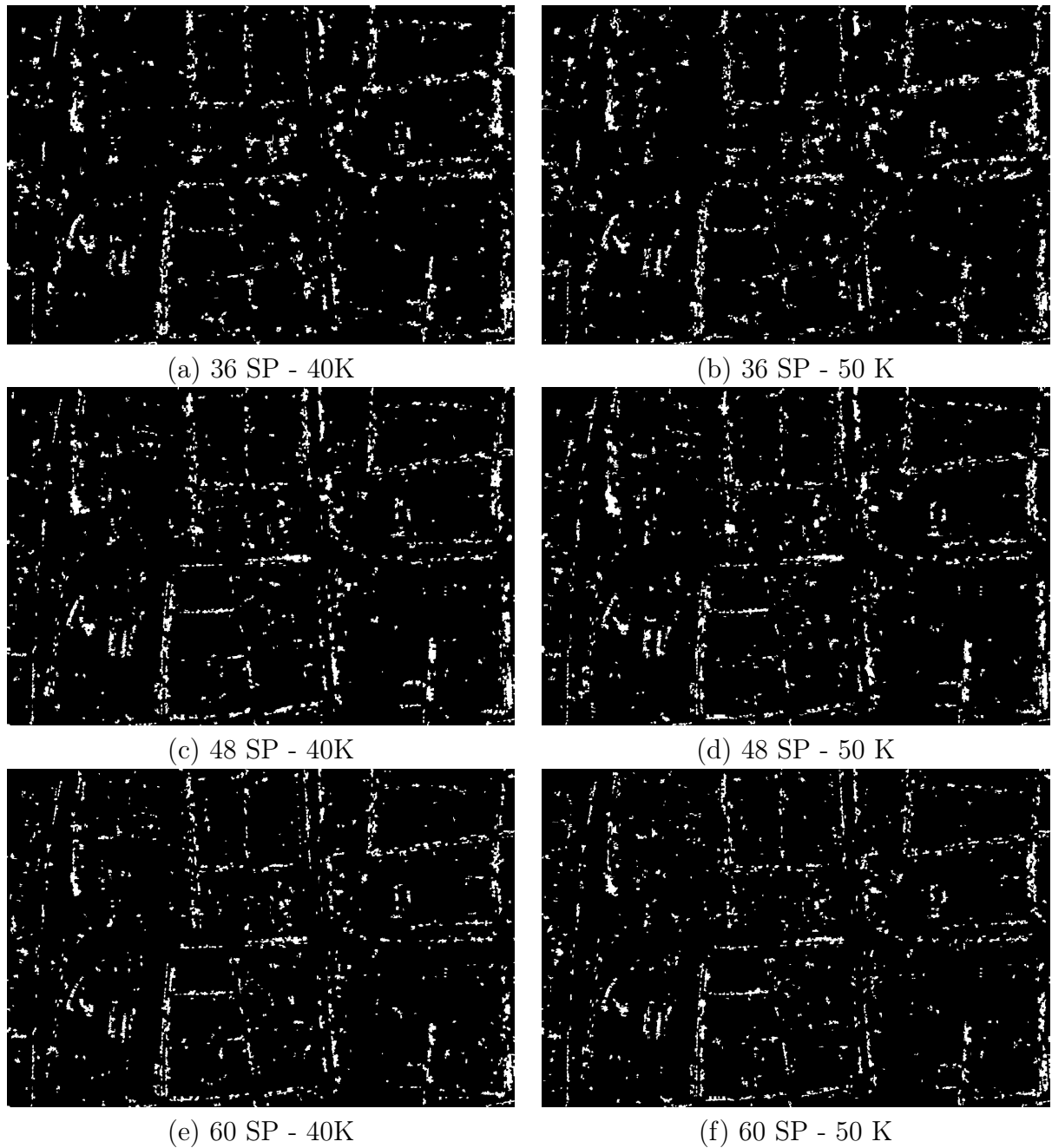


Figure 4.10. Map generated by the proposed method for image 0110 from the test subset from the Munich Vehicle Detection dataset, with the number of superpixels used for segmenting the entire image ranging from 40K to 50K and each selected frame segmented in 36, 48 and 60 superpixels.

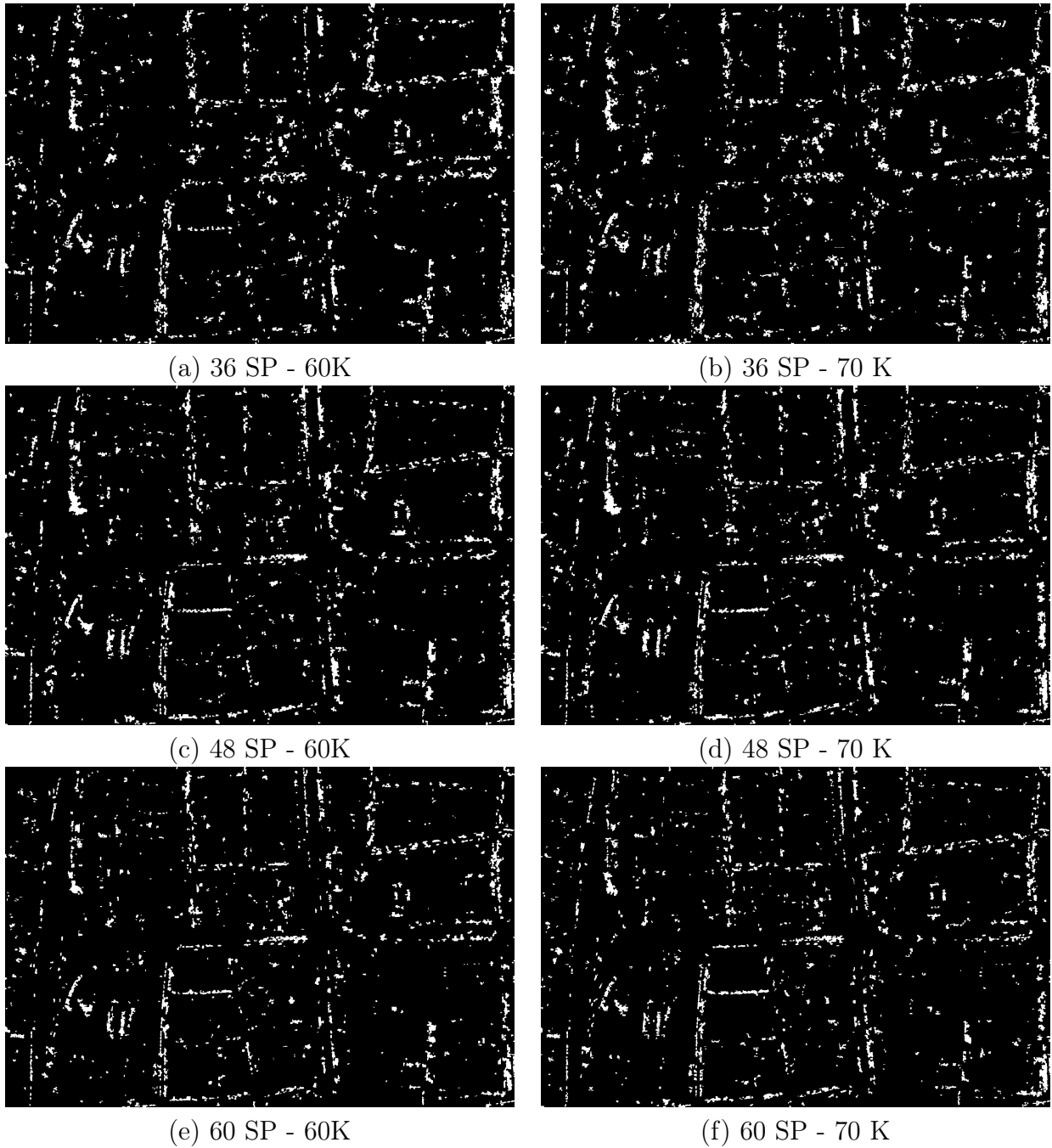
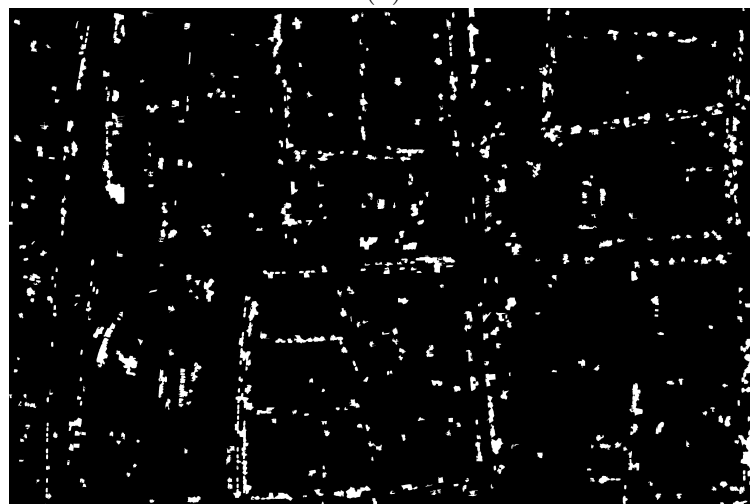


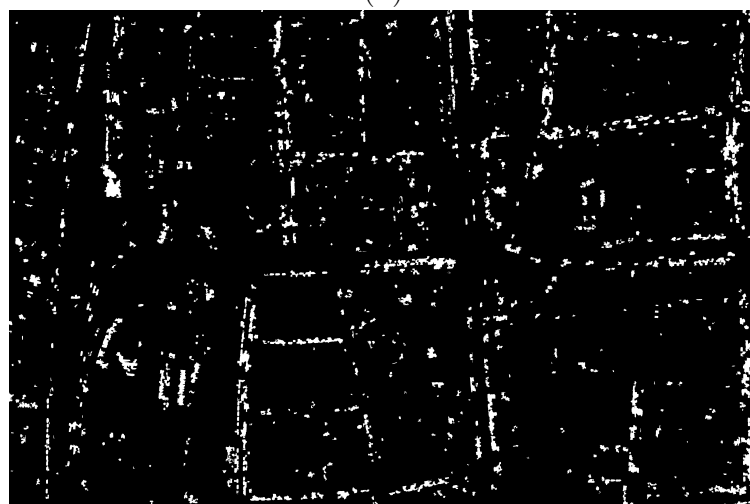
Figure 4.11. Map generated by the proposed method for image 0110 from the test subset from the Munich Vehicle Detection dataset, with the number of superpixels used for segmenting the entire image ranging from 60K to 70K and each selected frame segmented in 36, 48 and 60 superpixels.



(a)

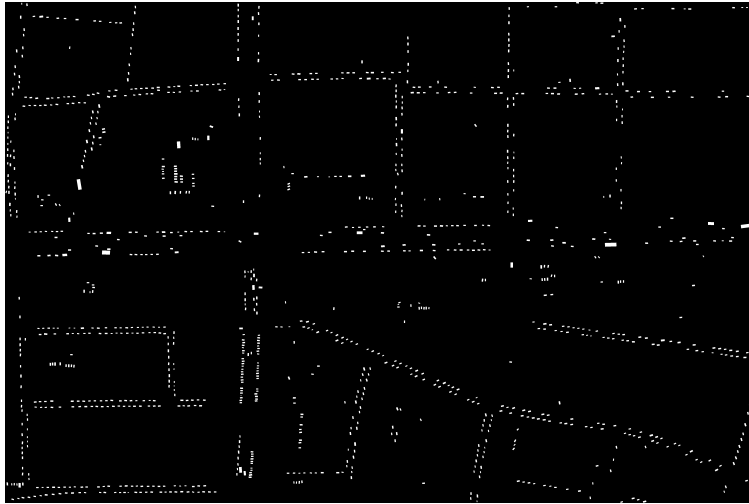


(b)

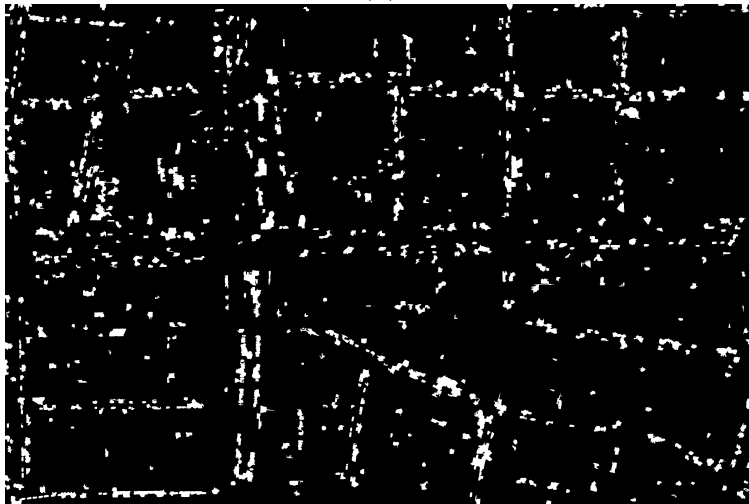


(c)

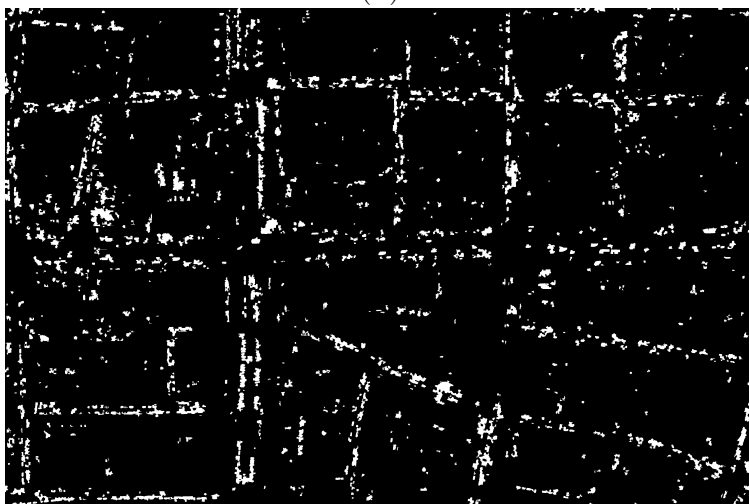
Figure 4.12. Visual comparison of the results obtained for image 0110 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.



(a)



(b)



(c)

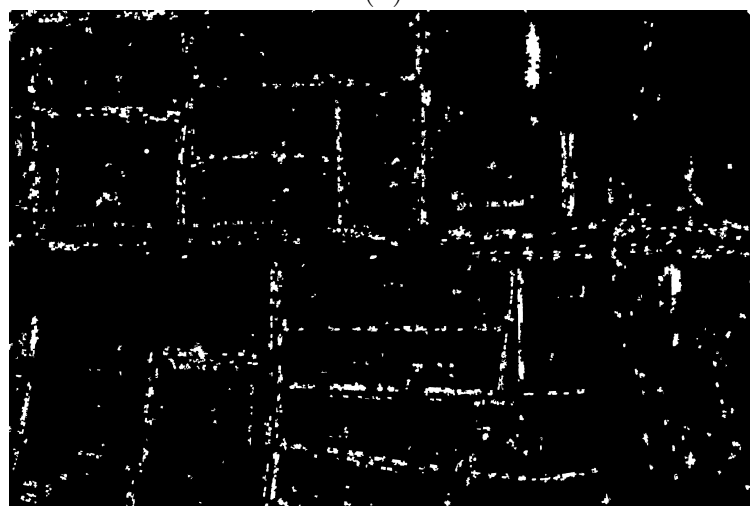
Figure 4.13. Visual comparison of the results obtained for image 0120 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.



(a)

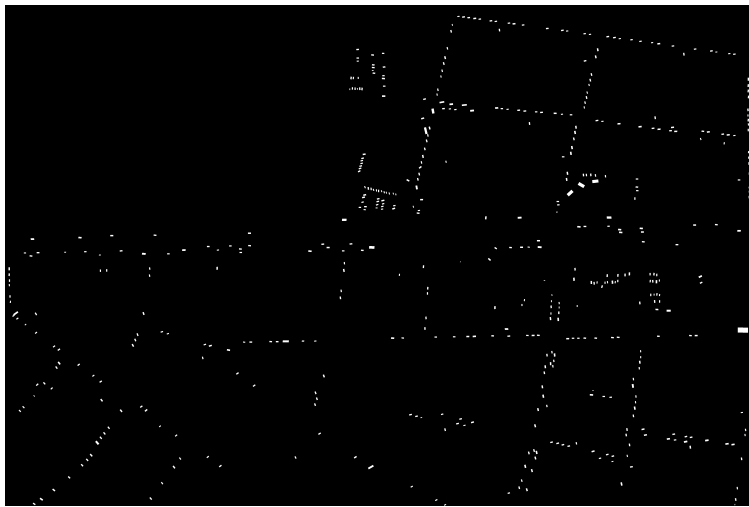


(b)

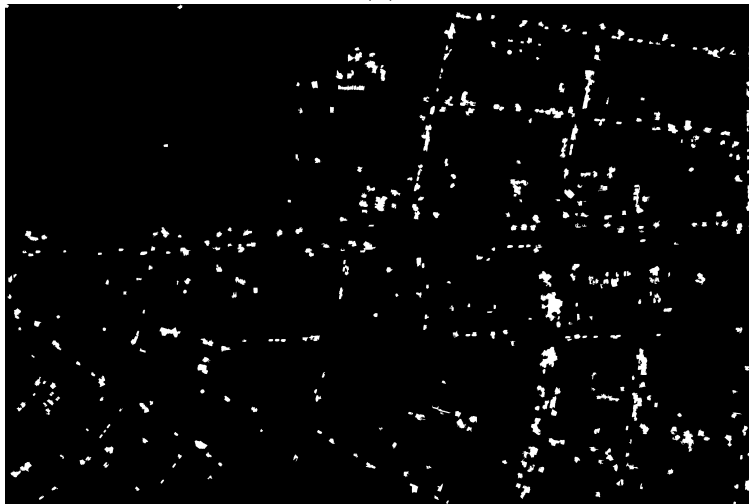


(c)

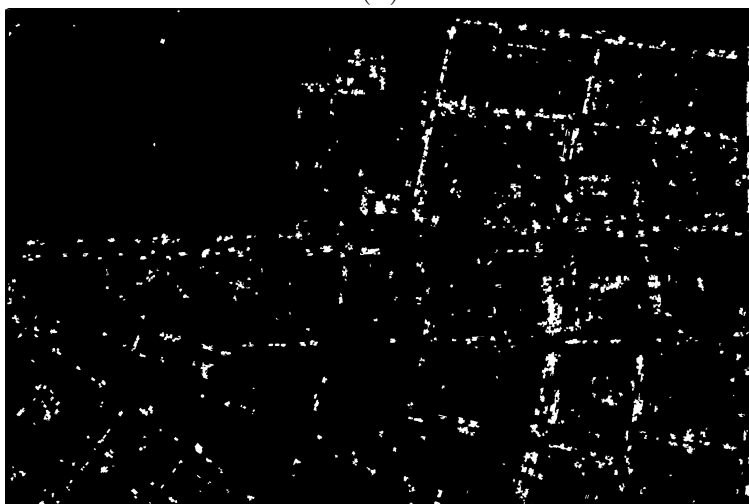
Figure 4.14. Visual comparison of the results obtained for image 0120 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.



(a)



(b)

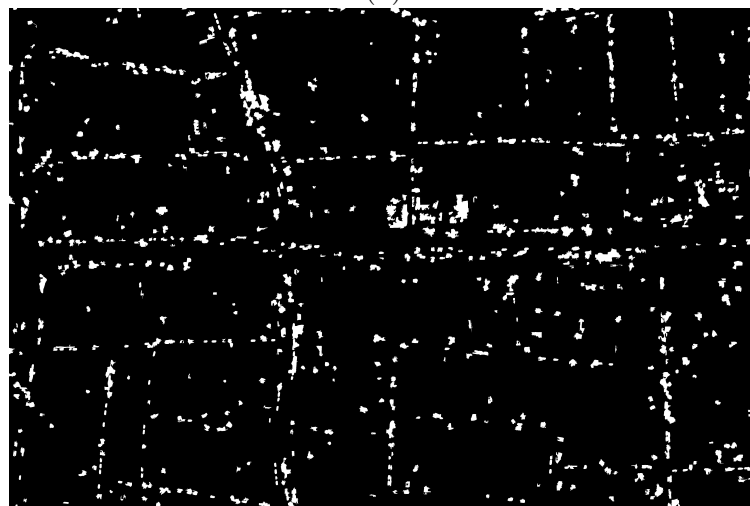


(c)

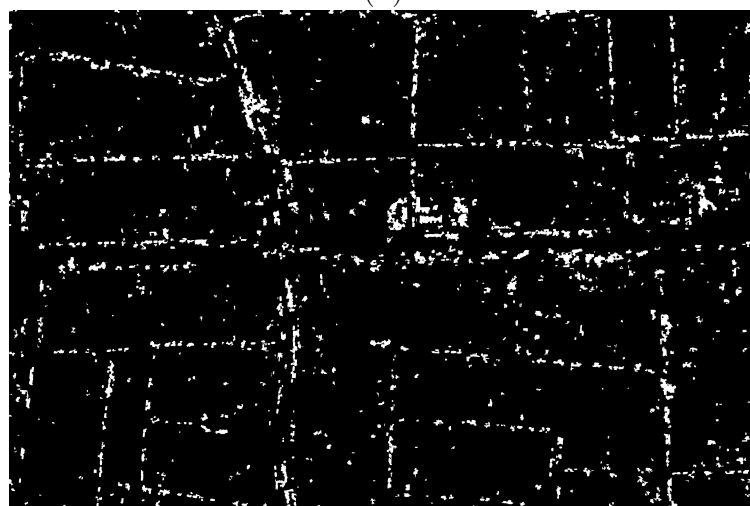
Figure 4.15. Visual comparison of the results obtained for image 0140 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.



(a)



(b)

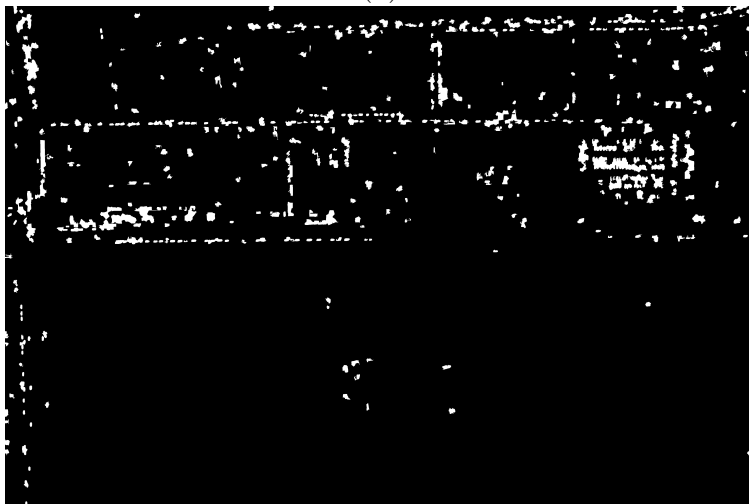


(c)

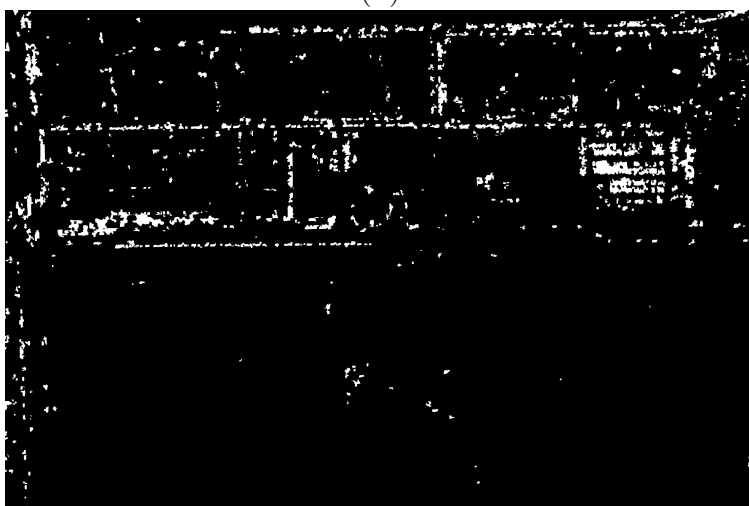
Figure 4.16. Visual comparison of the results obtained for image 0150 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.



(a)



(b)

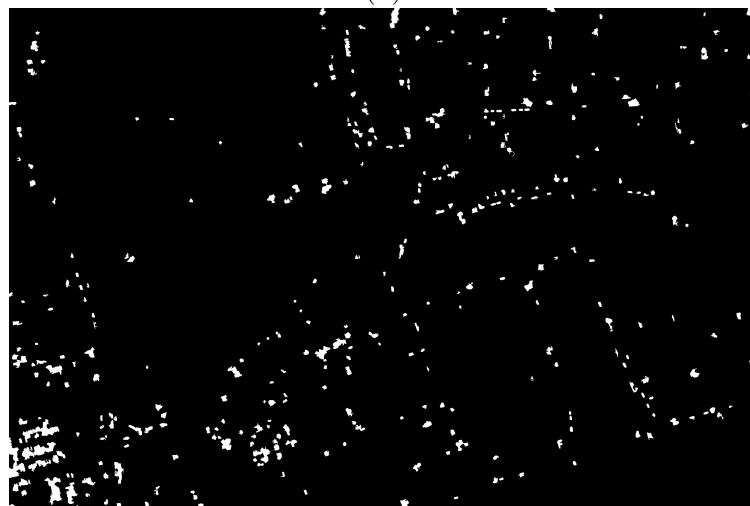


(c)

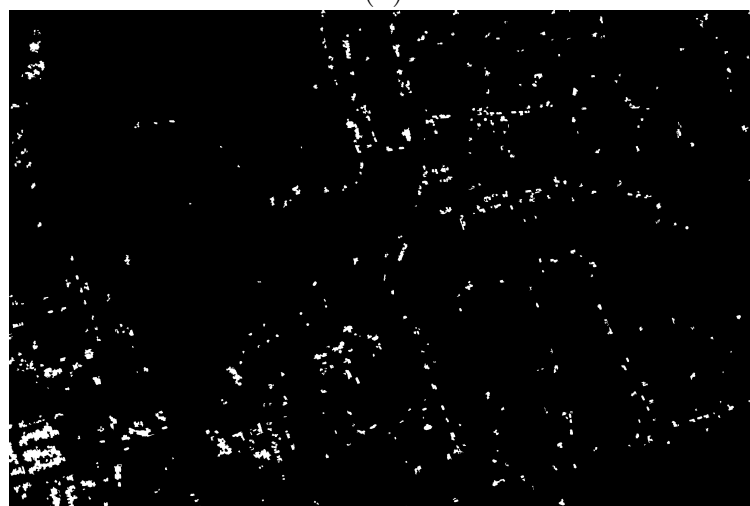
Figure 4.17. Visual comparison of the results obtained for image 0160 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.



(a)

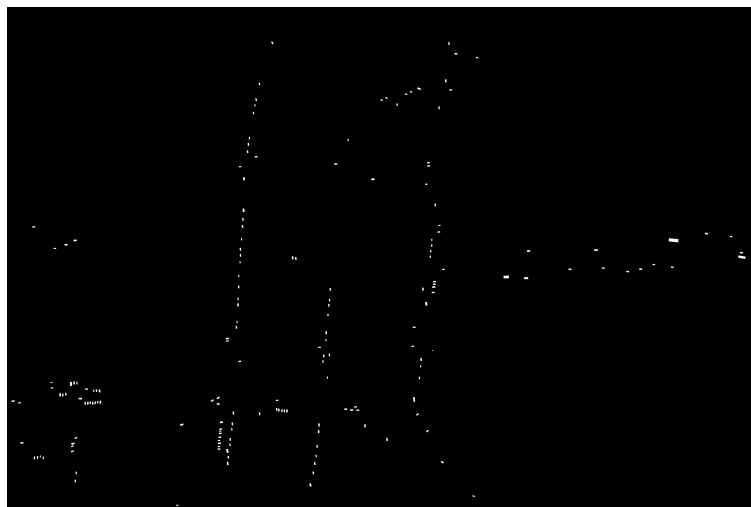


(b)

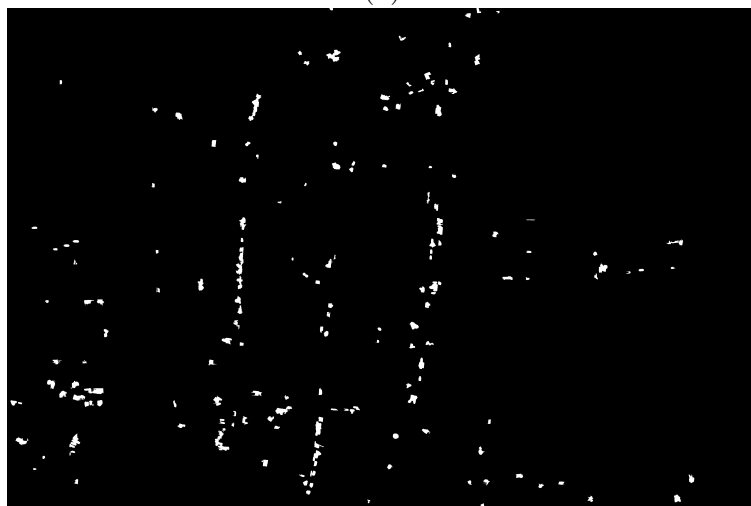


(c)

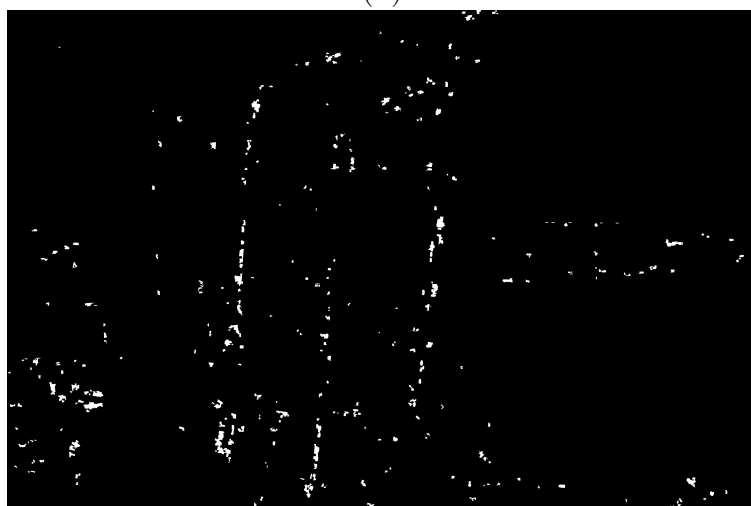
Figure 4.18. Visual comparison of the results obtained for image 0250 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.



(a)



(b)



(c)

Figure 4.19. Visual comparison of the results obtained for image 0265 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.



(a)



(b)

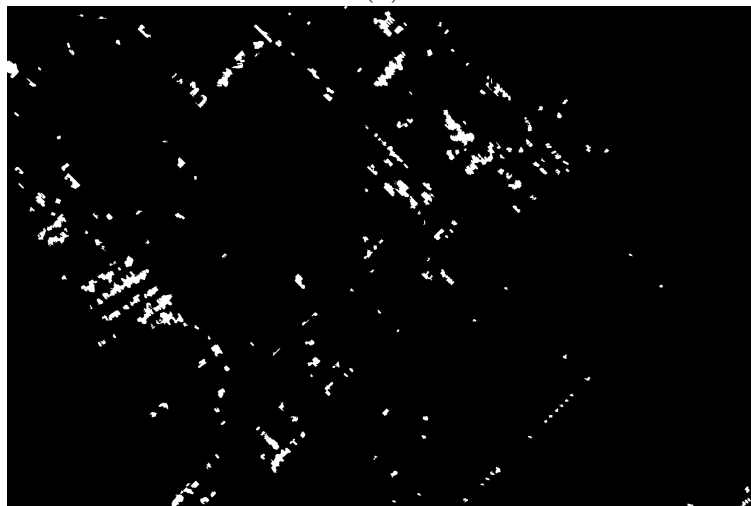


(c)

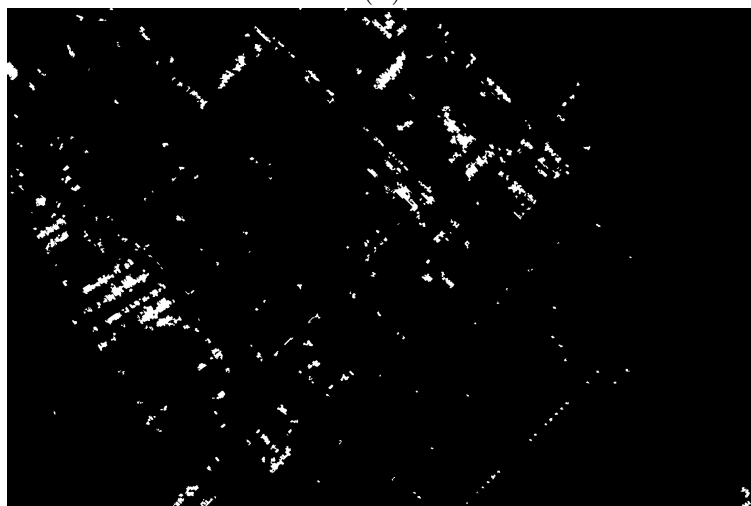
Figure 4.20. Visual comparison of the results obtained for image 0278 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.



(a)



(b)



(c)

Figure 4.21. Visual comparison of the results obtained for image 0120 from the test subset of the Munich Vehicle Detection dataset: (a) groundtruth, (b) map generated with the second segmentation parameter set as 20K and the first as 60, (c) map generated with the second segmentation parameter set as 80K and the first as 60.

Chapter 5

Conclusion

This work explored the effects of the use of contextual information provided by convolutional neural networks to the task of object detection in remote sensing images. The activation values of different layers of the convolutional networks were used as features, with different pooling methods and the concatenation of different layers being tested. The use of superpixel segmentation instead of the usually employed pixel-wise approach was employed. The proposed modifications were tested on the task of thematic map generation (for which the original method was initially used) and has provided slight improvements in the accuracy levels obtained, showing that the pooling method employed may lead to better results. The suitability of the method to the task of object detection was assessed on two custom built databases created through the extraction of patches from the Potsdam dataset, one containing positive and negative samples of trees while the second was built for a car detection scenario. The results obtained showed the success of the proposed method to such task, with accuracy levels above 99% being yielded. In order to obtain an assessment of a less controlled environment, the publicly available Munich Vehicle Dataset was also used. The effects of the variation of the number of superpixels used to segment the images and the individual patches processed was analyzed. As more superpixels were used, the number of true positives obtained increased, but at the same time the number of false positives also grew. Despite the bigger precision levels obtained with smaller number of superpixels, a visual analysis of the results shows that as more superpixels are employed, the sharper and more well delineated the cars detected are presented in the resulting maps. In future works we intend to explore the use of different convolutional networks and also analyze the performance of the method for the detection of multiple classes.

Bibliography

- R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, Nov 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2012.120.
- E. Alpaydin. *Introduction to machine learning*. MIT press, 2009.
- N. Ammour, H. Alhichri, Y. Bazi, B. Benjdira, N. Alajlan, and M. Zuair. Deep learning approach for car detection in uav imagery. *Remote Sensing*, 9(4):312, 2017.
- X. Chen, S. Xiang, C. L. Liu, and C. H. Pan. Vehicle detection in satellite images by hybrid deep convolutional neural networks. *IEEE Geoscience and Remote Sensing Letters*, 11(10):1797–1801, Oct 2014. ISSN 1545-598X. doi: 10.1109/LGRS.2014.2309695.
- G. Cheng and J. Han. A survey on object detection in optical remote sensing images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 117:11–28, 2016.
- N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 886–893, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2372-2. doi: 10.1109/CVPR.2005.177. URL <http://dx.doi.org/10.1109/CVPR.2005.177>.
- Z. Deng, H. Sun, S. Zhou, J. Zhao, and H. Zou. Toward fast and accurate vehicle detection in aerial images using coupled region-based convolutional neural networks. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(8):3652–3664, Aug 2017. ISSN 1939-1404. doi: 10.1109/JSTARS.2017.2694890.
- W. Diao, X. Sun, X. Zheng, F. Dou, H. Wang, and K. Fu. Efficient saliency-based object detection in remote sensing images using deep belief networks. *IEEE Geoscience and*

- Remote Sensing Letters*, 13(2):137–141, Feb 2016. ISSN 1545-598X. doi: 10.1109/LGRS.2015.2498644.
- J. A. dos Santos, O. A. B. Penatti, R. d. S. Torres, P. Gosselin, S. Philipp-Foliguet, and A. Falcão. Improving texture description in remote sensing image multi-scale classification tasks by using visual words. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 3090–3093, Nov 2012.
- L. Fang, S. Li, W. Duan, J. Ren, and J. A. Benediktsson. Classification of hyperspectral images by exploiting spectral and spatial information of superpixel via multiple kernels. *IEEE Transactions on Geoscience and Remote Sensing*, 53(12):6663–6674, Dec 2015a. ISSN 0196-2892. doi: 10.1109/TGRS.2015.2445767.
- L. Fang, S. Li, X. Kang, and J. A. Benediktsson. Spectral and spatial classification of hyperspectral images with a superpixel-based discriminative sparse model. *IEEE Transactions on Geoscience and Remote Sensing*, 53(8):4186–4201, Aug 2015b. ISSN 0196-2892. doi: 10.1109/TGRS.2015.2392755.
- C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, Aug 2013. ISSN 0162-8828. doi: 10.1109/TPAMI.2012.231.
- K. Fukushima. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2):119 – 130, 1988. ISSN 0893-6080. doi: [https://doi.org/10.1016/0893-6080\(88\)90014-7](https://doi.org/10.1016/0893-6080(88)90014-7). URL <http://www.sciencedirect.com/science/article/pii/0893608088900147>.
- B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *2009 IEEE 12th International Conference on Computer Vision*, pages 670–677, Sept 2009. doi: 10.1109/ICCV.2009.5459175.
- C. Galleguillos and S. Belongie. Context based object categorization: A critical survey. *Computer vision and image understanding*, 114(6):712–722, 2010.
- R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '14*, pages 580–587, Washington, DC, USA, 2014a. IEEE Computer Society. ISBN 978-1-4799-5118-5. doi: 10.1109/CVPR.2014.81. URL <https://doi.org/10.1109/CVPR.2014.81>.

- R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '14*, pages 580–587, Washington, DC, USA, 2014b. IEEE Computer Society. ISBN 978-1-4799-5118-5. doi: 10.1109/CVPR.2014.81. URL <http://dx.doi.org/10.1109/CVPR.2014.81>.
- R. C. Gonzalez, R. E. Woods, S. L. Eddins, et al. *Digital image processing using MATLAB.*, volume 624. Pearson-Prentice-Hall Upper Saddle River, New Jersey, 2004.
- I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. The MIT Press, 2016. ISBN 0262035618, 9780262035613.
- B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 447–456, 2015.
- K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016a. doi: 10.1109/CVPR.2016.90.
- Z. He, L. Liu, S. Zhou, and Y. Shen. Learning group-based sparse and low-rank representation for hyperspectral image classification. *Pattern Recognition*, 60:1041 – 1056, 2016b. ISSN 0031-3203. doi: <https://doi.org/10.1016/j.patcog.2016.04.009>. URL <http://www.sciencedirect.com/science/article/pii/S0031320316300425>.
- D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, 195(1):215–243, 1968. doi: 10.1113/jphysiol.1968.sp008455. URL <https://physoc.onlinelibrary.wiley.com/doi/abs/10.1113/jphysiol.1968.sp008455>.
- S. Jia, B. Deng, J. Zhu, X. Jia, and Q. Li. Superpixel-based multitask learning framework for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(5):2575–2588, May 2017. ISSN 0196-2892. doi: 10.1109/TGRS.2017.2647815.
- A. Kloss. *Object Detection Using Deep Learning-Learning where to search using visual attention*. PhD thesis, Universität Tübingen Tübingen, Germany, 2015.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference*

- on Neural Information Processing Systems - Volume 1*, NIPS'12, pages 1097–1105, USA, 2012a. Curran Associates Inc. URL <http://dl.acm.org/citation.cfm?id=2999134.2999257>.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, NIPS'12, pages 1097–1105, USA, 2012b. Curran Associates Inc. URL <http://dl.acm.org/citation.cfm?id=2999134.2999257>.
- C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008. doi: 10.1109/CVPR.2008.4587586.
- Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, Dec 1989. ISSN 0899-7667. doi: 10.1162/neco.1989.1.4.541.
- J. Leitloff, D. Rosenbaum, F. Kurz, O. Meynberg, and P. Reinartz. An operational system for estimating road traffic information from aerial images. *Remote Sensing*, 6(11):11315–11341, 2014. ISSN 2072-4292. doi: 10.3390/rs61111315. URL <http://www.mdpi.com/2072-4292/6/11/11315>.
- J. Li, H. Zhang, and L. Zhang. Efficient superpixel-level multitask joint sparse representation for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 53(10):5338–5351, Oct 2015. ISSN 0196-2892. doi: 10.1109/TGRS.2015.2421638.
- S. Li, T. Lu, L. Fang, X. Jia, and J. A. Benediktsson. Probabilistic fusion of pixel-level and superpixel-level hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 54(12):7416–7430, Dec 2016. ISSN 0196-2892. doi: 10.1109/TGRS.2016.2603190.
- W. Liao, X. Huang, F. V. Coillie, S. Gautama, A. Pižurica, W. Phillips, H. Liu, T. Zhu, M. Shimoni, G. Moser, and D. Tuia. Processing of multiresolution thermal hyperspectral and digital color data: Outcome of the 2014 ieeegrss data fusion contest. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):2984–2996, June 2015. ISSN 1939-1404. doi: 10.1109/JSTARS.2015.2420582.

- T. Lillesand, R. W. Kiefer, and J. Chipman. *Remote sensing and image interpretation*. John Wiley & Sons, 2014.
- K. Liu and G. Mattyus. Fast multiclass vehicle detection on aerial images. *IEEE Geoscience and Remote Sensing Letters*, 12(9):1938–1942, Sept 2015. ISSN 1545-598X. doi: 10.1109/LGRS.2015.2439517.
- J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, June 2015. doi: 10.1109/CVPR.2015.7298965.
- D. Marr and E. Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London B: Biological Sciences*, 207(1167):187–217, 1980. ISSN 0080-4649. doi: 10.1098/rspb.1980.0020. URL <http://rspb.royalsocietypublishing.org/content/207/1167/187>.
- P. R. Meneses, T. d. Almeida, et al. Introdução ao processamento de imagens de sensoriamento remoto. *Brasília: UnB*, pages 01–33, 2012.
- R. S. Michalski, J. G. Carbonell, and T. M. Mitchell. *Machine learning: An artificial intelligence approach*. Springer Science & Business Media, 2013.
- H. A. Mohammed. *Object detection and recognition in complex scenes*. PhD thesis, 2014.
- M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3376–3385, June 2015. doi: 10.1109/CVPR.2015.7298959.
- N. M. Nasrabadi. Pattern recognition and machine learning. *Journal of electronic imaging*, 16(4):049901, 2007.
- O. A. Pappas, A. M. Achim, and D. R. Bull. Superpixel-guided cfar detection of ships at sea in sar imagery. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1647–1651, March 2017. doi: 10.1109/ICASSP.2017.7952436.
- O. A. B. Penatti, K. Nogueira, and J. A. dos Santos. Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? In *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 44–51, June 2015. doi: 10.1109/CVPRW.2015.7301382.

- F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.
- S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. 39, 06 2015.
- T. M. H. C. Santana, K. Nogueira, A. M. C. Machado, and J. A. dos Santos. Deep contextual description of superpixels for aerial urban scenes classification. In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 3027–3031, July 2017. doi: 10.1109/IGARSS.2017.8127636.
- C. Shi and C.-M. Pun. Superpixel-based 3d deep neural networks for hyperspectral image classification. *Pattern Recogn.*, 74(C):600–616, Feb. 2018. ISSN 0031-3203. doi: 10.1016/j.patcog.2017.09.007. URL <https://doi.org/10.1016/j.patcog.2017.09.007>.
- K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. URL <http://arxiv.org/abs/1409.1556>.
- N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, Jan. 2014. ISSN 1532-4435. URL <http://dl.acm.org/citation.cfm?id=2627435.2670313>.
- O. Stenroos et al. Object detection from images using convolutional neural networks. 2017.
- Z. Sun and M. Chi. Superpixel-based active learning for the classification of hyperspectral images. In *2015 7th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4, June 2015. doi: 10.1109/WHISPERS.2015.8075388.
- J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2): 154–171, Sep 2013. ISSN 1573-1405. doi: 10.1007/s11263-013-0620-5. URL <https://doi.org/10.1007/s11263-013-0620-5>.
- A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *2009 IEEE 12th International Conference on Computer Vision*, pages 606–613, Sept 2009. doi: 10.1109/ICCV.2009.5459183.

- X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *2013 IEEE International Conference on Computer Vision*, pages 17–24, Dec 2013. doi: 10.1109/ICCV.2013.10.
- D. Yu, H. Wang, P. Chen, and Z. Wei. Mixed pooling for convolutional neural networks. In D. Miao, W. Pedrycz, D. Ślzak, G. Peters, Q. Hu, and R. Wang, editors, *Rough Sets and Knowledge Technology*, pages 364–375, Cham, 2014. Springer International Publishing. ISBN 978-3-319-11740-9.