CODIFICANDO CONTEXTO DE SUPERPIXELS PARA MELHORAR MAPAS URBANOS DE COBERTURA TERRESTRE

TIAGO MOREIRA HÜBNER CANÇADO SANTANA

CODIFICANDO CONTEXTO DE SUPERPIXELS PARA MELHORAR MAPAS URBANOS DE COBERTURA TERRESTRE

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
Coorientador: Alexei Manso Côrrea Machado

Belo Horizonte
Setembro de 2017

TIAGO MOREIRA HÜBNER CANÇADO SANTANA

ENCODING CONTEXT FROM SUPERPIXELS TO IMPROVE LAND-COVER URBAN MAPS

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
Co-Advisor: Alexei Manso Côrrea Machado

Belo Horizonte
September 2017

Santana, Tiago Moreira Hübner Cançado

S232e Encoding context from superpixels to improve land-cover urban maps / Tiago Moreira Hübner Cançado Santana. — Belo Horizonte, 2017.

xxiv, 81 f. : il. ; 29cm.

Dissertação (mestrado) — Universidade Federal de Minas Gerais — Departamento de Ciência da Computação.

Orientador: Jefersson Alex dos Santos
Coorientador: Alexei Manso Côrrea Machado

1. Computação - Teses. 2. Visão por computador. 3. Sensoriamento Remoto. 4. Reconhecimento de Padrões. informação – Teses. I. Orientador. II. Coorientador. III. 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

Encoding context from superpixels to improve land-cover maps

TIAGO MOREIRA HÜBNER CANÇADO SANTANA

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

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

PROF. ALEXEI MANSO CORREA MACHADO - Coorientador
Departamento de Ciência da Computação - PUC - MG

PROF. FÁBIO AUGUSTO MENOCCI CAPPABIANCO
Instituto de Ciência e Tecnologia - UNIFESP

PROF. WILLIAM ROBSON SCHWARTZ
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 28 de setembro de 2017.

I dedicate this work to my Father, the only God, Adonai, who has loved me unconditionally, redeemed me, saved me by grace through faith and has been so kind to me. He gave His best for me: Jesus, the Messiah. I also dedicate this work to my wife Géssica, to my parents Julio and Regina and to my brother Carlos for their love, support and comprehension.

Acknowledgments

First of all, I thank my Father, Adonai, who gave me life, intelligence and resilience, allowing me to finish the master's degree. I also would like to thank the Holy Spirit, who has been my company in each and every moment and has supported me mainly through the hardest ones. Thanks for the unconditional love.

Secondly, I am very grateful to my wife, parents and brother. My parents Julio and Regina for their unconditional love, striving to raise me giving the best of themselves. My brother Carlos for being the best brother ever. And my wife Géssica for her love, patience and support.

In the third place, I would like to thank my advisor Jefersson for never giving up on me, for the advices that led me to grow up as a person and researcher and for always believing in my potential, even when I did not. He is a great person and advisor.

I also thank all my professors, colleagues and employees of the Department of Computer Science. Each of you helped me a lot.

Finally, I would like to thank agencies CNPq, CAPES and FAPEMIG for funding this work.

"To every thing there is a season, and a time to every purpose under the heaven:"
(Holy Bible – King James Version. Ecclesiastes 3:1)

Resumo

Desde o começo da década de 70, quando as primeiras Imagens de Sensoriamento Remoto (ISRs) tornaram-se disponíveis a partir de satélites civis, o mapeamento automático de cobertura terrestre tem sido um tema central de pesquisa no campo devido à sua importância socio-econômica: tais mapas são uma das principais fontes de informação para estudos que embasam a criação de políticas públicas em atividades como o planejamento urbano e o monitoramento ambiental, por exemplo. O processo para gerar os mapas a partir das imagens é frequentemente modelado como um problema de classificação supervisionada, onde algumas amostras de cada classe alvo anotadas pelo usuário na imagem são usadas para treinar um classificador, que é utilizado para anotar as amostras restantes. Na medida em que a resolução espacial dos sensores usados para adquirir as imagens tornou-se mais fina, o paradigma de anotar e classificar pixels que foi dominante desde as primeiras abordagens deu espaço para o baseado em regiões, uma vez que cada objeto significativo contido em ISRs agora é composto de vários pixels. No entanto, descritores de baixo nível como cor e forma não são suficientes para produzir uma representação discriminativa para as amostras que representam objetos que compartilham aparência visual semelhante. Em tais situações, agregar informações da cena como um todo ou de objetos vizinhos pode ser útil para ajudar a distinguí-los. No intuito de explorar essa abordagem que está apenas começando a ser usada para o paradigma baseado em regiões, foram propostos três métodos para codificar o contexto de superpixels neste trabalho: o primeiro método modela cada vizinhança local composta de um superpixel e seus vizinhos como um Grafo de Adjacência de Regiões (GAR) e combina representações de baixo nível extraídas dos vértices e arestas em um único vetor de características que codifica tanto a aparência visual quanto o contexto do superpixel; o segundo codifica o contexto semântico de uma vizinhança local ao redor do superpixel contando a co-ocorrência de palavras visuais dentro dele e de seus vizinhos; e o último método proposto explora *ConvNets* para calcular características contextuais profundas a partir de estruturas de imagem com forma irregular, como é o caso dos superpixels. Confirmando estudos anteriores que mostraram que codificar contexto seja de pixels ou regiões é uma abordagem promissora, todos os três métodos propostos foram capazes de melhorar os mapas gerados ao incorporar contexto nas representações usadas para alimentar o classificador.

Palavras-chave: Sensoriamento Remoto, Mapas de Cobertura Terrestre, Classificação baseada em Regiões, Descritor Contextual.

Abstract

Since the early 1970's, when the first Remote Sensing Images (RSIs) became available from civilian satellites, automatic land-cover mapping has been a central research topic in the field due to its socioeconomic importance: such maps are one of the main sources of information for studies that support the creation of public policies in activities like urban planning and environmental monitoring, for instance. The process to generate the maps from the images is often modeled as a supervised classification problem, where few samples of each target class annotated by the user on the image are used to train a classifier, which is used to annotate the remaining samples. In so far as the spatial resolution of the sensors used to acquire RSIs got finer, the paradigm of annotating and classifying pixels that has been dominant since the initial approaches gave room to the region-based one, once each meaningful object depicted on RSIs is now composed of many pixels. Nevertheless, low-level descriptors like color and shape are not enough to produce a discriminative representation for the samples that represent objects that share similar visual appearance. In such situations, aggregating information from the scene as a whole or neighboring objects may be helpful to distinguish between them. In order to exploit this approach which is only beginning to be used for the region-based paradigm, three methods to encode the context of superpixels are proposed in this work: the first method models each local neighborhood composed of a superpixel and its neighbors as a Region Adjacency Graph (RAG) and combines low-level representations extracted from vertices and edges into just one feature vector that encodes both the visual appearance and the context of the superpixel; the second one encodes the semantic context from the local neighborhood around the superpixel by counting co-occurrences of visual words within it and its neighbors; and the last proposed method exploits ConvNets to compute deep contextual features from irregular-shaped image structures, as is the case of superpixels. Confirming previous studies which have shown that encoding context of either pixels or regions is a promising approach, all three proposed methods were able to improve the generated maps by incorporating context in the representations used to feed the classifier.

Palavras-chave: Remote Sensing, Land-cover maps, Region-based Classification, Contextual Descriptor.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

The intensive Brazilian urbanization and industrialization that have happened since the 50's led to the concentration of the population in some privileged regions, which have attracted people for their great job spots offering and better living conditions. In fact, throughout the 20th century the urbanization process resulted in many benefits, such as child mortality drop, life expectancy raise, fertility and illiteracy rates drop and increase in sanitation conditions and household waste collection. Despite all the advantages, urbanization also has its drawbacks like the compromise of environmental areas (such as stream borders, dunes and woods), favelas and illegal occupations growth, floodings due to the soil sealing, slopes collapse resulting in deaths and the compromise of water resources due to sewage pollution. In addition, as a result of the economic slowdown experienced after the 70's, the unemployment rate has raised and many public policies were withdrawn, resulting in the violence increasing [Vasconcelos et al., 2007]. The conclusion of Vasconcelos et al. [2007] is that urban planning is essential to solve such problems, provided that the projects are kept by the new governmental authorities, adapted to the local reality and there is popular participation.

A concrete example of the problems caused by the lack of urban planning is Betim, a city of the state of Minas Gerais, in Brazil. In the 1970's, when the automaker FIAT went to Betim, politicians started a market campaign that announced that Betim was the city that most generated jobs in Brazil. By that time, many people mainly from the northeastern region migrated to Betim. Without the technical requirements for the job opportunities, most of them remained unemployed and did not even have the money to return to their home cities. The politicians had not drawn up any policy capable of welcoming these people. As consequences, the city grew in a disorganized way, several favelas appeared and Betim is now the most violent city in the state. Another example that may be mentioned is the Tancredo Neves international airport, which was built on calcareous rocks. Observing remote sensing images, it is possible to see cave ceilings sinking in the region around the airport.

According to Duarte [2009], urbanism concerns about the physical-territorial aspect of cities, whereas urban planning is a more comprehensive term which also includes social and economical issues and methodologies from sociology, economy, geography, engineering, law and administration, in order to deploy policies and prohibitions that aim at improving the quality of life of the city dwellers. To accomplish such goals, detailed data is required to provide the information needed to make decisions. Currently, land cover/use thematic maps are one of the main sources of information used in this context [Hu et al., 2016].

Despite the similarity between the terms land cover and land use can be misleading, there is a clear distinction between them: the former denotes a physical description of the Earth surface, whereas the last one refers to the socioeconomic aspect of how human beings are using the land [Hu et al., 2016]. In this sense, a land cover thematic map is the geographical distribution of the materials covering the surface of a specif area depicted in a map.

Besides the ground survey, such maps are usually generated by means of a semi-automatic process carried out by experts aided by some Geographic Information System (GIS). Despite the high accuracy of the maps generated in this way, it is an expensive and time-consuming process [Wulder et al., 2004; Ippoliti et al., 2012]. Concerning urban scenarios, for instance, the dimensions of cities and the variety of shapes and materials found on them are great obstacles [Santana et al., 2017]. In this context, the automatic creation of land cover thematic maps arises as a feasible alternative which have been studied ever since the first multispectral imagery became available from civilian satellites in the 1970's. It is often modeled as a supervised classification process where the user annotates few samples (pixels, regions, superpixels, etc.) from which features are computed and used to feed a classifier during the training stage. At the completion of the training, the classifier should then be able to annotate the remaining samples [Wilkinson, 2005; Vargas et al., 2015] and, by mapping each class label into a color, a land-cover map can be built. An overview of the process is depicted in Figure 1.1.

Many approaches have been proposed in order to generate more accurate maps from satellite or aerial images. Typically, the *pixel-based* approach has been a dominant paradigm in the remote sensing field since its very beginning [Blaschke et al., 2014]. It performs a "hard" classification assigning just one land cover class to the pixel-based on features computed from it. Notwithstanding, a pixel could contain the spectral signature of more than one material (spectral mixture). There were many attempts

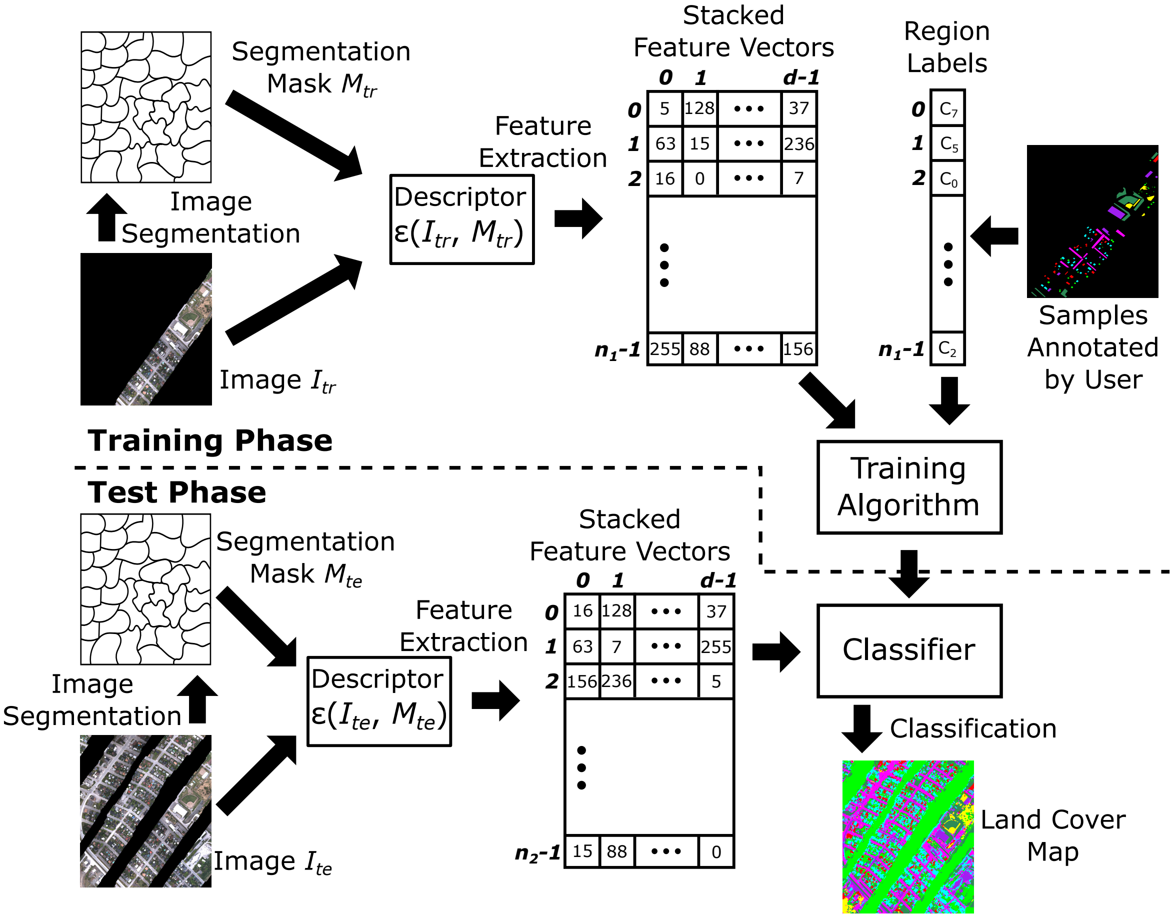


Figure 1.1: Overview of the process to automatically generate land cover maps: the image is segmented into n_1 regions from which d-dimensional feature vectors are extracted by using a descriptor ϵ; the vectors along with the labels of the regions annotated by the user are used to feed the training algorithm that fits a model; the resulting classifier is then used to annotate the remaining samples, which were extracted in the same way as the training ones; finally, the classification results are used to generate the land cover map.

to overcome such problem by developing methods that aimed to classify several land cover classes within a pixel (and, therefore, they were named *sub-pixel classification*) mainly through spectral unmixing [Blaschke et al., 2000]. This focus in pixel-based approaches for a long time is understandable as long as the pixel resolutions were relatively coarse, i.e., the the size of target objects were smaller or similar to the spatial resolution of the sensors and, therefore, each pixel used to depict at most a single target. Nevertheless, as the spatial resolution gets finer, single target objects are now represented by several pixels and, therefore, such approaches do not make sense anymore and neglect a lot of information that only can be extracted from a set of pixels, like texture or shape [Hay and Castilla, 2008]. Furthermore, while the problem of mixed pixels decreases, a new problem emerges: a pixel initially contained the spectral properties of several materials covering the Earth's surface and, thus, resulted in similar representations for all samples in each class and, consequently, low intra-class variance; once the pixel now contains the spectral properties of a single target, the intra-class variance increases, making the classification based on isolated pixels unsuitable. Additionally, the number of pixels required to cover the same area and, consequently, the volume of data needed to be stored and processed in analysis tasks, increases drastically making such tasks extremely time-consuming. [Blaschke et al., 2014]. For these reasons, the pixel-based approach has severely been criticized since the beginning of 2000's [Fisher, 1997; Blaschke and Strobl, 2001; Yu et al., 2006].

Along with the increasing in the spatial resolution of remotely sensed data, new approaches emerged as alternatives to the traditional pixel-based methods. They were based mainly on image texture and contextual information from pixels. Although these two terms are often taken as synonyms, there is a subtle distinction between them: the former can be defined as the "relationships between grey levels in neighboring pixels which contribute to the overall appearance of the image" [Marceau et al., 1990, p. 514] while the second one refers to the description of a kind of association between neighboring pixel values [Blaschke et al., 2014]. From the previous definition, we may observe that texture is much more related to a visually perceptible concept which is independent of color, just the differences or similarities in the brightness of neighboring pixels is enough to create such visual pattern called texture. On the other hand, the context of a pixel presupposes the existence of a latent association that causes them to occur according to the observed pattern.

However, this difference between the concepts is much more semantic than pragmatic, since some texture descriptors can indeed encode contextual information. It is the case of Gray Level Co-occurrence Matrix (GLCM) [Haralick et al., 1973], one of the most widely used texture descriptors, that counts co-occurrences of pixels when they are adjacent. It turns out that counting co-occurrences is the main way of encoding semantic context [Galleguillos and Belongie, 2010]. Thus, whether GLCM is encoding texture or context will depend on the semantic meaning of the image elements (pixels, regions, superpixels, etc.) and, therefore, on the scale of the image. To clarify, let us take an example: in a vehicle detection task, when the images under consideration are Very High Resolution (VHR) images of 10 cm, a car would be depicted by many pixels and, therefore, GLCM would encode the car texture; on the other hand, if the images were of 5 m of resolution, just a pixel would represent the same car and GLCM would encode its context. But the most important point here is that the use of texture as well as contextual information from pixels has improved classification results [Marceau et al., 1990; Tso and Mather, 1999; Stuckens et al., 2000].

Around the year 2000, a paradigm shift from the traditional pixel-based approach to the region-based perspective has begun. The emerging research area was named *Object-based Image Analysis* (OBIA) and, more recently, *Geographic Object-based Image Analysis* (GEOBIA) to make it clear that this sub-discipline of the Geographic Information Science (GIScience) derives its analysis from remotely sensed imagery, which depict the Earth's surface [Blaschke et al., 2014]. To understand this new paradigm, the concept of object must be determined. According to [Blaschke et al., 2014, p. 180], "image-objects represent 'meaningful' entities or scene components that are distinguishable in an image (e.g., a house, tree or vehicle in a 1:3000 scale colour airphoto)". From this definition, it is clear that whether a target on the surface is an object in a specific Remote Sensing Image (RSI) depends on the spatial resolution of the sensor used to image it. Another issue concerning GEOBIA is that it depends on a previous segmentation step to delineate the objects contained in the image [Blaschke et al., 2000; Blaschke, 2010; Blaschke et al., 2014]. In the current scenario, where VHR images are becoming more and more common and the spatial resolution of sensors keeps increasing, more objects of interest regarding a specific analysis task will be composed of many pixels in RSIs. Furthermore, the region-based approach reduces drastically the number of elements to be processed since groups instead of individual pixels will be handled and also allows one to exploit visual cues of the objects (e.g., texture and shape) that were neglected by the previous paradigm.

Although the usage of contextual information is very usual in the pixel-based paradigm, as can be noticed by the large number of references in the literature [Marceau et al., 1990; Binaghi et al., 1997; Tso and Mather, 1999; Stuckens et al., 2000; Melgani and Serpico, 2002; Fauvel et al., 2012; Moser et al., 2013], only very recently it begun to be exploited in the region-based approach [Santana et al., 2016]. To the best of our knowledge, just the work of Vargas et al. [2015] besides this dissertation have exploited contextual information from regions in the remote sensing field, even though it is a promising approach which is more widely used in the computer vision field [Galleguillos and Belongie, 2010; Parikh et al., 2012; Mostajabi et al., 2015]. The usage of contextual information from objects is of great relevance since low-level visual features (e.g., color, texture and shape) are limited to capture the differences in the appearance of distinct objects depicted in images where some modifying factors, like noise and impaired lighting conditions, are present or when the objects are visually similar. In such conditions, the feature vector generated for the objects is similar, though the objects might be from distinct classes, what reduces the class separability. In this case, the coherent arrangement of the objects or the visual cues of the neighboring objects might help to improve the class separability [Galleguillos and Belongie, 2010; Parikh et al., 2012; Santana et al., 2016, 2017].

Thus, the major contributions of this dissertation are twofold:

- Analysis of the implications of the definitions of semantic, spatial and scale context to encode them for objects in RSIs in practice;
- Proposal, development and validation of three methods to encode context in RSIs:
    1. Star, which encodes both the visual appearance and contextual information of superpixels by modeling each local neighborhood around them as a Region Adjacency Graph (RAG) and, then, combining feature vectors extracted from vertices and edges into one representation;
    2. VWCM, that encodes semantic context by counting co-occurrences of codewords both only inside the superpixel and co-occurrences between the it and each of its neighbors;
    3. MCL, that exploits ConvNets to compute deep contextual features from many context levels surrounding superpixels by keeping the mapping between the pixels within each of them and the feature maps across the network.

Some of the achieved results were published in the conference proceedings of CIARP 2016 (21st Iberoamerican Congress on Pattern Recognition) and IGARSS'17 (2017 IEEE International Geoscience and Remote Sensing Symposium). It is worth to mention that preliminary results of this work were presented at the Workshop of Works in Progress (WIP) within the 29th SIBGRAPI - Conference on Graphics, Patterns and Images, where the work was awarded a honorable mention. A more comprehensive description of the contributions of this work with new results is in progress and will be submitted to the IEEE Transactions on Geoscience and Remote Sensing (TGRS) journal.

The remainder of the text is organized as follows: Chapter 2 brings a literature review presenting related works and key concepts; Chapter 3 presents the three methods proposed; the settings and protocols used to validate the methods are described in Chapter 4; Chapter 5 presents the achieved results and brings a brief discussion about them; finally, Chapter 6 presents the conclusions and future works.

Chapter 2

Literature Review and Background

Since the 1970's, when the first multispectral imagery became available from civilian satellites, land-cover mapping has been a research topic that attracted much interest due to its socioeconomic importance. A land-cover map is a type of thematic map, which may be defined as a map that depicts the geographical distribution of some data on a specific theme or subject area, according to the [Intergovernmental Committee on Surveying and Mapping, 2017], that is a committee of the ANZLIC – the Australian and New Zealand Spatial Information Council. As opposed to general reference maps whose purpose is to show only physical features in order to summarize an area and help the reader's orientation, thematic maps use such features as a geographical reference to the data. Specifically in the case of the land-cover thematic maps, the phenomenon being mapped is the type of material covering the Earth surface.

The manual process to compose thematic maps from remotely sensed images involves the interpretation of the data which is done by an expert. Besides the dependence on the understanding of a person, land-cover mapping is very time-consuming when carried out in this way because of factors like the great number of spectral bands to analyze, the large area of study or the existence of many multi-temporal images [Meneses and de Almeida, 2012].

Therefore, the automatic generation of thematic maps arises as a feasible option. It is usually modeled as a supervised classification problem, where the user annotates few samples (may be pixels, regions, superpixels or image patches) from each of the target classes and the classifier learns patterns extracted from those samples via one or more kinds of descriptors. Then, the classifier is expected to able to annotate the remaining samples that were left without any labeling [Wilkinson, 2005; Vargas et al., 2015]. Finally, in order to build a thematic digital map, which is a digital image representing the geographical distribution of the phenomenon being mapped over the area of study, a different color is assigned to each target class and then all pixels within the area corresponding to a non-labeled sample are painted according to the class predicted by the classifier for that sample.

The process to automatically generate land-cover maps can be divided into 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, label prediction and map composition.

The first step is the detection and storage of the electromagnetic radiation reflected or emitted by objects or phenomena on the Earth surface. The sensors used may either only register the radiation (called passive sensors) or emit the radiation and register it after its interaction with one or more targets (named active sensors). In this sense, image acquisition is in the core of the definition of Remote Sensing as a science, according to Meneses and de Almeida [2012]: "Remote Sensing is a science which aims the development of the acquisition of images from the Earth surface by means of the detection and quantitative measurement of the answers of the interaction of electromagnetic radiation with terrestrial materials". Once distinct types of sensors are sensitive to electromagnetic radiation of different wavelengths, it is common to use a data-fusion approach in order to leverage their ability to highlight specific materials [de Andrade et al., 2015; Mou et al., 2017].

Nevertheless, remotely sensed imagery usually contain noise and errors due to the atmospheric interference and imaging geometry. This is the reason why the second step is needed: pre-processing techniques are applied in order to soften noise and correct radiometric and geometric distortions [Meneses and de Almeida, 2012].

Third step consists in employing some algorithm to delineate objects 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. Once the word object may have distinct meanings depending on the context where it is used, it is necessary to clarify which one we are referring to:

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

Notice from Definition 1 that what is distinguishable and therefore what is an object in each RSI depends on the spatial resolution of the image. Therefore, it is also important to define what is the spatial resolution. Combining the definitions proposed by Gonzalez and Woods [2006] to Meneses and de Almeida [2012], it may be defined 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 aims at producing a representation for image samples that describes them in terms of some kind of visual cue, such as color [Pass et al., 1996; Stehling et al., 2002], texture [Haralick et al., 1973; Unser, 1986; Huang and Liu, 2007] and shape [Dalal and Triggs, 2005], or even a combination of them. These representations are regarded as low-level, once they are based on computations over the own pixels. More complex representations also encode the context of the objects depicted in the images [Vargas et al., 2015; Santana et al., 2016, 2017] or apply some transformations over the low-level representations in order to generate a higher-order one, named mid-level representation [Sivic and Zisserman, 2003; Perronnin and Dance, 2007; Jégou et al., 2010]. Thereby, all the mentioned representations depend somehow on the low-level ones, which in turn are computed through image descriptors. Torres and Falcão [2006] defined a descriptor as:

Definition 3. *An image descriptor is a tuple (f_D, δ_D), where:*

- $f_D : I \rightarrow \Re^n$ *is a function which maps an image* I *to a point* v_I *(also known as feature vector) in the space* $\Re^n$*;*
- $\delta_D : \Re^n \times \Re^n \rightarrow \Re$ *is a similarity function which computes the similarity between two images* I_a *and* I_b *as a function of the distance between their corresponding feature vectors* v_{I_a} *and* v_{I_b}

Definition 4. *A feature vector* v_I *of an image* I *is a point in the space* $\Re^n : v_I = (v_1, v_2, v_3, \ldots, v_n)$*, where n is the dimension of the vector and usually* $v_1, v_2, v_3, \ldots, v_n \in \Re$.

It is important to notice that, even though Definition 3 and Definition 4 are in terms of image descriptors, which are presumably a single feature vector for each image, descriptors can also be computed from image regions, patches or superpixels.

dos Santos et al. [2010] evaluated several color and texture descriptors for RSIs classification. Four of them were chosen among the best ones, being two color descriptors and two texture descriptors:

- Border/Interior Pixel Classification (BIC) is a color descriptor that works on a RGB color space uniformly quantized in 64 colors. Then, each pixel is classified as either border or interior: if the pixel has the same color as its 4-neighbors, it is regarded as interior; when the pixel is at the border of the image or at least one of its 4-neighbors has a different color, it is classified as border. After pixel classification, two histograms are computed: one considering only border pixels and one taking into account only the interior ones [Stehling et al., 2002];

- Color Coherence Vectors (CCV) is also a color descriptor that works on a quantized color space, but only after blurring the image. After color quantization, connected components are computed considering pixels of the same color among the 8-neighbors. As a consequence, the image is segmented. Then, each pixel is classified as either coherent or incoherent: a pixel is coherent when it belongs to a connected component composed of at least τ pixels (τ is usually set to 300); otherwise it is incoherent. The final representation is the concatenation of two color histograms: one computed considering only the coherent pixels and the other regarding only the incoherent ones [Pass et al., 1996];

- Quantized Compound Change Histogram (QCCH) is a texture descriptor which describes the distribution of the rate of change around each pixel. The rate of change is determined for each direction (horizontal, vertical, diagonal and anti-diagonal) considering the 8-neighbors of the pixel. It is the absolute difference between the average gray tone of the 8-neighbors of the previous and next pixel regarding the target pixel in a given direction. The mean of the four rates of change is named compound rate of change. After computing the compound rate of change for each pixel, a non-uniform quantization is applied to map the resulting value into 40 integers. Finally, a histogram is composed by counting the number of pixel with each quantized compound rate of change [Huang and Liu, 2007];

- Unser also describes textural features from the image. It was developed as an alternative to the traditional gray-level co-occurrences matrix aiming at reducing its memory usage while maintaining almost the same accuracy. It begins by computing a histogram of sums and a histogram of differences considering pairs of pixels with a specific displacement. For each displacement formed by the combination of a radial distance and four angles (specifically, we used the distance 1.5 and angles 0°, 45°, 90°and 135°), a pair of histograms is computed (sums and differences). From each pair of histograms, 8 types of characteristics are extracted: mean, contrast, correlation, energy, entropy, homogeneity, maximal probability and standard deviation. This results in a 32-dimensional feature vector [Unser, 1986]

Besides using the aforementioned descriptors to represent superpixels, the feature vectors built by them were used to compose a mid-level representation. A mid-level representation may be defined as global features built from a composition of low-level ones [?]. They are a bridge between the low-level representations and the high-level ones [Peirce, 2015]. Among the various approaches available to compute mid-level representations, the most known is the traditional Bag of Visual Words (BoVW) [Sivic and Zisserman, 2003]. It is inspired on the method for text retrieval called Bag of Words (BoW), which represents each document as a vector of word frequencies. Likewise, BoVW represents an image as a vector of "visual words" frequencies. The method begins by extracting low-level features from the image by means of either sparse or dense sampling (the original paper proposes to detect key points before extracting features, even though dense sampling may also be employed). Then, vector quantization is performed in order to group feature vectors into clusters which are the "visual words". Next, a visual word is assigned to each key point or dense grid cell according to its similarity to the feature vector extracted from it. Finally, the frequencies that the visual words appear in the image (or region) are counted in a histogram, which now is taken as the image representation.

Model training and label prediction are two closely related steps. According to Alpaydin [2014], the purpose of machine learning is to solve problems based on example data or previous experiences. The problems are solved by using 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.

The learning process may be either supervised or unsupervised, depending on whether the output corresponding to each input example of the training data is available (supervised) or not (unsupervised). More specifically, in this work we are interested in supervised classification, which consists in predicting to which of a set of categories (or classes) a new sample belongs based on samples whose category is already known. There are several classifiers that fall into this group:

- Gaussian Naive Bayes [Alpaydin, 2014], which assumes that the distribution of the samples of each category is Gaussian and then uses the Bayes' theorem to compute the likelihood of a sample belongs to each category, choosing the category with higher probability;

- Support Vector Machines (SVM) [Cortes and Vapnik, 1995], that aims at finding the hyper-plane (or set of hyper-planes when the problem involves multi-class classification) that separates two classes with the largest margin, which is the distance between the hyper-plane to the closest feature vectors (called support vectors);

- Decision Trees [Breiman et al., 1984], which are non-parametric models that learns a set of decision rules using a tree structure where each internal node contains a decision based on the value of a non-empty set (may be a unitary set) of features, each branch is a decision made and the leaf nodes contain the predictions;

- k-Nearest Neighbors (kNN) [Cover and Hart, 1967], which is a lazy classifier (i.e., requires no training) that makes a prediction using the majority vote of the labels from the k neighbors from the training examples which are closest to the new observation;

- Multilayer Perceptron (MLP) [Hopfield, 1982], that is bio-inspired classifier which is actually an artificial neural network composed of layers of units called neurons. Each neuron receives some weighted inputs, sums them up and generates an activation using a specific function. The neurons of a layer are connected to those of the next one in order to send the value of the activation. The activations are forwarded across the network until the last layer, which output the prediction.

Many others examples of classifiers might be mentioned here, but these are very representative once there are many variations of them which are widely used nowadays. In this work specifically, we used SVM with Radial-basis Function (RBF) kernel in some experiments in order to handle non-linearly separable data, and eXtreme Gradient Boosting (XGBoost) [Chen and Guestrin, 2016] which is based on decision trees.

The label prediction step is to use the classifier after training in order to make predictions. In automatic land-cover mapping problem, the classifier is used to predict which is the material covering the Earth surface at the ground extension corresponding to each given feature vector computed from the RSI. Once the predictions were made by the classifier and one knows which pixels a feature vector was extracted from, it is possible to create a thematic land-cover map by painting those pixels with the color assigned to the class predicted by the classifier. This is the last step of the pipeline to automatically create land-cover maps.

Once this work is focused on the contextual description of superpixels from RSIs, context encoding is further explained in the following section.

2.1 Context Features

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. Each category is specified in the following:

Definition 5. *The semantic context of an object O is the likelihood of O is in a given image I with a set of objects S.*

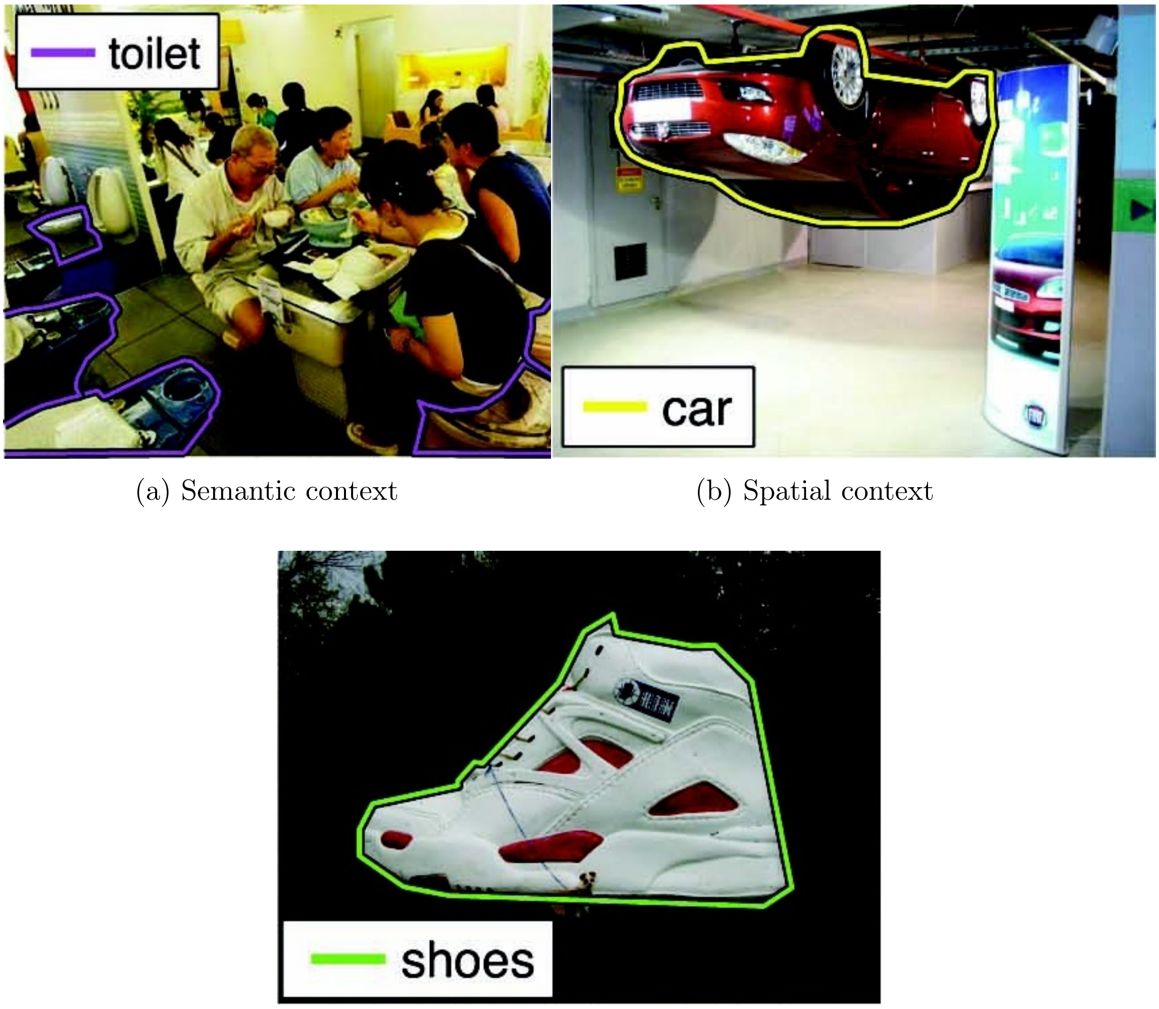
Definition 6. *Spatial context of an object O is the orientation and localization of O relative to each other object in image I.*

Definition 7. *Scale context of an object O is the size of O relative to all other objects in an given image I.*

Definitions 5, 6 and 7 have some implications. The first one concerning the semantic context is that it requires the previous identification of all objects in an image. On the other hand, in order to take scale context of an object into account it is necessary to identify at least one other object in the image and also employ some algorithm to obtain information of distance and depth between them. Another implication is that spatial and scale context implicitly include semantic context, once they depend on the identification of other objects, what constitutes a co-occurrence relation.

Such implications are not so strict in RSIs. Once the images are remotely sensed, information about the type and altitude of the sensor used is often readily available. Additionally, in so far as the distance between objects is nearly insignificant when compared to the distance from the sensor to the sensed targets, scale context becomes less sensitive to depth differences. Another important aspect that distinguishes context encoding in RSIs is the perspective from which the objects are imaged: once the position of the sensor with relation to the objects sensed has small variations across different RSIs and the perspective is always from top, the rotation of objects does not affect their context as much as in regular images taken on the ground.

For human beings, Definitions 5, 6 and 7 are very intuitive. Our previous visual experiences affect what objects we expect to found in a given scene, where we expect to see them within the scene and the size we expect them to have in order to judge that



(a) Semantic context

(b) Spatial context

(c) Scale context

Figure 2.1: Objects out of context. Images from SUN09 Dataset [Choi et al., 2010].

scene as coherent. Previous studies had proven that this knowledge accumulated from the many past visual experiences unconsciously take part in the process performed by the brain to identify objects [Palmer, 1975; Bar, 2004]. It is easy to notice this statement when looking at images like those in Figure 2.1, which shows objects out of context.

However, take advantage of context to identify objects is not such a trivial task for machines, once it involves a higher level of abstraction related to the identification of each object in an image, the own scene depicted and semantic, spatial and scale relations among them.

It is also important to highlight that context is considered from either a local or global image level [Galleguillos and Belongie, 2010]. Global context captures the interaction between the scene and objects and plays an essential role imposing a constraint on which objects should be expected to be in the image. On the other hand, local context consider areas surrounding the target object and therefore focuses on the interactions between pixels, regions/patches or objects. It supports the description of the objects, mainly those which can not be easily identified by visual cues [Mostajabi et al., 2015].

The first available methods used to be based on fixed and predefined inference rules and constraints that the objects (or parts of them) should satisfy in order to regard the scene arrangement as coherent. These rules and constraints were hand coded by experts [Hanson and Riseman, 1978; Strat and Fischler, 1991; Fischler and Elschlager, 1973]. Such rules determined which contextual relations were consistent and which were not. The main weaknesses of those early approaches was that they were constrained to a specific domain for which the rules were created and that they were not able to handle the uncertainty that is inherent to the real world.

More effective approaches emerged by introducing machine learning techniques Rabinovich et al. [2007]; Torralba [2003]; Shotton et al. [2009]. The contextual relations that used to be hand coded are now learned from features extracted from groups of either pixels [Parikh et al., 2008; Shotton et al., 2009] or regions [Galleguillos et al., 2008; Zhang and Saligrama, 2017].

A recent trend consists on combining different kinds of context to improve the classification Mottaghi et al. [2014], which is nevertheless computationally inefficient and, therefore, less used so far.

The main drawback of the methods presented so far is the requirement of previous identification of other objects in the image, once most of them are based on graphical models, like Markov Random Fields (MRFs) or Conditional Random Fields (CRFs), which demand a lot of labeled data to be trained. Such constraint reduces the number of methods that could be applied to land-cover automatic mapping, once if the users had to annotate most of the objects in the image, manual mapping would become a feasible option due to the higher accuracy of the maps generated.

A way to overcome this deficiency is through feature engineering, which consists in building a representation for image objects/regions that implicitly encodes their context. This approach must somehow include co-occurrences, scale or spatial relationships between descriptors of image elements without labeling them. An example can be found in the work of Lim et al. [2009] that represents the scene as a tree of regions where the leaves are described as a combination of features from their ancestors. This resulting descriptor encodes context in a top-down fashion.

To the best of our knowledge, the only approach of this type in remote sensing was proposed by Vargas et al. [2015] to create thematic maps. In that work, each superpixel of the image is described through a histogram of visual elements, using the method Bag of Visual Words (BoVW). Then, contextual information is aggregated by concatenating the superpixel description with a combination of the histograms of its neighbors to generate a new contextual descriptor. Even though this method represents each local neighborhood as a Region Adjacency Graph (RAG) where each vertex is a superpixel and edges model the adjacency relations between them, no description for the edges is included in the final representation.

Chapter 3

Encoding Context

In this chapter we describe the methods proposed in order to aggregate contextual information into the representation generated for image superpixels. The first method is called Star (Section 3.1), once it models each local neighborhood surrounding a superpixel as a Region Adjacency Graph (RAG) in star topology and, then, extracts feature vectors from vertices and edges, which are combined into one final representation that encodes both visual appearance and context. Section 3.2 presents the second one, which is referred to as Visual Words Co-occurrences Matrix (VWCM) once it is based on the widely known Gray-level Co-occurrences Matrix (GLCM) because it encodes semantic context by counting co-occurrences of codewords both only inside the superpixel and co-occurrences between it and each of its neighbors, what is similar to the count of co-occurrences of gray levels to compute texture features. The last proposed method exploits ConvNets 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, as described in Section 3.3. This method uses many layers of the ConvNet to compute features at different levels of context around the superpixel, from local to global so that we called it Many Context Levels (MCL).

3.1 Star Descriptor

Unlike the most methods found in the literature, the Star descriptor builds a representation for image segments that implicitly encodes co-occurrences (semantic context) and spatial relations (spatial context) without the need of labeling the segments. The pipeline to generate the Star descriptor is summarized in Figure 3.1 and further explained in the following.

3.1.1 Region Adjacency Graph

Given an image segmented into N superpixels, the local neighborhood of each superpixel s_i, $i = 1, \ldots, N$, is regarded as a graph $G_i(V, E)$ in star topology (see Figure 3.1) where V are the superpixels and the edges in E represent the adjacency relations between s_i and the remaining superpixels. Formally, two superpixels s_x and s_y are adjacent if and only if at least one pixel of s_x is 4-connected to a pixel of s_y. In addition, the target superpixel s_i is the central vertex (or root), each of its n neighboring superpixels ns_j, $j = 1, \ldots, n$, are the leaves and there is an edge e_k, $k = 1, \ldots, n$ linking the mass centers of s_i and every ns_j. Such a graph modeling provides a clear understanding of the proposed descriptor in terms of the level of context taken into account and the types of context exploited: local spatial relations and co-occurrences between the visual features extracted from s_i and its neighborhood.

3.1.2 Vertex Descriptor

A visual feature descriptor is computed within every superpixel in a given local neighborhood modeled as a star graph $G_i(V, E)$. More formally, a feature vector - referred to as target vertex descriptor (TVD_i) - is extracted from the target superpixel s_i. Likewise, a neighbor vertex descriptor NVD_j is built for each neighboring superpixel ns_j,

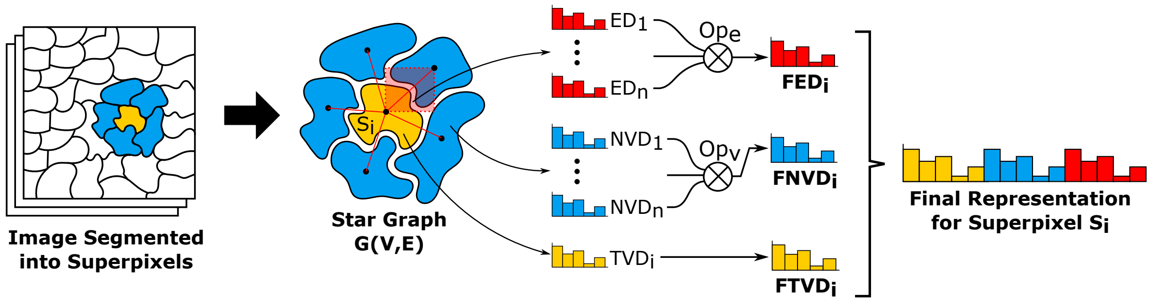


Image Segmented into Superpixels
Si
Star Graph G(V,E)
ED1
EDn
Ope
FEDi
NVD1
NVDn
Opv
FNVDi
TVDi
FTVDi
Final Representation for Superpixel Si

Figure 3.1: Process to generate the Star contextual descriptor for a superpixel s_i. Given a segmented image, the local neighborhood of s_i is modeled as a Region Adjacency Graph (RAG) $G_i(V, E)$ in star topology where s_i is the central vertex (or the root), the adjacent superpixels are the leaves and edges link the mass centers of them to s_i. A feature descriptor is extracted from s_i and from each of its n neighbors. Every edge is then taken as the diagonal of a rectangle (reddish region) from which another descriptor is computed. The n resultant edge descriptors are combined into one of the same dimensionality through some operation Op_e. Likewise, the n neighbor vertex descriptors are used to build only one through Op_v. Finally, the final contextual descriptor for s_i is composed by concatenating its own vertex descriptor, the final neighborhood vertex descriptor and the final edge descriptor, in this order and after individually normalizing each of them.

$j = 1, \ldots, n$, as can be seen in Figure 3.1. Notice that the same algorithm is used for both TVD_i and every NVD_j.

Although the only restriction to choose the vertex descriptor is that it must represent every superpixel as a fixed-size numerical vector, we propose to use two types: global feature descriptors and BoVW for mid level representation. In the former approach, a global descriptor is extracted from each superpixel taking it as it were a whole image. To account for size differences among them, the resultant feature vector is normalized. The second representation was proposed by Vargas *et al.* Vargas et al. [2015]: dense grid sampling is applied and a feature descriptor is computed from each 5x5 local patch around the selected pixels; the extracted feature vectors are used to conform the codebook using the k-means clustering algorithm; hard assignment is used to assign the closest visual word to each pixel of the grid; a histogram is then computed for every superpixel by taking into account the central grid pixels within it; finally, each histogram is normalized dividing it by the number of grid pixels inside its respective superpixel.

3.1.3 Edge Descriptor

The edge descriptor proposed by Silva et al. [2013] was used to better capture the visual patterns found across the frontiers of two adjacent superpixels since it extracts features around the edge. More precisely, given a local neighborhood represented as a star graph $G_i(V, E)$, the k-th edge descriptor ED_k is computed by extracting a feature descriptor (although the authors used only texture descriptors, we propose to experiment also color descriptors and deep features) within the rectangle formed by taking e_k as its diagonal (as exemplified by the reddish area nearby the edge in Figure 3.1). This process is repeated for each of the n edges in E.

3.1.4 Composition Operations

As soon as the vertex and edge descriptors were extracted, they are combined into just one vertex and one edge descriptor through some operation. This step is applied to tackle with two issues: due to the large number of feature vectors extracted from each RAG, the computational cost to train a classifier with them would be prohibitively high and the variability in the number of leaves of the graphs would result in a feature vector of non-fixed size if a simple concatenation would be done.

More specifically, an operation Op_v is applied to summarize the n NVDs, resulting in one final neighbor vertex descriptor $FNVD_i$. Similarly, the n EDs are combined into just one final edge descriptor FED_i through an operation Op_e. The final target vertex descriptor $FTVD_i$ is the TVD_i itself. Because vertex and edge descriptors lie in different feature spaces, $FTVD_i$, $FNVD_i$ and FED_i are individually normalized using L_2 norm and then concatenated to compose the final representation for s_i which has $2 * |vertexdescriptor| + |edgedescriptor|$ dimensions.

The only constraint imposed to Op_v and Op_e is that they must summarize n p-dimensional vectors into one of same dimensionality. Concretely, we propose to use three operations commonly found in BoVW pooling step: sum, average and max pooling. These operations are formally defined as follows: let D_j be the j-th p-dimensional feature vector in a sequence $\langle D_1, \dots, D_n \rangle$, whose components are d_m, $m \in \{1, \dots, p\}$ as stated in Eq. 3.1; the m-th component of D_j can be summarized through either sum, average or max pooling, which are respectively showed in Eq. 3.2, 3.3 and 3.4.

$$D_j = \{d_m\}_{m \in \{1,\dots,p\}} \tag{3.1}$$

$$d_m = \sum_{j=1}^{n} d_{j,m} \tag{3.2}$$

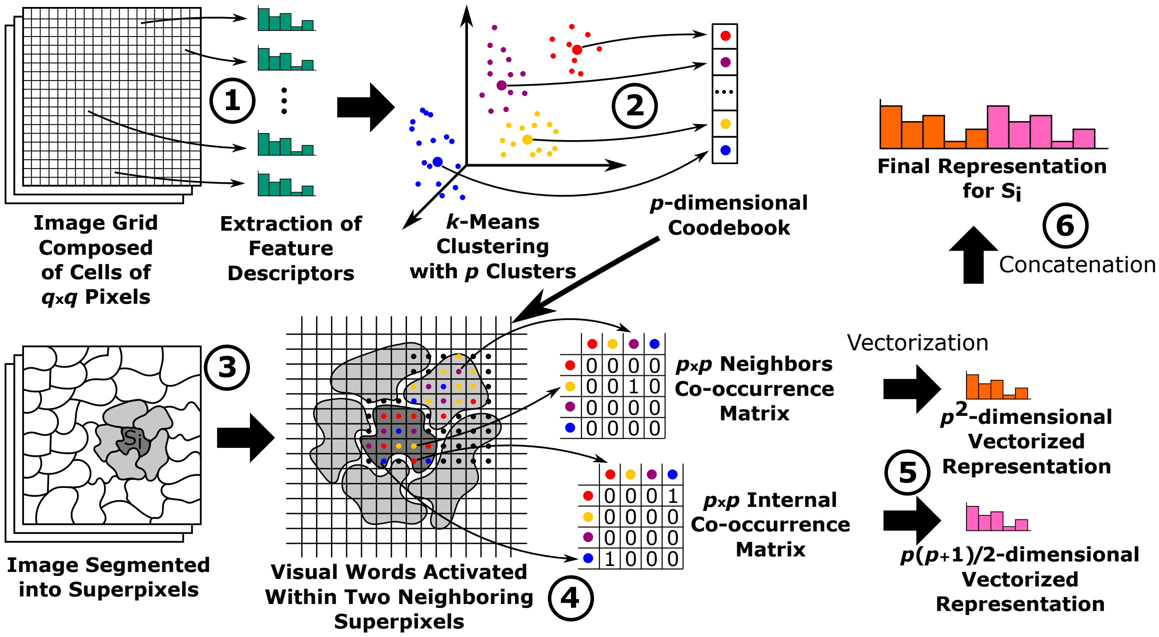
$$d_m = \frac{1}{n} \sum_{j=1}^{n} d_{j,m} \tag{3.3}$$

$$d_m = \max_{j \in \{1,\dots,n\}} d_{j,m} \tag{3.4}$$

Of course, such operations are invariant to rotation and translation since they result in the loss of some spatial information, e.g., the relative position of superpixels. Nevertheless, they are quite simple and perform very fast.

3.2 Visual Words Co-occurrence Matrix

Counting co-occurrences of elements is the main way to encode semantic context, which is related to the likelihood of a specific element be found in a scene either alone or along with other elements. However, since in Remote Sensing Images (RSIs) the scene is usually just an image mosaic containing possibly all elements under consideration for a specific classification problem, their co-occurrences would not encode much information. On one hand, we could delimit a small area to count co-occurrences so that we are still able to represent an element by encoding its context, but the resulting representation would be sparse due to the small number of elements considered. On


1
Image Grid Composed of Cells of *q*x*q* Pixels
Extraction of Feature Descriptors
2
k-Means Clustering with *p* Clusters
p-dimensional Codebook
Final Representation for Si
6
Concatenation
3
Si
Image Segmented into Superpixels
Visual Words Activated Within Two Neighboring Superpixels
4
*p*x*p* Neighbors Co-occurrence Matrix
*p*x*p* Internal Co-occurrence Matrix
Vectorization
5
*p*2-dimensional Vectorized Representation
p(*p*+1)/2-dimensional Vectorized Representation

	🔴	🟡	🟣	🔵
🔴	0	0	0	0
🟡	0	0	1	0
🟣	0	0	0	0
🔵	0	0	0	0

	🔴	🟡	🟣	🔵
🔴	0	0	0	1
🟡	0	0	0	0
🟣	0	0	0	0
🔵	1	0	0	0

Figure 3.2: Pipeline to represent a superpixel s_i by co-occurrences of visual words within its neighborhood. Firstly, grid sampling is applied to the image so that feature descriptors are computed, which are then clustered and the cluster centroids are stacked to compose a codebook. A visual word from the codebook is then assigned to each descriptor from the grid. Next, the image is segmented into superpixels and the resulting segmentation is superimposed with the grid in order to define which visual words lye inside each superpixel. The internal and the neighbors co-occurrence matrices are computed from s_i and its n adjacent superpixels. Finally, the matrices are vectorized, separately normalized and concatenated to compose the final representation for s_i.

the other hand, larger areas would result in very similar representations for adjacent elements due to the intersection of their contextual areas.

Aiming at tackling with such problems, we propose to represent a superpixel by counting co-occurrences of visual words inside it and its neighbors instead of counting the superpixels themselves. By counting visual words we overcome the sparsity problem while we are still able to generate discriminative representations by using just adjacent superpixels.

An overview of the proposed approach, which is called Visual Words Co-occurence Matrix (VWCM) in honor of the traditional Gray Level Co-occurrence Matrix (GLCM), can be seen in Figure 3.2.

3.2.1 Feature Extraction

The approach starts by defining a grid composed of cells of $q \times q$ pixels from which feature descriptors are extracted.

3.2.2 Codebook Composition

After performing the grid sampling, all descriptors computed are clustered into p clusters using the k-means algorithm. The cluster centroids are then stacked to compose the codebook which is used to look up which visual words is activated by each descriptor from the grid. The choice of the number of clusters p is critical, since a large value is going to generate a $p \times p$ co-occurrence matrices that are sparse while a small value would cause all the matrices to have nearly the same visual words and be therefore indistinguishable.

3.2.3 Segmentation

As soon as the coodebook is built, the image is segmented into N superpixels and the next steps are applied taking each superpixel s_i, $i = 1, \ldots, N$, along with all the n adjacent ones (notice that the definition of superpixel adjacency stated in Subsec. 3.1.1 is used here).

3.2.3.1 Counting Co-occurrence Visual Words

Given a codebook containing p visual words, two $p \times p$ matrices of co-occurrences are then computed for each superpixel s_i: the internal co-occurrence matrix and the neighbors co-occurrence matrix. The former one counts only co-occurrences of visual words that lye within s_i and therefore is symmetrical since the co-occurrence relation regardless the relative spatial position is inherently symmetrical. In the second one, the rows stand for the visual words inside s_i and the columns represent the ones within the adjacent superpixel ns_j, $j = 1, \ldots, n$, currently under consideration. It is worth to mention that the neighbors co-occurrence matrix is cumulative for all n adjacent superpixels. Notice that it is essential to define when a visual word is inside a superpixel in order to count the co-occurrences properly: an arbitrary visual word v_l lies within a superpixel s_x if and only if the central pixel of the grid cell c_l from which v_l was extracted lies within s_x when the grid and the segmentation are superimposed.

3.2.4 Vectorization

Once the matrices for s_i are computed they are vectorized so that s_i can be represented only by vectors. The symmetrical internal co-occurrence matrix results in a $p(p+1)/2$-dimensional vector because the diagonal elements are kept while the neighbors co-occurrence matrix becomes a p^2-dimensional vectors due to the non-symmetrical nature of the relation.

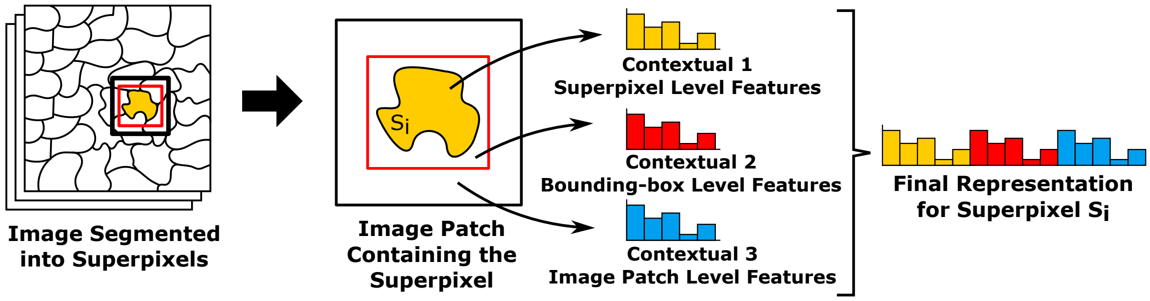


Image Segmented into Superpixels
Image Patch Containing the Superpixel
Si
Contextual 1 Superpixel Level Features
Contextual 2 Bounding-box Level Features
Contextual 3 Image Patch Level Features
Final Representation for Superpixel Si

Figure 3.3: The MCL 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 $\ell 2$-normalized features extracted from s_i, a small rectangular contextual area surrounding s_i and a large contextual area.

3.2.5 Concatenation

The final representation for s_i is the concatenation of the $\ell 2$-normalized vectors. Notice that the vectors are normalized separately.

3.3 Many Context Levels Representation

Most methods in the literature exploit either only a single level of context or a combination of local and global cues, which are claimed to be the most relevant ones since they determine what can be found in a scene (global) or the objects that interact with the target one. The main objective of the Many Context Levels (MCL) representation is to exploit a range of contextual levels, from the superpixel itself to an entire image patch containing it (which is regarded as global context once our entire dataset is sometimes just an image that is a mosaic composed of many patches).

A general overview of the proposed representation is shown in Figure 3.3. Given a superpixel s_i, we separately extract features concerning three contextual areas defined by different groups of pixels: (1) inside the region; (2) a small contextual area surrounding s_i; and (3) a larger contextual area. Small and larger contextual areas are limited by boxes centered in s_i and consider both internal and external pixels from the superpixel. The final representation is the concatenation of the $\ell 2$-normalized features extracted from all contextual levels. In spite of the simplicity of the proposed composition of contextual features, its strength lies on the exploitation of intermediate contextual levels, which are left out by the most methods.

Notice that the MCL is just an structure to compose non-contextual descriptors into a contextual one. For this reason, it provides flexibility on choosing the descriptor used to generate the feature vectors which are taken as input. There is only one constraint: the vectors output by the descriptor are to be fixed-size. Thereby, one could virtually choose any descriptor to extract features from the contextual regions, since most of them produce a fixed-size vector. For this work, the Convolutional neural networks (ConvNets) were chosen because of the impressive results they are achieving in a wide range of pattern recognition tasks and because the codification of context besides the use of graphical models is being more exploited recently.

3.3.1 Deep Features Extraction from Superpixels

An essential characteristic of the ConvNets that makes them suitable to encode context is that each layer is able to learn filters that enhance different visual semantic levels [Mostajabi et al., 2015]. The first convolutional layers emphasize low-level properties, such as 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 spatial relationships among them. Thus, ConvNets in their very nature are able to encode semantic context.

Nevertheless, an existing challenge concerning the feature extraction from arbitrary-shaped regions is that the main algorithms are designed for rectangular or squared image patches [dos Santos et al., 2012]. ConvNets 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 in this work. For creating a feature representation for a given region s_i, a square box around it is initially defined, which can be seen as an image patch I. The next step is to use a pre-trained ConvNet to create feature maps in different layers. Regardless of the ConvNet chosen, a convolutional layer with k filters produces k feature maps stacked so that there is 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 MCL method to make use of deep features which are intrinsically rich in semantic context, further improving MCL that already aggregates spatial context. Therefore, the final representation generated by the MCL 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

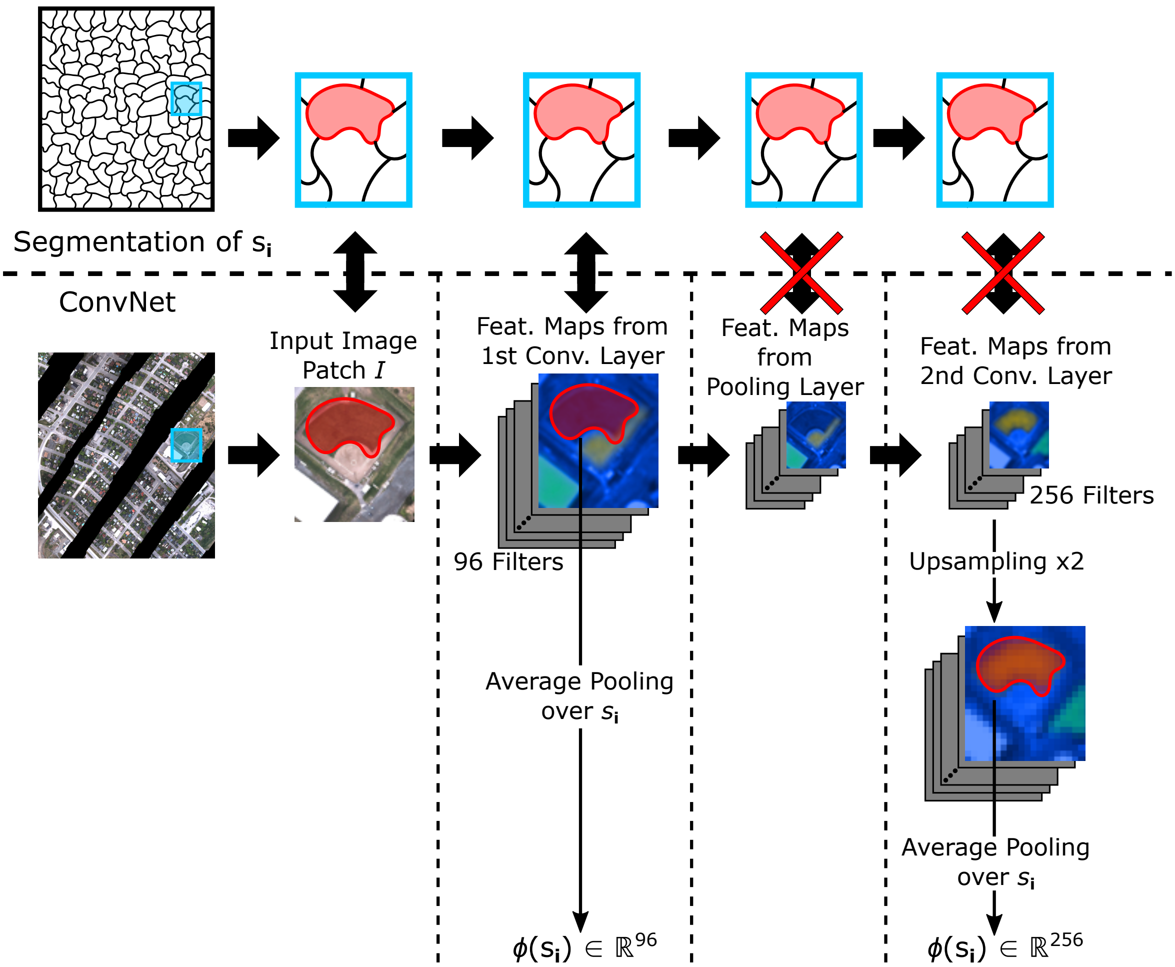

Segmentation of s_i
ConvNet
Input Image Patch I
Feat. Maps from 1st Conv. Layer
96 Filters
Average Pooling over s_i
$\phi(s_i) \in \mathbb{R}^{96}$
Feat. Maps from Pooling Layer
Feat. Maps from 2nd Conv. Layer
256 Filters
Upsampling x2
Average Pooling over s_i
$\phi(s_i) \in \mathbb{R}^{256}$

Figure 3.4: 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. This way, after each convolutional layer is possible to generate just one k-dimensional feature vector by average pooling the k feature maps over s_i. Every time that 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.

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 showed in Figure 3.4.

We have validated the proposed approach by using the well-known AlexNet [Krizhevsky et al., 2012]. Figure 3.5 illustrates the proposed strategy for deep contextual feature extraction by using this ConvNet.

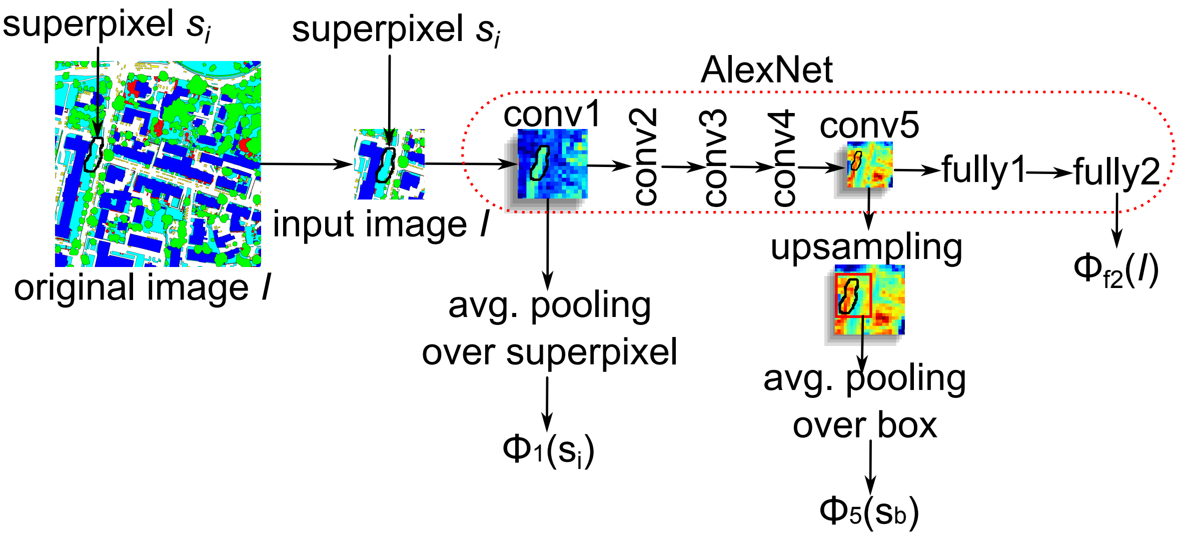


Figure 3.5: The proposed approach for deep contextual feature extraction with AlexNet. Given a superpixel s_i, an image patch I is created centering the superpixel and used as input for the AlexNet. The features are computed considering three levels of context (or three layers): $\phi_1(s_i)$, $\phi_5(s_b)$ and $\phi_{fc2}(I)$.

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 $\phi_5(s_b)$ computes features from a small context that should bring 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 tend to be more complex and contain more information, since they may represent whole buildings, streets, cars as well as the interactions among these. Specifically, the features are extracted from the fifth convolutional layer considering a small context with respect to the superpixel. The contextual area is delimited by a fixed-size bounding box surrounding s_i. 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 the 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 tree or a park. In other words, this layer may help to determine the presence of categories in the scene, i.e., it is responsible for imposing a constraint in the label space by eliminating classes which have no elements within the image patch I. In the proposed method, features are extracted from the last fully connected layer. The last level of context $\phi_{fc2}(I)$ only returns features values and not feature maps. At the end 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.

Notice that although the first, fifth and last layers were chosen, it is not required that those specific layers be selected to make the feature extraction work properly.

Chapter 4

Experimental Analysis

In the first section we present the experimentation protocol used to assess the proposed methods, which includes a description of the datasets, metrics used, statistical validation and training protocol. Section 4.2 brings the descriptions of the experiments performed to evaluate the methods on the grss_dfc_2014 dataset and the results achieved, while Section 4.3 shows the same informations for the ISPRS Potsdam one.

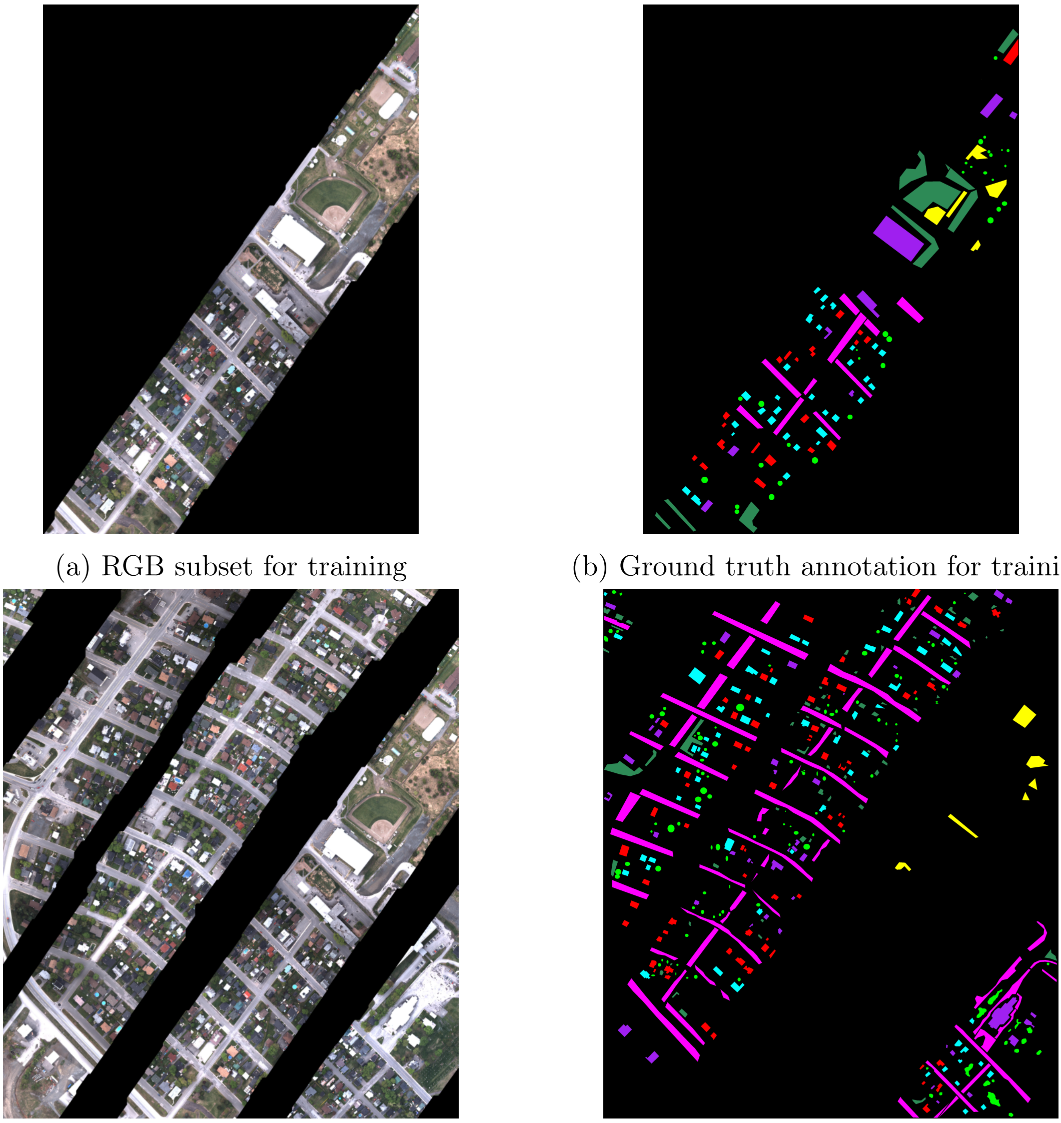
4.1 Experimental Protocol

4.1.1 Datasets

Two imbalanced multi-class datasets which are publicly available were selected for evaluating the effectiveness of the proposed methods: the grss_dfc_2014 and ISPRS Potsdam (ISPRS stands for International Society for Photogrammetry and Remote Sensing). Both datasets were chosen mainly due to the fact that they are widely used throughout the literature.

4.1.1.1 grss_dfc_2014

The first dataset was released for the IEEE GRSS Data Fusion Contest (DFC) in 2014. 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. Notice in Figure 4.1 that the color images acquired in distinct flight-lines are spatially disjoint, resulting in blank areas.



(a) RGB subset for training

(b) Ground truth annotation for training

(c) Whole RGB imagery for test

(d) Ground truth annotation for test

Figure 4.1: Images from grss_dfc_2014 used in the experiments.

All imagery were acquired and provided by Telops Inc. using sensors mounted on an airborne platform which overflew an urban area near Thetford Mines in Québec, Canada, on May 21st, 2013. Both sets of data were radiometrically and geometrically corrected posteriorly.

The ground truth was annotated into seven classes: road, tree, red roof, grey roof, concrete roof, vegetation and bare soil. The color used to annotate the pixels of each class can be seen in Figure 4.2 and the distribution of the number of pixels per class is showed in Table 4.1, from which it is clear the level of imbalance among classes. Additionally, there is an unclassified class which is used to annotate the remainder of the image that was not annotated as belonging to any of the seven thematic classes

Table 4.1: Class distribution in terms of pixels for grss_dfc_2014 dataset.

Images/Classes	Road	Trees	Red Roof	Grey Roof	Concrete Roof	Vegetation	Bare Soil
Training Image	19,79%	4,88%	8,20%	9,42%	17,22%	32,62%	7,87%
Test Image	55,73%	6,94%	9,42%	9,84%	7,55%	7,14%	3,39%
Entire Dataset	45,62%	6,36%	9,07%	9,72%	10,27%	14,30%	4,65%

as well as blank pixels. It is worth to mention that pixels annotated with this class are not taken into account during the training of the classifiers and neither to compute the metrics described in Subsection 4.1.5 for the grss_dfc_2014. Since the dataset was originally released for a contest, a specific subset containing just one flight-line is provided for training, resulting in an image with 2830×3989 pixels.

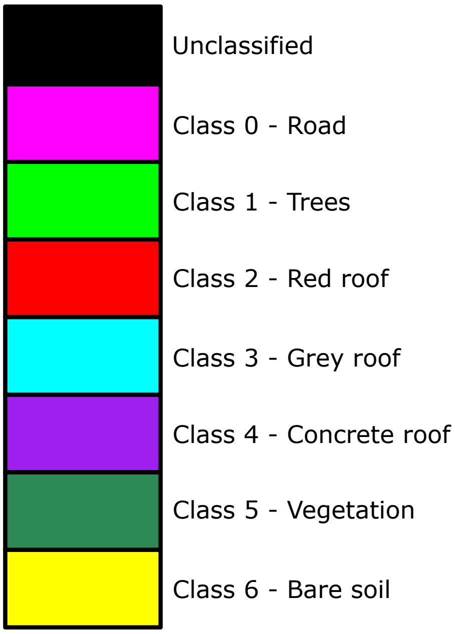


Figure 4.2: Colors and classes in which grss_dfc_2014 dataset was annotated.

Even though the contest aimed at encouraging the development of multisensor fusion, it is out of the scope of this work and, thereby, only the visible imagery of grss_dfc_2014 is going to be used in order to assess the proposed methods. Figure 4.1 shows the visible images used in the experiments and their respective ground truth annotation.

4.1.1.2 ISPRS Potsdam

The ISPRS Potsdam 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 data available for the Semantic Labeling Contest of the ISPRS.

Again, even though DSMs may be useful to improve classification results, the scope of this work is on visible images and, therefore only the VHR images were used, which are TIFF images with the following channel compositions: R-G-B, IR-R-G and R-G-B-IR. Since only usual low-level descriptors designed to work on RGB images are used for the experiments, only the composition R-G-B were tested.

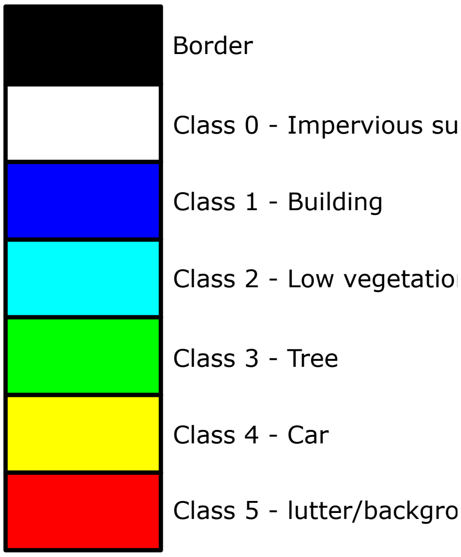


Figure 4.3: Colors and classes in which ISPRS Potsdam dataset was annotated.

The ground truth is provided for just 24 image patches and is annotated into 6 classes: impervious surfaces, building, low vegetation, tree, car and clutter/background. The colors used to annotate each class are presented in Figure 4.3 and the imbalance of the dataset can be seen from the class distribution in Table 4.2. It is worth to mention that the evaluation protocol specified for the dataset requires the annotation of the class clutter/background, which includes everything that looks different from the remaining objects (e.g. water bodies, containers, tennis courts and swimming pools). Since the intra-class variance of this class is very high, it impairs the classifier training and reduces the class separability. Additionally, because the dataset resolution is about four times higher than the grss_dfc_2014 one, the appearance of the objects is more heterogeneous and, consequently, the intra-class variance is also high for the other classes. All those factor together make this dataset very challenging.

Also regarding the evaluation of the classification results, it is important to highlight that there is a small border of black pixels surrounding each object annotated in

Table 4.2: Class distribution in terms of pixels for ISPRS Potsdam dataset.

Patches/Classes	Impervious Surfaces	Building	Low Vegetation	Tree	Car	Clutter/ Background
3_12	29,27%	25,53%	20,40%	21,33%	1,44%	2,03%
4_12	33,19%	35,16%	19,79%	7,45%	2,07%	2,33%
5_12	30,78%	50,61%	8,58%	5,78%	2,49%	1,76%
7_11	47,33%	29,55%	11,83%	8,07%	1,77%	1,45%
7_12	53,63%	31,80%	7,53%	4,00%	1,89%	1,16%
Entire Dataset	38,92%	34,55%	13,58%	9,27%	1,93%	1,75%

the ground truth. Such black pixels are not taken into account to compute the metrics presented in Subsection 4.1.5.

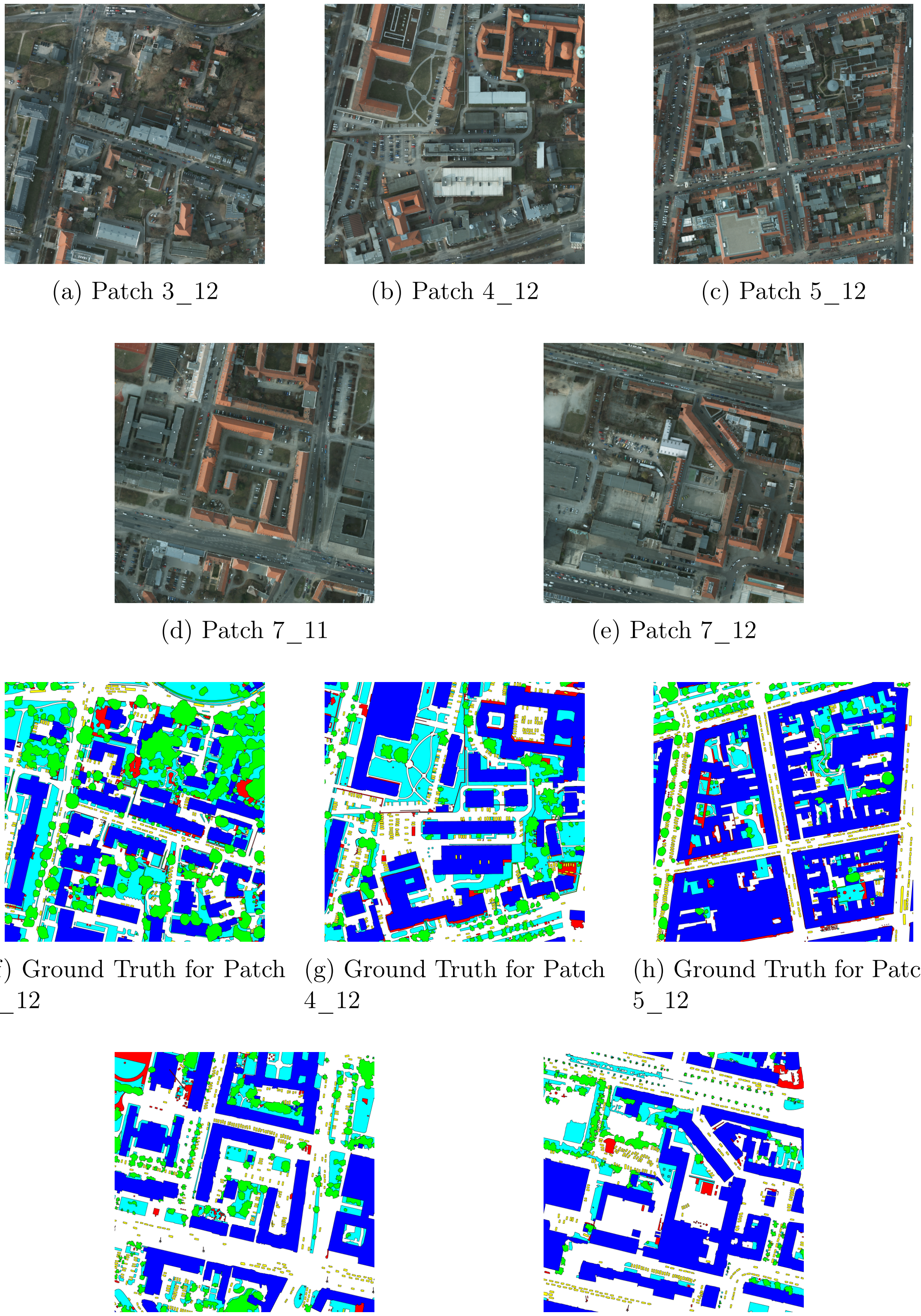
In our experiments, due to the large amount of descriptors extracted from each image, it would be infeasible to use all images to train many classifiers in a reasonable time, so that we randomly selected a smaller subset of 5 from the 24 annotated images to perform the experiments: 3_12, 4_12, 5_12, 7_11 and 7_12, showed in Figure 4.4.

4.1.2 Segmentation

All images were segmented into superpixels using SLICO, an adaptive version of the Simple Linear Iterative Clustering (SLIC) algorithm [Achanta et al., 2012]. SLIC is a good option since it performed better than several state-of-the-art superpixel methods according to the boundary recall and under-segmentation error metrics. Nevertheless, SLICO besides having a good adherence to the object boundaries also adaptively sets the compactness parameter, resulting in smooth regular-sized superpixels in both smooth and highly textured regions of the image. Such characteristics make SLICO an even better choice for this work, even though it does not guarantee the connectivity of any individual superpixel.

The grss_dfc_2014 training image was segmented initially into roughly 25000 superpixels, while the test image was divided into around 37500 superpixels of nearly the same size of the training ones (this number was chosen because the test image is about 50% bigger than the training one). All five ISPRS Potsdam images were segmented into around 30000 superpixels. These values were obtained empirically by trying to fit most of the object within one or more superpixels. Since the later dataset has a finer spatial resolution, less superpixels with bigger size are required to delineate objects.

Later on, these values were changed in steps of 5000 and 7500 for the grss_dfc_2014 training and test images, respectively, in order to assess the impact



(a) Patch 3_12

(b) Patch 4_12

(c) Patch 5_12

(d) Patch 7_11

(e) Patch 7_12

(f) Ground Truth for Patch 3_12

(g) Ground Truth for Patch 4_12

(h) Ground Truth for Patch 5_12

(i) Ground Truth for Patch 7_11

(j) Ground Truth for Patch 7_12

Figure 4.4: Image patches from ISPRS Potsdam used in the experiments.

of the segmentation scale chosen on the classification results. The number of superpixels was not modified for the second dataset, once it takes much longer to train a model with a specific configuration on it due to the huge volume of training data as compared to the first dataset. Thus, evaluating how the changes on the segmentation affects the classification results on ISPRS Potsdam dataset is a time-consuming process.

Aiming at providing more data about the resulting segmentation, some measurements were made on the superpixels: the area, width and height of the smallest rectangle that encloses it and the number of pixels within the superpixel. More precisely, denoting the segmentation image (i.e., the image containing the label of each pixel that determines to which superpixel it belongs) as a set of n disjoint superpixels $S = \{S_i \,|\, \bigcup_{i=1}^{n} S_i = S \; and \; \bigcap_{i=1}^{n} S_i = \emptyset\}$, where each superpixel S_i is itself a set $S_i = \{(x,y) \,|\, (x,y,t) \in I \; and \; f(x,y,t) = i\}$, where (x,y,t) is a pixel from the original image I, x and y are the coordinates of the pixel on the horizontal and vertical axis, respectively, t is the tone of the pixel and $f(x,y,t)$ is the function that maps a pixel to a superpixel based on some criteria, the measurements for a specific superpixel S_i can be defined as:

$$width(S_i) = \max_{\{(x,y) \in S_i\}} (x) - \min_{\{(x,y) \in S_i\}} (x) \tag{4.1}$$

$$height(S_i) = \max_{\{(x,y) \in S_i\}} (y) - \min_{\{(x,y) \in S_i\}} (y) \tag{4.2}$$

$$area(S_i) = width(S_i) \times height(S_i) \tag{4.3}$$

$$num.pixels(S_i) = |S_i| \tag{4.4}$$

The results of the measurements described in Equations 4.1, 4.2, 4.3 and 4.4 are reported in Tables 4.3 and 4.4 for the grss_dfc_2014 dataset and in Table 4.5 for the ISPRS Potsdam dataset.

4.1.3 Classifier and Training Protocol

We used the features extracted to train a Support Vector Machine (SVM) classifier, which is known for handling high-dimensional data. A Radial-Basis Function (RBF) kernel is used to train the SVM on the grss_dfc_2014 dataset because the background samples are not considered for training the classifier, resulting in few samples (around 2000). Nevertheless, the same is not true for the ISPRS dataset which requires back-

Table 4.3: Measurements of the size of superpixels in the training image of grss_dfc_2014.

Measure (in pixels)	Num. Superpixels	Min.	Max.	Avg.
Area (Width x Height)	20000	13 × 27 = 351	53 × 38 = 2014	629.32
	25000	12 × 24 = 288	68 × 29 = 1972	474.68
	30000	14 × 14 = 196	45 × 35 = 1575	384.17
Width	20000	13	53	24.83
	25000	11	68	21.56
	30000	12	45	19.42
Height	20000	17	47	24.98
	25000	13	50	21.68
	30000	13	43	19.49
Num. Pixels	20000	282	1238	577.02
	25000	227	888	441.09
	30000	188	727	361.61

Table 4.4: Measurements of the size of superpixels in the test image of grss_dfc_2014.

Measure (in pixels)	Num. Superpixels	Min.	Max.	Avg.
Area (Width x Height)	30000	22 × 14 = 308	60 × 41 = 2460	698.58
	37500	28 × 10 = 280	44 × 53 = 2332	577.62
	45000	13 × 16 = 208	48 × 73 = 3504	465.87
Width	30000	12	74	26.20
	37500	10	63	23.83
	45000	10	53	21.38
Height	30000	12	58	26.31
	37500	10	53	23.92
	45000	9	73	21.49
Num. Pixels	30000	275	1336	530.94
	37500	220	990	443.95
	45000	183	1125	362.98

ground annotation, resulting in around 120000 training samples. Once the training process using RBF kernel is quadratic in the number of samples ($O(n^2)$), its usage for the latter dataset would be extremely time-consuming so that we choose just a linear kernel and normalized the features to unit mean and zero variance in order to speed up convergence.

Additionally, experiments were performed using an improved version of Gradient Boosting Machine (GBM) called eXtreme Gradient Boosting (XGBoost), which is an end-to-end system to train an ensemble of trees [Chen and Guestrin, 2016]. Its main

Table 4.5: Measurements of the size of superpixels in the ISPRS Potsdam dataset segmented in 30000 superpixels.

Measure (in pixels)	Image	Min.	Max.	Avg.
Area (Width x Height)	3_12	12 x 61 = 732	99 x 61 = 6039	1643.22
	4_12	13 x 46 = 598	76 x 112 = 8512	1629.31
	5_12	26 x 26 = 676	57 x 121 = 6897	1637.58
	7_11	27 x 27 = 729	45 x 110 = 4950	1610.06
	7_12	33 x 22 = 726	104 x 50 = 5200	1626.96
Width	3_12	12	127	40.70
	4_12	12	143	40.39
	5_12	16	126	40.51
	7_11	17	91	40.11
	7_12	18	104	40.59
Height	3_12	16	99	40.30
	4_12	12	120	40.28
	5_12	19	121	40.37
	7_11	18	110	40.09
	7_12	20	82	39.99
Num. Pixels	3_12	601	2555	1232.75
	4_12	602	3903	1236.01
	5_12	601	2665	1234.23
	7_11	605	2758	1233.38
	7_12	604	2819	1233.00

advantages include scalability, invariance to feature normalization and robustness to redundant features, since it selects the feature which brings more gain to the model in each step of tree growing. Such characteristics are making XGBoost very popular: it is used by big companies like Google and most of the winning solutions of Kaggle competitions are based on XGBoost.

It is important to highlight that the hyper-parameter optimization for both classifiers was done via grid search along with 5-fold Cross-validation. The ranges used to optimize each hyper-parameter is shown on Table 4.6. Once performing grid search using just one grid for XGBoost would be infeasible due to large number of hyper-parameters (indeed one would have to build $4 \times 3 \times 5 \times 3 \times 3 \times 5 \times 5 = 2700$ entire models besides the last one which is built using 5-fold Cross-validation in each of the 5000 iterations), 5 grids were used: *Max Tree Depth* and *Min Child Weight*; *Min Loss Reduction* γ; *Random Subsampling* and *Features Random Subsampling*; *Regularizer* λ; *Max Delta Step*. After optimizing these hyper-parameters, they were used to build a model in 5000 iterations. A 5-fold Cross-validation is applied to assess the accuracy of the model during its training, which is stopped when there is no loss reduction above a threshold ϵ and, thereby, the best number of iterations is determined. Unless otherwise stated throughout Sections 4.2 and 4.3, this process is employed in all experiments.

Table 4.6: Hyperparameters optimized in SVM and XGBoost.

Classifier	Hyperparameter	Range	Num. of Values from Range	Increment
SVM	Soft-margin (C)	1×10^{-4} to 1×10^{1}	6	Log.
SVM	Kernel Gamma (γ)	1×10^{-2} to 1×10^{5}	8	Log.
XGBoost	Max Tree Depth	2 to 8	4	Linear
XGBoost	Min Child Weight to Split	1 to 5	3	Linear
XGBoost	Min Loss Reduction to Split (γ)	$\{0\} \cup \{1 \times 10^{-2}$ to $1 \times 10^{1}\}$	5	Log.
XGBoost	Random Subsamplig	0.7 to 0.9	3	Linear
XGBoost	Features Random Subsampling	0.7 to 0.9	3	Linear
XGBoost	Regularizer (λ)	$\{1 \times 10^{-5}\} \cup$ $\{1 \times 10^{-2}$ to $1 \times 10^{1}\}$	5	Log.
XGBoost	Max Delta Step	0 to 8	5	Linear
XGBoost	Num of Iterations	1 to 5000	5000	Linear

Since the first dataset was originally released for a contest, a specific training subset is provided to train a classifier that should then be able to annotate the test samples in order to generate a map which is evaluated.

For the Potsdam ISPRS, we performed a validation protocol similar to the 5-fold cross-validation, except that each set is composed exclusively of samples from just one of the images. The validation was done in this way because we need to annotate samples from an unique test image in order to build an entire map to be assessed at the end of process.

4.1.4 Statistical Validation

Once the grss_dfc_2014 dataset provides a specific training subset which is meant to be completely used to train the classifier, there is no randomness to evaluate the statistic significance of the results. Thus, after generating the land cover map based on the predictions of the classifier, each of the metrics described on Subsection 4.1.5 is pixel-wisely calculated and reported.

For the ISPRS Potsdam dataset, the sample mean is calculated for all metrics from Subsection 4.1.5 which are in turn computed individually and pixel-wisely for each of the five maps generated. Then, the sample standard deviation is calculated for each metric and used to compute the confidence intervals. Once the entire sample is composed of just five measurements, the confidence intervals are calculated using the Student's t-distribution with 4 d.f. and 95% of confidence due to its suitability for small samples.

4.1.5 Metrics

The results achieved are reported in terms of overall accuracy (Ovr. Acc.), average accuracy (Avg. Acc.) and Cohen's Kappa Index (κ), which are computed over a confusion matrix built from the classification results.

A confusion matrix M is a square table that presents in a organized way four distinct types of counts for each class C_t $(t = 1, \ldots, T)$ considered in the domain of the classification task: the number of samples that were correctly recognized as belonging to a class C_t (true positives - TP); the number of samples correctly recognized as not belonging to C_t (true negatives - TN); the number of samples which actually belong to C_t but were incorrectly classified into another class $C_u, u \neq t$ (false negatives - FN); and the number of samples from other classes which were assigned to C_t (false positives - FP). Such counts, shown in Table 4.7 are essential to evaluate the correctness of the classification result as many metrics are computed from them [Sokolova and Lapalme, 2009].

Table 4.7: Example of a multi-class confusion matrix M.

		Predicted			
	Classes	C_1	C_2	$\cdots$	C_T
Actual	C_1	$m_{1,1}$	$m_{1,2}$	$\cdots$	$m_{1,T}$
	C_2	$m_{2,1}$	$m_{2,2}$	$\cdots$	$m_{2,T}$
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	C_T	$m_{T,1}$	$m_{T,1}$	$\cdots$	$m_{T,T}$

Given the generic confusion matrix M shown in Table 4.7, the four counts aforementioned can be obtained for the class t via the following sums:

$$TP_t = m_{t,t} \tag{4.5}$$

$$FN_t = \sum_{u \neq t,\ u=1,\ldots,T} m_{t,u} \tag{4.6}$$

$$FP_t = \sum_{u \neq t,\ u=1,\ldots,T} m_{u,t} \tag{4.7}$$

$$TN_t = \sum_{u,v=1,\ldots,T} m_{u,v} - TP_t - FN_t - FP_t \tag{4.8}$$

From the counts expressed by Equations 4.5, 4.6, 4.7 and 4.8, one can compute many metrics in order to assess distinct aspects of the classification performed, including those which were chosen for this work.

The overall accuracy is a general and widely used metric that gives the overall effectiveness of a classifier by measuring the inter-rater agreement regardless class imbalance or chance of random agreement, i.e., it evaluates the agreement between the labels only correctly predicted by the classifier and the ground truth annotations with relation to all predictions made without taking into account any class information or the likelihood of agreeing at random. The overall accuracy is expressed as:

$$OverallAccuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{4.9}$$

Notice that as the overall accuracy does not considers class information, Equation 4.9 is not expressed using class indices and, thus, it is simply the sum of the main diagonal of the confusion matrix M over the sum of all entries of M [Sokolova and Lapalme, 2009].

However, in a scenario where there is a strong imbalance among the number of samples of each class, the overall accuracy is not very informative. This is because even if the classifier only predicts correctly the samples from one class and that is the larger class, Equation 4.9 will indicate a good overall effectiveness while the classifier learned just how to identify one class, what is almost nothing for a classification problem.

For this reason, average accuracy was also used in this work. The average accuracy is the sum of accuracies calculated individually for each of the T classes divided by the number of classes:

$$AverageAccuracy = \frac{\sum_{t=1}^{T} \frac{TP_t + TN_t}{TP_t + FN_t + FP_t + TN_t}}{T} \tag{4.10}$$

Although the average accuracy can cope with the class imbalance, it does not take into account the likelihood of the prediction of the classifier is made at random, i.e., not based on any latent process that maps the input features to the class label.

Therefore, Cohen's Kappa index (represented by the Greek letter κ and also known as Kappa statistic or just Kappa index) is taken as reference to compare the effectiveness of the methods, since it is a statistic that measures the level of agreement between the predictions of the classifier and the ground truth annotations taking into account the likelihood that they agree by chance [Cohen, 1960]. Kappa statistic can be calculated as:

$$\hat{\kappa} = \frac{p_a - p_e}{1 - p_e} \tag{4.11}$$

where

$$p_a = \frac{\sum_{t=1}^{T} m_{t,t}}{l} \tag{4.12}$$

$$p_e = \sum_{t=1}^{T} \frac{\sum_{u=1}^{T} m_{t,u}}{l} \times \frac{\sum_{u=1}^{T} m_{u,t}}{l} \tag{4.13}$$

and l is the total number of ratings or predictions. As any correlation measure, kappa ranges from -1 to 1. A possible understanding on how good the value of the Kappa statistic is can be seen in Table 4.8, which was proposed by Landis and Koch [1977]. The most relevant and widely accepted thresholds are that 0 represents a random classification, between 0.6 and 0.8 is a good result and above 0.8 stands for an almost perfect classification.

Table 4.8: Possible reference for Kappa statistic proposed by Landis and Koch [1977].

Value of Kappa Statistic	Strength of Agreement
-1.00 $\leq \kappa <$ 0.00	Poor
0.00 $\leq \kappa \leq$ 0.20	Slight
0.20 $< \kappa \leq$ 0.40	Fair
0.40 $< \kappa \leq$ 0.60	Moderate
0.60 $< \kappa \leq$ 0.80	Substantial
0.80 $< \kappa \leq$ 1.00	Almost Perfect

4.1.6 Baselines

We have chosen Vargas et al. [2015] descriptor as the baseline, which is a representation for superpixels that encode context and is based on BIC color descriptor. In that work, each superpixel of the image is described by a histogram of visual elements, using the mid-level representation generated by the Bag of Visual Words (BoVW). Then, contextual information is aggregated by concatenating the superpixel description with a combination of the histograms of its neighbors. It appears in Sections 4.2 and 4.3 under the name Vargas.

Additionally, low-level descriptors computed over the superpixels are used as a baseline which appears in Sections 4.2 and 4.3 under the name No-ctxt.

4.2 Results on grss_dfc_2014 Dataset

In this section, the experiments carried out and the results achieved on the grss_dfc_2014 dataset by each of the three proposed methods are presented. A brief discussion follows the description of all experiments.

4.2.1 Results of the Experiments with the Star Descriptor

Initially, we have performed experiments to find the best configuration of Star and the baselines. Two global color descriptors were selected according to dos Santos et al. [2010] using a trade-off criteria between dimensionality (and therefore efficiency) and accuracy: *Border/Interior Pixel Classification* (BIC) and *Color Coherence Vectors* (CCV). Although any kind of characteristic could be used instead of color, it usually achieves better accuracy in Remote Sensing [dos Santos et al., 2010]. Both global low-level and mid-level representations were evaluated. For the mid-level representation, the number of visual words in the codebook and the size of the cell used for grid sampling were initially fixed at 128 and 5x5, respectively.

Since Star also requires an edge descriptor, besides the color descriptors we selected two texture descriptors because they are able to capture sharp tone changes which are typical of objects' borders [Silva et al., 2013] as well as providing complementary information: *Quantized Compound Change Histogram* (QCCH) and the descriptor proposed by Unser [1986], which we called just Unser.

All three composition operations were evaluated along with both global low-level and mid-level representations. The best configurations with respect to the edge composition operation for Star, vertex composition operation for Vargas and vertex descriptor for No-ctxt are presented in Tables 4.9, 4.10, 4.11 and 4.12.

The first thing to notice is the best configurations of each method. Star performed better using BIC, QCCH, max and sum pooling as vertex descriptor, edge descriptor, vertex composition operation and edge composition operation, respectively, for low-level representation and XGBoost, achieving up to 0.747 in Kappa statistic which could be regarded as substantial according to the reference presented in Table 4.8. The good effectiveness was repeated also for mid-level representation and both SVM

Table 4.9: Best configurations of Star and baselines using SVM with RBF kernel and mid-level representation.

Method	Vertex Desc.	Edge Desc.	Vertex Comp. Op.	Edge Comp. Op.	Ovr. Acc.	Avg. Acc.	Kappa
Star	BIC	BIC	Avg	Max	73,22%	92,35%	0,632
			Max	Max	73,57%	92,45%	0,635
			Sum	Max	73,22%	92,35%	0,632
		CCV	Avg	Max	73,62%	92,46%	0,637
			Max	Max	74,07%	92,59%	0,642
			Sum	Max	73,62%	92,46%	0,637
		Unser	Avg	Sum	74,51%	92,72%	0,645
			Max	Max	75,49%	93,00%	0,658
			Sum	Sum	74,51%	92,72%	0,645
		QCCH	Avg	Sum	78,98%	93,99%	0,703
			Max	Sum	80,86%	94,53%	0,727
			Sum	Sum	78,98%	93,99%	0,703
	CCV	BIC	Avg	Max	68,62%	91,03%	0,574
			Max	Max	71,42%	91,83%	0,606
			Sum	Max	68,62%	91,03%	0,574
		CCV	Avg	Max	69,18%	91,19%	0,581
			Max	Sum	71,48%	91,85%	0,607
			Sum	Max	69,18%	91,19%	0,581
		Unser	Avg	Sum	72,02%	92,01%	0,614
			Max	Sum	72,52%	92,15%	0,618
			Sum	Sum	72,02%	92,01%	0,614
		QCCH	Avg	Sum	75,58%	93,02%	0,657
			Max	Sum	75,78%	93,08%	0,661
			Sum	Sum	75,58%	93,02%	0,657
Vargas	BIC	-	Max	-	75,36%	92,96%	0,660
	CCV	-	Max	-	72,99%	92,28%	0,624
No-ctxt	BIC	-	-	-	71,47%	91,85%	0,609
	CCV	-	-	-	66,61%	90,46%	0,544

and XGBoost. The only exception was the combination of low-level representation and SVM that achieved better results with Unser as edge descriptor instead of QCCH. The best configuration of Vargas includes BIC and sum pooling as vertex descriptor and composition operation combined with low-level representation and SVM, achieving up to 0.702 in Kappa. In general, all configurations of Vargas using BIC performed better and sum pooling produced better results along with low-level representations while max pooling fitted better to the mid-level ones. The best result of No-ctxt was 0.641 of Kappa, produced by using BIC, low-level representation and XGBoost. BIC was also the best descriptor for No-ctxt, that achieved better results using low-level

Table 4.10: Best configurations of Star and baselines using SVM with RBF kernel and low-level descriptors.

Method	Vertex Desc.	Edge Desc.	Vertex Comp. Op.	Edge Comp. Op.	Ovr. Acc.	Avg. Acc.	Kappa
Star	BIC	BIC	Avg	Max	76,90%	93,40%	0,677
			Max	Max	77,88%	93,68%	0,686
			Sum	Max	76,90%	93,40%	0,677
		CCV	Avg	Sum	73,61%	92,46%	0,634
			Max	Sum	74,11%	92,60%	0,640
			Sum	Sum	73,61%	92,46%	0,634
		Unser	Avg	Sum	80,11%	94,32%	0,714
			Max	Sum	80,59%	94,45%	0,722
			Sum	Sum	80,11%	94,32%	0,714
		QCCH	Avg	Sum	75,71%	93,06%	0,655
			Max	Sum	74,89%	92,82%	0,643
			Sum	Sum	75,71%	93,06%	0,655
	CCV	BIC	Avg	Max	67,34%	90,67%	0,552
			Max	Sum	71,55%	91,87%	0,604
			Sum	Max	67,34%	90,67%	0,552
		CCV	Avg	Max	68,82%	91,09%	0,573
			Max	Max	71,66%	91,90%	0,609
			Sum	Max	68,82%	91,09%	0,573
		Unser	Avg	Max	74,81%	92,80%	0,648
			Max	Sum	75,00%	92,86%	0,647
			Sum	Max	74,81%	92,80%	0,648
		QCCH	Avg	Sum	78,50%	93,86%	0,696
			Max	Sum	77,77%	93,65%	0,687
			Sum	Sum	78,50%	93,86%	0,696
Vargas	BIC	-	Sum	-	78,83%	93,95%	0,702
	CCV	-	Max	-	75,16%	92,90%	0,645
No-ctxt	BIC	-	-	-	73,49%	92,43%	0,631
	CCV	-	-	-	68,99%	91,14%	0,573

representations.

Comparing the four tables (4.9, 4.10, 4.11 and 4.12), one can easily notice that BIC was the best vertex descriptor, what complies with the results presented by dos Santos et al. [2010] that compared BIC to several other color descriptors. The same authors showed that texture descriptors are not as effective as the color-based ones in RSIs classification and that is the reason to use them only as edge and not as vertex descriptors. Nevertheless, one can see from the aforementioned tables that texture descriptors are able to leverage the discriminative power of the color ones by providing complementary information: when used as edge descriptors, QCCH was better in 75%

Table 4.11: Best configurations of Star and baselines using XGBoost and mid-level representation.

Method	Vertex Desc.	Edge Desc.	Vertex Comp. Op.	Edge Comp. Op.	Ovr. Acc.	Avg. Acc.	Kappa
Star	BIC	BIC	Avg	Max	75,78%	93,08%	0,661
			Max	Max	76,12%	93,18%	0,665
			Sum	Max	75,78%	93,08%	0,661
		CCV	Avg	Max	74,87%	92,82%	0,651
			Max	Max	74,92%	92,83%	0,650
			Sum	Max	74,87%	92,82%	0,651
		Unser	Avg	Max	79,66%	94,19%	0,708
			Max	Sum	80,09%	94,31%	0,713
			Sum	Max	79,66%	94,19%	0,708
		QCCH	Avg	Sum	80,62%	94,46%	0,722
			Max	Sum	80,82%	94,52%	0,722
			Sum	Sum	80,62%	94,46%	0,722
	CCV	BIC	Avg	Sum	74,23%	92,64%	0,641
			Max	Sum	75,04%	92,87%	0,650
			Sum	Sum	74,23%	92,64%	0,641
		CCV	Avg	Max	72,88%	92,25%	0,623
			Max	Max	75,12%	92,89%	0,650
			Sum	Max	72,88%	92,25%	0,623
		Unser	Avg	Sum	77,01%	93,43%	0,671
			Max	Sum	78,35%	93,81%	0,688
			Sum	Sum	77,01%	93,43%	0,671
		QCCH	Avg	Sum	75,82%	93,09%	0,658
			Max	Max	77,22%	93,49%	0,673
			Sum	Sum	75,82%	93,09%	0,658
Vargas	BIC	-	Max	-	73,88%	92,54%	0,632
	CCV	-	Max	-	71,05%	91,73%	0,591
No-ctxt	BIC	-	-	-	71,73%	91,92%	0,608
	CCV	-	-	-	69,09%	91,17%	0,568

of the times while Unser produced higher Kappa statistics in the other 25%. With respect to the vertex composition operation, max pooling is often the best option (about 80% of the times), even though sum pooling sometimes outputs higher accuracies mainly for low-level representations. For the edge composition operation, there is a balance between max and sum pooling with respect to the type of representation (mid/low-level), but color descriptors are prone to yield better results when combined via max pooling while the texture gives good maps using sum pooling.

Although not exploited by Vargas et al. [2015], using color descriptors extracted globally from each superpixel may be useful to generate even better land cover maps. In

Table 4.12: Best configurations of Star and baselines using XGBoost and low-level descriptors.

Method	Vertex Desc.	Edge Desc.	Vertex Comp. Op.	Edge Comp. Op.	Ovr. Acc.	Avg. Acc.	Kappa
Star	BIC	BIC	Avg	Max	75,36%	92,96%	0,654
			Max	Max	77,48%	93,56%	0,680
			Sum	Max	75,36%	92,96%	0,654
		CCV	Avg	Max	75,24%	92,93%	0,653
			Max	Max	75,82%	93,09%	0,661
			Sum	Max	75,24%	92,93%	0,653
		Unser	Avg	Sum	82,08%	94,88%	0,738
			Max	Sum	82,02%	94,86%	0,737
			Sum	Sum	82,08%	94,88%	0,738
		QCCH	Avg	Sum	82,53%	95,01%	0,744
			Max	Sum	82,67%	95,05%	0,747
			Sum	Sum	82,53%	95,01%	0,744
	CCV	BIC	Avg	Max	75,98%	93,14%	0,660
			Max	Max	75,95%	93,13%	0,660
			Sum	Max	75,98%	93,14%	0,660
		CCV	Avg	Sum	72,54%	92,15%	0,619
			Max	Max	73,62%	92,46%	0,632
			Sum	Sum	72,54%	92,15%	0,619
		Unser	Avg	Max	81,79%	94,80%	0,733
			Max	Max	81,96%	94,85%	0,735
			Sum	Max	81,79%	94,80%	0,733
		QCCH	Avg	Sum	81,61%	94,75%	0,734
			Max	Max	82,34%	94,95%	0,741
			Sum	Sum	81,61%	94,75%	0,734
Vargas	BIC	-	Sum	-	76,44%	93,27%	0,667
	CCV	-	Max	-	76,07%	93,16%	0,658
No-ctxt	BIC	-	-	-	74,37%	92,68%	0,641
	CCV	-	-	-	71,12%	91,75%	0,599

the experiments reported in Tables 4.9, 4.10, 4.11 and 4.12, such approach often output higher values for each metric used in assessment, except some specific cases. That is true mainly for the XGBoost classifier and for the baselines taken into account. Also concerning the classifier, XGBoost performed better than SVM with RBF kernel most of the times and the main reason is that XGBoost performs a kind of feature selection during the tree growing process, when the training algorithm selects the feature that produces more gain to the model to grow the tree in each step of the current iteration. Moreover, once each level of the trees is a split over a specific feature, every tree grown represents a decision based on a relationship among many features. This allows for selecting the best features or sets of them from a vector that contains a lot of redundancy and noise.

Also concerning these four tables, it is important to notice that the sum and average pooling produced the same results when used as both vertex and edge composition operations. This is due to their similar nature: they are distinct only by a scale factor. Once XGBoost is invariant to scale transformations and SVM with RBF kernel also may be invariant under some specific constraints [Abe, 2003], the results were exactly the same. Therefore, we have chosen the sum rather than the average pooling as a best result once it saves the division operation while producing exactly the same accuracy. Nevertheless, it is worth to highlight that it is not true for all classifier nor all kernel used with SVM.

Another important aspect to find out the best settings for Star is to determine the best number of visual words in the codebook used to generate the mid-level representation for the vertices. Even though it is often assumed that the more the number of visual words is, the higher will be the accuracy achieved, it is not true in this case. It turns out that a large codebook will reduce the likelihood of two or more of the same visual word being within the same superpixel once it is a small area as compared to the entire image. Consequently, sparse histograms which are very similar to each other will be generated and therefore low accuracy is expected for both classifiers.

To evaluate the behavior of the methods when the size of the codebook changes, the best configurations of Star, Vargas and No-ctxt in Tables 4.9 and 4.11 were used to generate maps using 64, 256 and 512 visual words besides the 128 initial ones. The experiments carried out confirmed the hypothesis stated above and the results are presented in Figures 4.5 (namely, Star using BIC and max pooling for the vertex descriptor and QCCH and sum pooling for the edge one, Vargas using BIC and max pooling and No-ctxt using BIC) and 4.6 (namely, Star using BIC, max pooling, QCCH and sum pooling, Vargas using BIC and sum pooling and No-ctxt with BIC). From the charts one can see that the best number of visual words is 128 for all methods using both classifiers, except No-ctxt with XGBoost that was slightly better with a codebook composed of 256 visual words. These numbers are much smaller than the size of codebooks traditionally found in literature, often above 500 visual words.

Once a region-based approach is employed on this work, it is essential to assess the impact of varying the number of superpixels on the results. It is even more important to do this analysis due to the nature of superpixels which are much more homogeneous in shape and size than traditional regions, what results in some objects being well delineated by just one superpixel while others are represented by many of them, depending on the scale of the objects depicted on the image. To do such analysis,

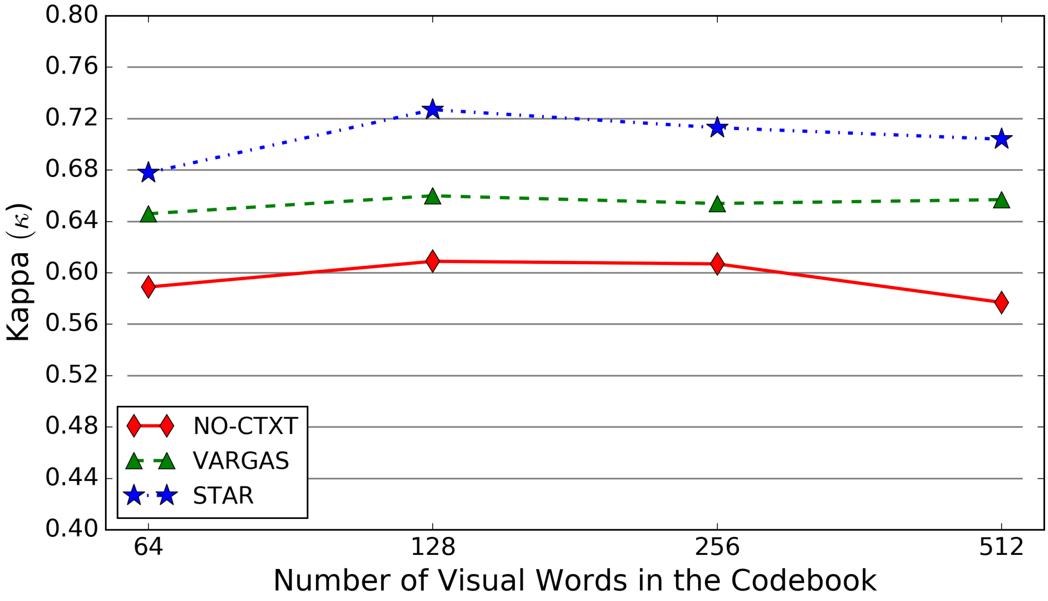


Figure 4.5: Impact of changing the number of visual words for the mid-level representations with SVM.

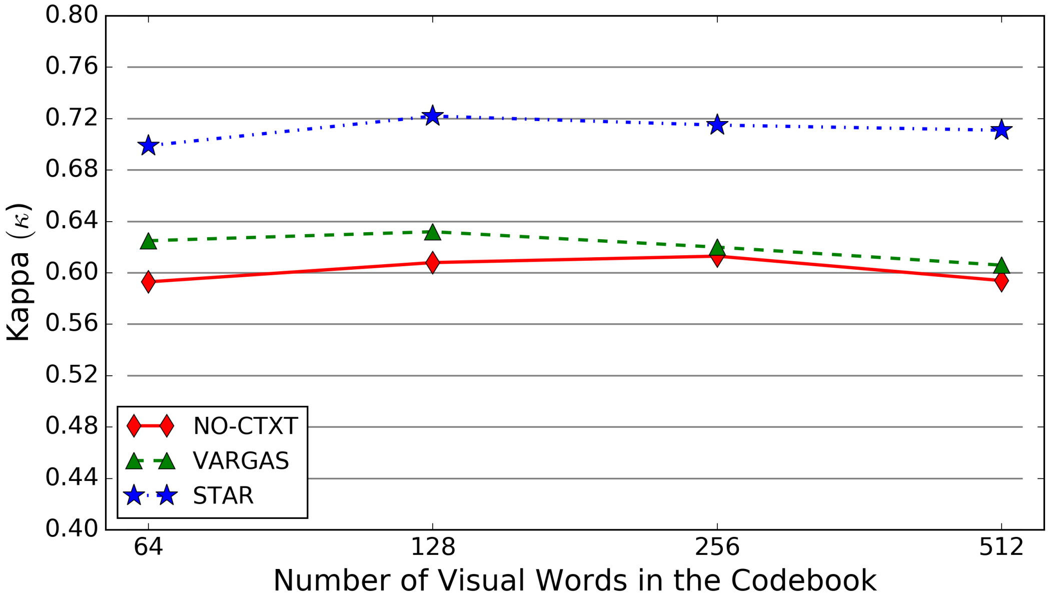


Figure 4.6: Impact of changing the number of visual words for the mid-level representations with XGBoost.

the best configurations for Star, Vargas and No-ctxt from Table 4.10 and the chart on Figure 4.5 and the best configurations from Table 4.12 and Figure 4.6 were used to generate maps now using also 30000 and 45000 as the parameter number of superpixels for SLICO. Namely, the configurations used with SVM were the Star using mid-level representation, BIC with max pooling for vertex descriptor and QCCH with sum pooling for the edge one, Vargas using low-level representation with BIC and sum pooling and No-ctxt using low-level representation with BIC. For XGBoost, the configurations were also Star with BIC, max pooling, QCCH and sum pooling, Vargas with BIC and sum pooling and No-ctxt with BIC, but all of them using mid-level representation.

It is expected that small numbers as the parameter result in large superpixels often containing more than one object while large values will generate very small superpixels and, thus, most of the objects will be oversegmented. The former case implies that the low-level descriptors chosen will capture visual cues from a mixture of objects what can be considered a kind of noise that impairs the classifier. On the other hand, the second case will result in superpixels with little information to be extracted by the descriptors and, therefore, poor representations. Thus, an intermediate value is supposed to yield the best results. The results of the experiments are shown in Figures 4.7 and 4.8.

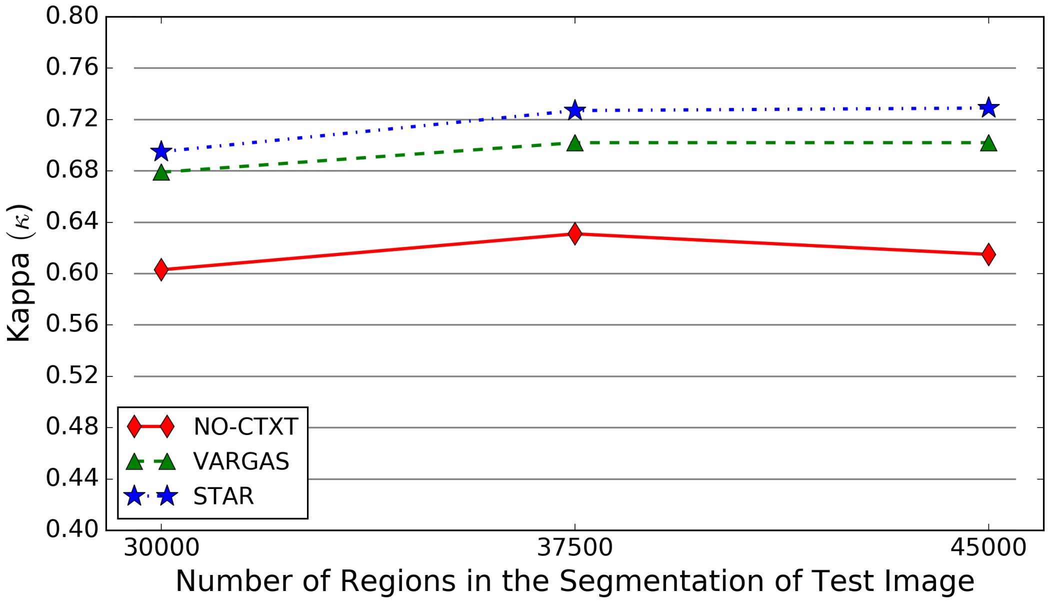


Figure 4.7: Impact of changing the number of superpixels used to segment the test image in the results of the best configurations of the methods with SVM.

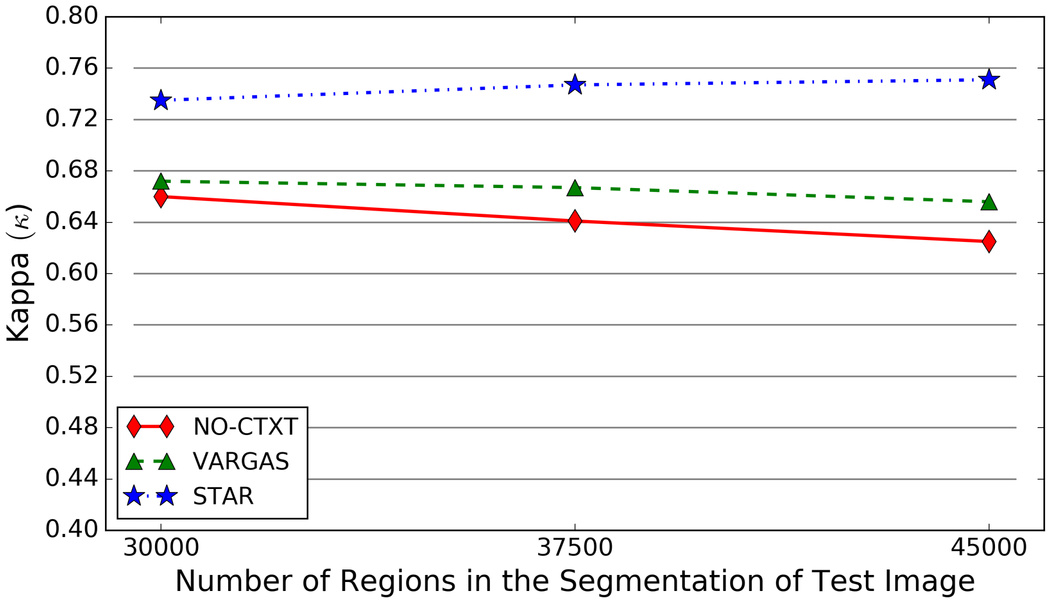


Figure 4.8: Impact of changing the number of superpixels used to segment the test image in the results of the best configurations of the methods with XGBoost.

From these charts it is possible to see that Star performed better with the number of superpixels of SLICO set to 45000 for the test image, even though the difference from 37500 superpixels is very small (indeed the difference is an improvement of just 0.002 and 0.001 in terms of Kappa using SVM and XGBoost, respectively). All the baselines achieved their top results using either 30000 or 37500 for the parameter, showing that the usage of an edge descriptor makes Star more robust to variations like those on the segmentation.

After evaluating many aspects of the proposed method and the baselines, a question may be raised: Does the usage of context benefit all kinds of objects in RSIs? One may wonder that the answer is obviously no, because even though every single object exists in a context, the upper perspective of RSIs is not favorable to capture the context of all objects. For instance, although a tree in the middle of a dense forest is in a context, its visual appearance is very similar to all trees next to it, even taking their context into account. Thereby, in order to address this question, the results for the best configurations found in the charts from Figures 4.7 (namely: Star using mid-level representation, BIC, max pooling, QCCH, sum pooling and 45000 superpixels; Vargas using low-level representation, BIC, sum pooling and 37500 superpixels; No-ctxt using low-level representation, BIC and 37500 superpixels) and 4.8 (namely: Star using low-level representation, BIC, max pooling, QCCH, sum pooling and 45000 superpixels; Vargas using low-level representation, BIC, sum pooling and 30000 superpixels; No-ctxt using low-level representation, BIC and 30000 superpixels) were calculated for each class individually and reported on Tables 4.13 and 4.14, respectively.

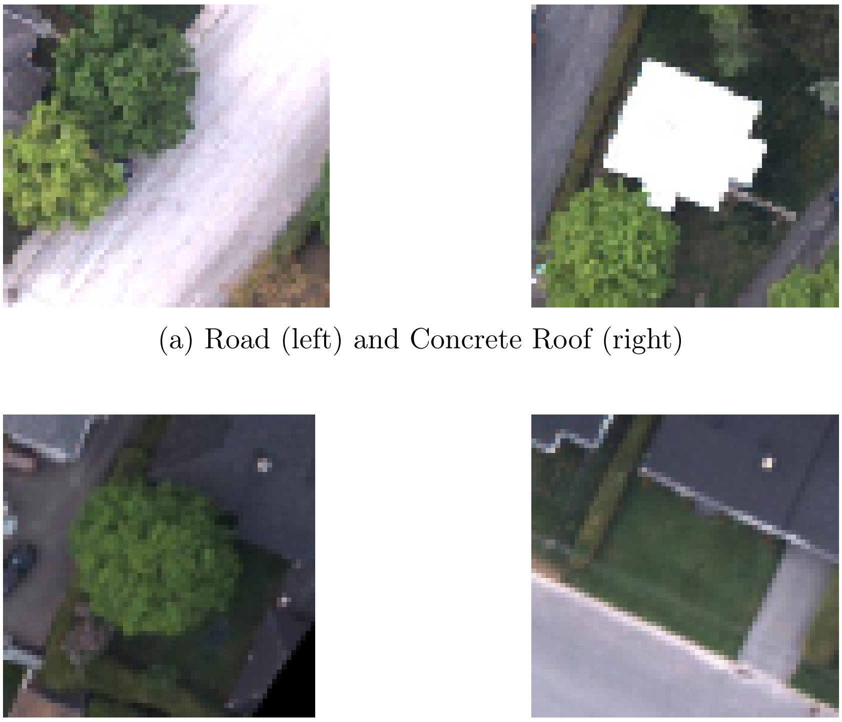
Table 4.13: Per class analysis in terms of Kappa of best results using SVM.

Method	Overall Kappa	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
		Road	Trees	Red Roof	Grey Roof	Concrete Roof	Veget.	Bare Soil
Star	0,729	0,738	0,837	0,817	0,573	0,668	0,758	0,819
Vargas	0,702	0,717	0,724	0,888	0,727	0,483	0,596	0,927
No-ctxt	0,631	0,677	0,734	0,729	0,561	0,472	0,571	0,693

Table 4.14: Per class analysis in terms of Kappa of best results using XGBoost.

Method	Overall Kappa	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
		Road	Trees	Red Roof	Grey Roof	Concrete Roof	Veget.	Bare Soil
Star	0,751	0,801	0,826	0,809	0,649	0,662	0,657	0,795
Vargas	0,672	0,718	0,755	0,856	0,644	0,437	0,490	0,846
No-ctxt	0,660	0,713	0,741	0,777	0,539	0,515	0,590	0,731

Looking at the results presented on these tables, it is easy to see that the usage of context improves classification results for most of the classes, except few specific situations: taking the results from the contextual methods using either SVM or XGBoost that achieved Kappa statistic worse than No-ctxt or improvements smaller than 1.00% (relative), we have the results of Vargas on classes Road, Trees, Concrete Roof and Vegetation. One possible reason for this decreasing in the results is that such classes may be hardly distinguished by color or even context. Trees and Vegetation often share similar color and the same context in urban scenes: some vegetation, other trees, a roof and a sidewalk. Classes like Road and Concrete Roof may be mistaken for each other due to a radiometric correction side effect that can be seen in Figure 4.9. In such situations, the edge descriptor of Star was helpful to confer robustness in the classification. Nevertheless, it is essential to highlight that the edge descriptor is not very beneficial for all classes: for Red Roof, Grey Roof and Bare Soil, which easily distinguished even by just color, Vargas achieved higher values for Kappa statistic, though Star also achieved results better than No-ctxt.



(a) Road (left) and Concrete Roof (right)

(b) Trees (left) and Vegetation (right)

Figure 4.9: Classes that may be mistaken for each other when using just color and context.

4.2.2 Results of the Experiments with the Visual Words Co-occurrence Matrix

The purpose of the initial experiments on VWCM is to find its best configuration with respect to the parameters that may be adjusted. Likewise the initial experiment of Star, the same global color and texture descriptors were selected from the work of dos Santos et al. [2010]: BIC, CCV, Unser and QCCH. In order to find out which one is the best descriptor to generate the visual words that will be counted, the other parameters were fixed: the grid sampling is performed using cells of 5×5 pixels and the codebook is composed of 16 visual words. Although using just 16 visual words may seem insufficient, it is important to remember that the final VWCM descriptor using a codebook of p visual words is going to be $p(p+1)/2$-dimensional, or 136-dimensional in this case. Moreover, the more the size of the codebook increases the more sparse the co-occurrences matrix is going to be as well as the VWCM descriptors, given that the area considered to count the co-occurrences remains the same. Additionally, a concatenation of VWCM with the BIC descriptor extracted from each superpixel was evaluated. The results using SVM with RBF kernel and XGBoost classifiers are reported in Table 4.15.

Table 4.15: Finding the best combination of descriptors for VWCM using cells of 5×5 pixels and codebooks of 16 visual words.

Classifier	Descriptors Used	Region Descriptor	Descriptor to Count Co-occurrences	Ovr. Acc.	Avg. Acc.	Kappa
SVM-RBF	Both	BIC	BIC	68,58%	91,02%	0,568
SVM-RBF	Both	BIC	CCV	67,44%	90,70%	0,553
SVM-RBF	Both	BIC	Unser	67,31%	90,66%	0,547
SVM-RBF	Both	BIC	QCCH	46,66%	84,76%	0,264
SVM-RBF	Co-occur.	-	BIC	68,58%	91,02%	0,568
SVM-RBF	Co-occur.	-	CCV	66,96%	90,56%	0,547
SVM-RBF	Co-occur.	-	Unser	64,07%	89,73%	0,504
SVM-RBF	Co-occur.	-	QCCH	46,48%	84,71%	0,262
XGBoost	Both	BIC	BIC	75,31%	92,94%	0,652
XGBoost	Both	BIC	CCV	74,36%	92,67%	0,640
XGBoost	Both	BIC	Unser	78,77%	93,93%	0,696
XGBoost	Both	BIC	QCCH	78,89%	93,97%	0,701
XGBoost	Co-occur.	-	BIC	70,73%	91,64%	0,594
XGBoost	Co-occur.	-	CCV	66,78%	90,51%	0,540
XGBoost	Co-occur.	-	Unser	61,55%	89,01%	0,478
XGBoost	Co-occur.	-	QCCH	46,74%	84,78%	0,278

The first point to highlight from Table 4.15 is the top result achieved for each classifier: VWCM using QCCH to compute the visual words and concatenated with BIC descriptor for XGBoost and VWCM using BIC to compose the codebook but without concatenating the region descriptor for the SVM classifier. Although concatenating the BIC descriptor computed from the superpixel itself has improved all other results except that of the own BIC also being used to compute visual words and using SVM as classifier, the version without concatenation was chosen as the best because it is 128-dimensions smaller and, therefore, faster to extract and also makes the model training faster.

Comparing the same settings classified by SVM with RBF kernel to those classified by XGBoost, only two of eight, i.e., 25% of them achieved higher values for the metrics using SVM. These results ratify the effectiveness of XGBoost to leverage features by selecting the best ones to grow trees.

Since the best combinations of descriptors were selected, it is essential to assess the behavior of the proposed method with relation to the other parameters. Thus, the two top settings from Table 4.15 were used to generate new land-cover maps, but now varying the size of the cells used for grid sampling. Each cell measures $q \times q$ pixels, and q assumed the values 3, 5, 7, and 9 for this experiment. The results can be seen in Figure 4.10.

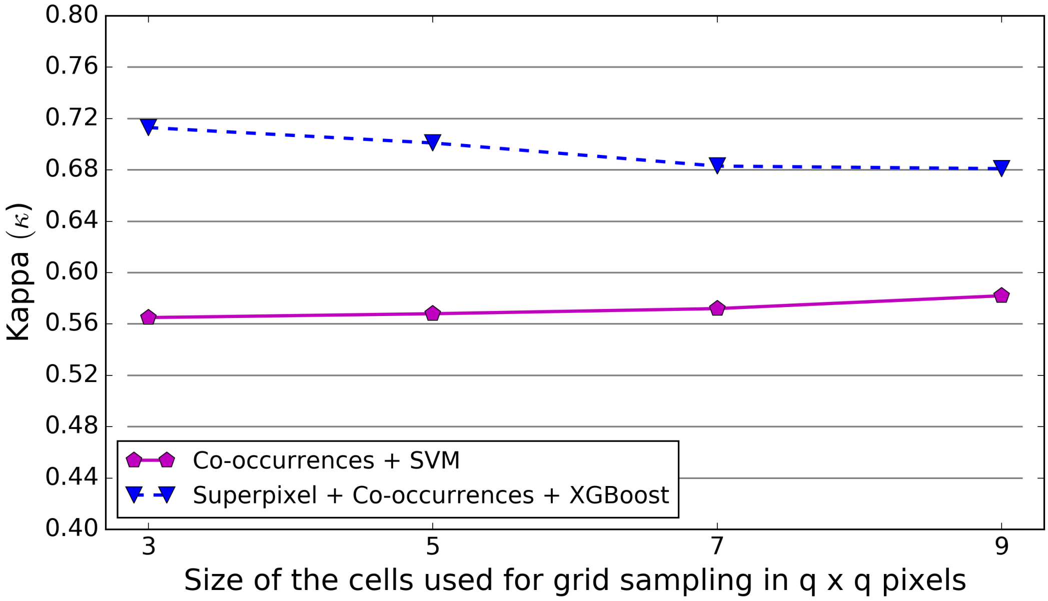


Figure 4.10: Looking for the best size of cells used for grid sampling.

One may observe from the chart in Figure 4.10 that the two best configurations have opposite behaviors: as the size of the cells of the grid increases, VWCM with BIC and SVM achieves a higher Kappa statistic while VCWM with QCCH, concatenated with BIC and using XGBoost has the same metric decreased. Of course, such reduction in effectiveness is not due to the BIC descriptor concatenated with VWCM, since it is not affected by the variation on the cell size. The decreasing is more likely explained by the nature of the low-level descriptor used. On one hand, QCCH is a histogram of changes computed for each pixel and its 8-neighbors. On the other hand, BIC and CCV are histograms computed taking into account each pixel and Unser is a set of measurements over two histograms (sum and difference) computed concerning each pair of pixels separated by a specific displacement. Therefore, for such a small area like a grid cell of at most 9×9 pixels which is likely to be homogeneous, there is a great probability that QCCH only adds another count in the same bin of the histogram for three main reasons: (1) the pixels are right next to each other and the area over which the change is computed is large (3×3 pixels) if compared to the area of the cell (at most 9×9 pixels), (2) there is intersection on the neighborhoods of adjacent pixels and (3) the 256 possible values for the change are quantized into just 40 bins. These reasons and the other results achieved by VWCM using QCCH shows that it is not a good choice for low-level descriptor to VWCM.

The next parameter to be evaluated is the number of visual words in the codebook. To assess the impact of varying this parameter, the best configurations (VWCM with QCCH using cells of 3×3 pixels and concatenated with BIC, and VWCM with BIC using cells of 9×9) were taken from the previous experiment and used to generate new maps using 8, 32 and 64 visual words, besides the initial 16. The results are reported in Figure 4.11.

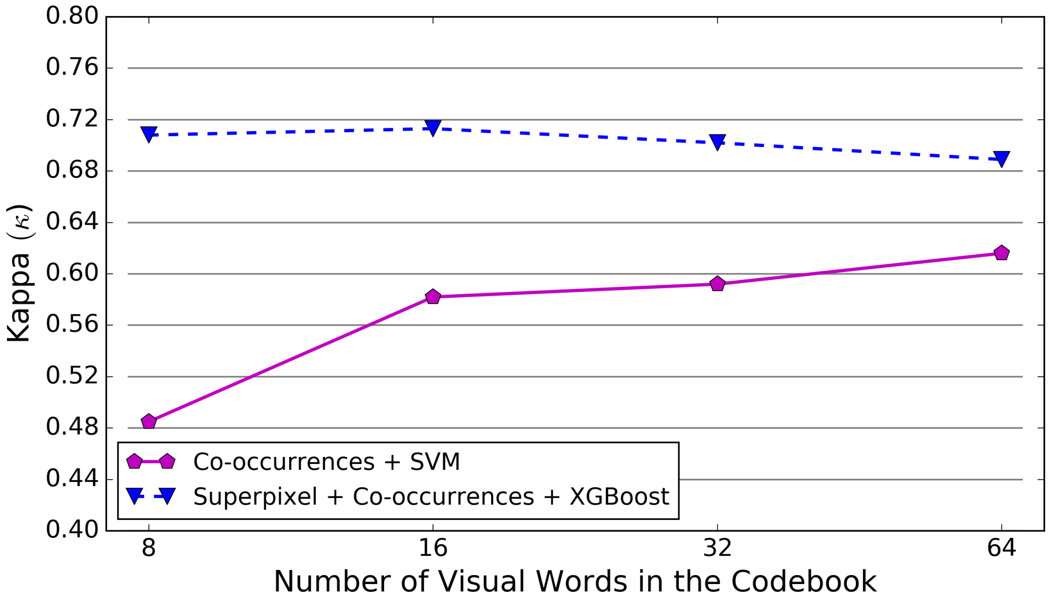


Figure 4.11: Impact of changing the number of visual words in the codebook of VWCM.

Similarly to the experiments using Star, it was expected that an intermediate number of visual words would yield the best value for the Kappa statistics. For VWCM computed from QCCH, concatenated with BIC and using XGBoost, this number was 16. For the other configuration it is a higher value. Nevertheless, once the dimensionality of the final descriptor grows quadratically on the number of visual words in the codebook and, consequently, the time to compute them and train the classifier becomes prohibitive, we did not used more than 64 visual words. Indeed using 128 visual words would result in descriptor of $128(129)/2 = 8256$ dimensions.

Another important aspect to evaluate is the impact of the number and, consequently, the size of superpixels on the quality of the maps generated at the end of the process. In order to carry out such evaluation, the best configurations found in the previous experiment (VWCM with QCCH using 16 visual words and concatenated with BIC and VWCM with BIC using 64 visual words) were used but now setting the number of superpixels in which the test image is segmented to 30000 and 45000, besides the initial 37500. The results are depicted on the chart in Figure 4.12.

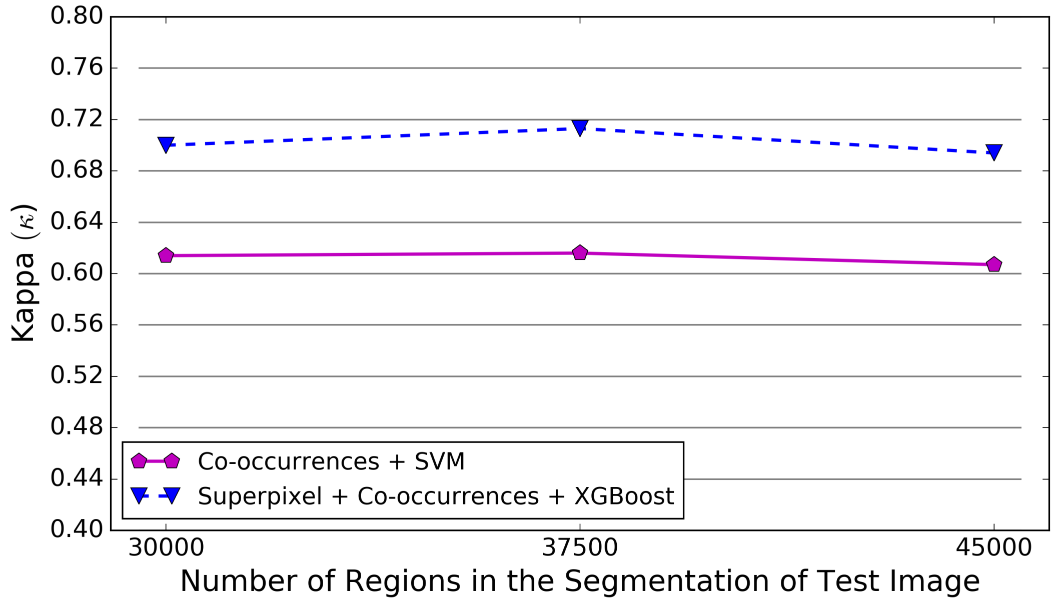


Figure 4.12: Impact of changing the number of superpixels used to segment the test image in the results of the best configurations of VWCM.

Similarly to the analysis done for the expected behavior of Star on varying the number of superpixels on Subsection 4.2.1, it was expected that VWCM achieves the top results using an intermediate value for this parameter. Indeed both configurations reached the highest value of Kappa statistics when the number of superpixels was set to 37500.

Once VWCM is based on GLCM, which has often been replaced since the 1970's by fourteen textural features (or a subset of them) computed from the matrix in order to make feature extraction and model training faster (widely known as Haralick features, proposed by Haralick et al. [1973]), an essential question raises: Could the same features be successfully applied to VWCM? To answer this question, a new experiment was performed replacing the vectorized representation of VWCM by thirteen of the fourteen Haralick features computed from the matrix of VWCM. The last feature (maximum correlation coefficient) was left out because it is often considered to be unstable. This comparison is presented in Table 4.16.

From those results it is possible to notice that using the Haralick features resulted in a large decrease in the results when combined with SVM. Although the 13 features

Table 4.16: Comparison of the vectorized representation to haralick features for the best settings of VWCM.

Classifier	Desc. Used	Region Desc.	Desc. to Count Co-occ.	Cell Size	Codebook Size	Use Haralick Feats.?	Ovr. Acc.	Avg. Acc.	Kappa
SVM	Co-occ.	-	BIC	9	64	No	72,38%	92,11%	0,616
SVM	Co-occ.	-	BIC	9	64	Yes	28,35%	79,53%	0,095
XGB	Both	BIC	QCCH	3	16	No	79,90%	94,26%	0,713
XGB	Both	BIC	QCCH	3	16	Yes	78,78%	93,94%	0,698

are not as discriminative as the entire matrix, XGBoost was able to compensate for this loss. Therefore, it is worth to replace the matrix representation when combined with a better classifier, once the training time is substantially reduced by using 13-dimensional feature vectors instead of 136 (when the codebook has only 16 codewords).

Finally, the results of the best settings of VWCM are reported class by class in Table 4.17 along with the baseline No-ctxt using BIC with low-level representation for comparison. Contrasting the results of VWCM with SVM to those of No-ctxt also with SVM, it is possible to find out that only for classes Vegetation and Bare Soil the first method achieved improvements over the baseline. On the other hand, the same comparison between VWCM concatenated with BIC and No-ctxt both using XGBoost, results on VWCM being much better for classes that are not easily distinguished only by color descriptors. Indeed, the relative improvement of Kappa statistic for the classes Trees, Concrete Roof and Vegetation was $21,24\%$, $28,04\%$ and $19,62\%$, respectively. Only class Road, which is very similar in terms of color to the class Concrete Roof, achieved a small improvement ($2,82\%$). Nevertheless, for the classes Red Roof and Grey Roof, there was a relative loss of $2,17\%$ and $0,46\%$. Taking into account that the BIC concatenated to VWCM is the same representation which is referred to as No-ctxt with BIC, one may conclude that using semantic context encoded by VWCM is beneficial when combined with a visual descriptor, but it is not enough to replace it.

4.2.3 Results of the Experiments with Many Context Levels Descriptor

The first experiment with MCL is an analysis of the contribution of the layers to the final result. To assess the importance of features from different layers, we trained the classifiers using all possible concatenations of the three layers of MCL. We also evaluated the same three layers (first, fifth and last fully connected) from the method proposed by Mostajabi et al. [2015] using AlexNet in order to have a baseline for

Table 4.17: Analysis of the best settings of VWCM detailed per class and reported in terms of Kappa.

Method	Overall Kappa	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
		Road	Trees	Red Roof	Grey Roof	Concrete Roof	Veget.	Bare Soil
VWCM + SVM	0,616	0,6570	0,7031	0,727	0,5058	0,4502	0,6094	0,7474
Visual + VWCM + XGB	0,713	0,7328	0,8989	0,7602	0,5367	0,659	0,7055	0,7844
No-ctxt Global BIC + SVM	0,631	0,677	0,734	0,729	0,561	0,472	0,571	0,693
No-ctxt Global BIC + XGB	0,660	0,713	0,741	0,777	0,539	0,515	0,590	0,731

comparison. The method is referred to as Zoom-out throughout this subsection. The results achieved are reported in Tables 4.18 and 4.19

Table 4.18: Layer importance analysis in grss_dfc_2014 dataset using SVM.

Layers	Method	Ovr. Acc.	Avg. Acc.	Kappa
Conv1	MCL	78,43%	93,84%	0,700
	Zoom-out 3 layers	78,43%	93,84%	0,700
Conv5	MCL	69,08%	91,17%	0,569
	Zoom-out 3 layers	77,33%	93,52%	0,674
Conv7	MCL	60,65%	88,76%	0,449
	Zoom-out 3 layers	60,65%	88,76%	0,449
Conv1 + Conv5	MCL	81,71%	94,77%	0,741
	Zoom-out 3 layers	84,29%	95,51%	0,775
Conv1 + Conv7	MCL	85,48%	95,85%	0,792
	Zoom-out 3 layers	85,48%	95,85%	0,792
Conv5 + Conv7	MCL	66,59%	90,45%	0,532
	Zoom-out 3 layers	79,19%	94,05%	0,698
Conv1 + Conv5 + Conv7	MCL	84,69%	95,63%	0,781
	Zoom-out 3 layers	85,96%	95,99%	0,798

Table 4.19: Layer importance analysis in grss_dfc_2014 dataset using XGBoost.

Layers	Method	Ovr. Acc.	Avg. Acc.	Kappa
Conv1	MCL	81,12%	94,61%	0,734
	Zoom-out 3 layers	81,12%	94,61%	0,734
Conv5	MCL	67,73%	90,78%	0,546
	Zoom-out 3 layers	74,10%	92,60%	0,639
Conv7	MCL	62,11%	89,18%	0,460
	Zoom-out 3 layers	62,11%	89,18%	0,460
Conv1 + Conv5	MCL	84,88%	95,68%	0,783
	Zoom-out 3 layers	86,42%	96,12%	0,803
Conv1 + Conv7	MCL	84,94%	95,70%	0,784
	Zoom-out 3 layers	84,94%	95,70%	0,784
Conv5 + Conv7	MCL	71,44%	91,84%	0,591
	Zoom-out 3 layers	79,06%	94,02%	0,694
Conv1 + Conv5 + Conv7	MCL	85,78%	95,94%	0,794
	Zoom-out 3 layers	85,77%	95,93%	0,795

When the layers are evaluated individually, the same behavior is observed for both classifiers and methods: the first layer has the greatest impact on the results and, as the image patches are forwarded across the network, the features generated become less discriminative. This is the opposite to what was expected, once the abstraction or semantic level increases from initial to final layers. Actually, using just the first layer we would be able to generate a good map, whose Kappa statistic may be regarded as substantial.

It is worth to notice that the results for combinations that do not include the layer Conv5 (the fifth one) are equal for both methods, once the Conv1 is the layer responsible for extracting features from the superpixel itself and Conv7 from the entire image patch and, thus, are the same for MCL and Zoom-out.

Observing the combinations using two layers, it is possible to notice a tendency that the composition of the first and last layers is the best one, followed by Conv1 with Conv5 and then by Conv5 with Conv7. Such behavior is expected, once Conv1 and Conv7 are complementary in terms of abstraction level (low and high, respectively) and contextual area (local and global, respectively).

With respect to the top values for Kappa statistics, there is no clear observable pattern: MCL performed better using just Conv1 and Conv7 with SVM and using all three layers with XGBoost while Zoom-out achieved the best results using all three layers with SVM and using Conv1 and Conv5 with XGBoost. It is also important to mention that Zoom-out using Conv1 and Conv5 with XGBoost is the only setting that is considered almost perfect according to the reference from Table 4.8.

The next aspect to assess is the impact of the segmentation on the maps generated. In order to do so, both methods using the three layers and additionally Zoom-out using all layers (what resulted in feature vectors of $96+256+384+384+256+4096+4096 = 9568$ dimensions) were used to create new maps after setting the number of superpixels parameter of SLICO to 30000 and 45000 besides the initial 37500 ones for the test image. The results are presented in Figures 4.13 and 4.14.

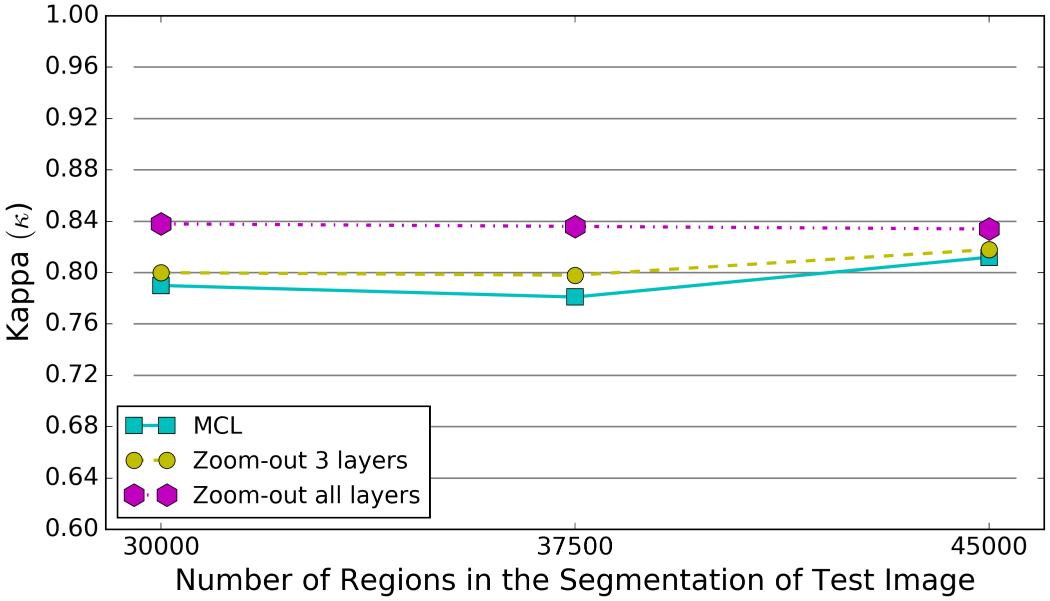


Figure 4.13: Impact of changing the number of regions used to segment the test image in the results of MCL and baselines with SVM.

Likewise Star and VWCM, the Kappa statistics is expected to reach the top value after increasing the number of regions and then starts to drop. Observing the results, one can easily see that only Zoom-out using all layers with SVM decreased the results when the number of superpixels was increased. All other settings achieved the top results using 45000 superpixels. Even though the optimal value for the parameter might be larger, no values beyond 45000 were tried due to efficiency issues. It is worth to highlight that using this number of superpixels, MCL and Zoom-out with 3 layers achieved 0.812 and 0.818 with SVM, respectively, and Zoom-out with all layers and XGBoost has reached 0.841 of Kappa statistics. All these values are regarded as almost perfect according to Table 4.8.

After assessing all these aspects of MCL, it is important to detail the best results by class in order to evaluate how the method behaves for different materials covering the Earth surface. Tables 4.20 and 4.21 show the results of MCL, Zoom-out (using three

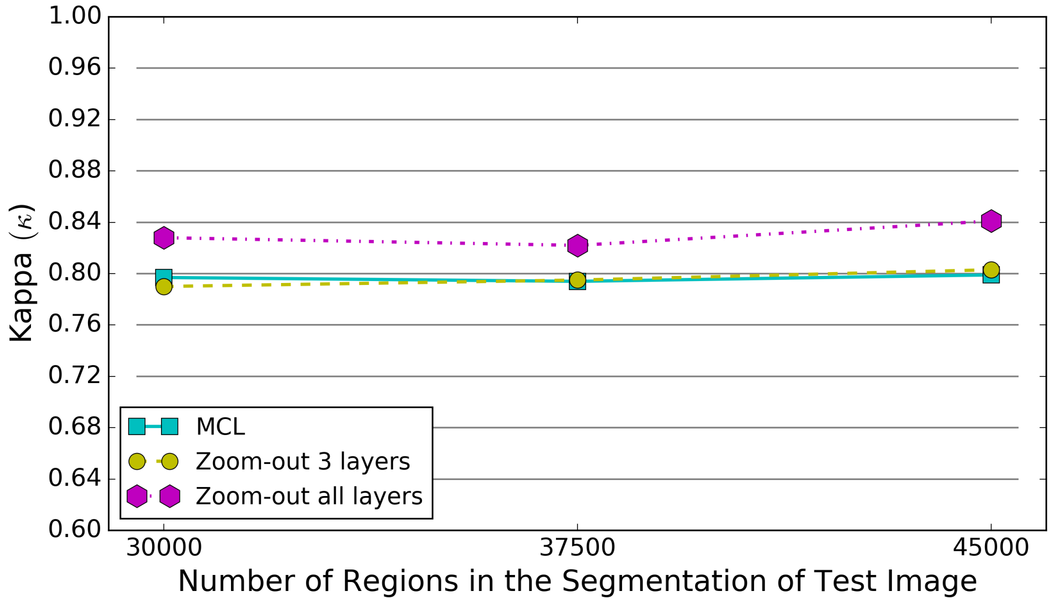


Figure 4.14: Impact of changing the number of regions used to segment the test image in the results of MCL and baselines with XGBoost.

and all layers) and the baseline No-ctxt for both SVM and XGBoost after employing SLICO with the parameter related to the number of superpixels set to 45000, which was in general the best configuration.

Table 4.20: Analysis of the results of MCL with SVM detailed per class and reported in terms of Kappa.

Method	Overall Kappa	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
		Road	Trees	Red Roof	Grey Roof	Concrete Roof	Veget.	Bare Soil
MCL	0,812	0,816	0,864	0,898	0,767	0,688	0,837	0,876
Zoom-out 3 layers	0,818	0,818	0,879	0,918	0,786	0,655	0,864	0,886
Zoom-out all layers	0,834	0,832	0,949	0,937	0,808	0,644	0,890	0,862
No-ctxt Global BIC + SVM	0,631	0,677	0,734	0,729	0,561	0,472	0,571	0,693

Comparing the results of MCL and Zoom-out to those of the baseline, one can see that the most benefited classes were Grey Roof, Concrete Roof and Vegetation, where

Table 4.21: Analysis of the results of MCL with XGBoost detailed per class and reported in terms of Kappa.

Method	Overall Kappa	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
		Road	Trees	Red Roof	Grey Roof	Concrete Roof	Veget.	Bare Soil
MCL	0,799	0,822	0,860	0,917	0,731	0,653	0,787	0,835
Zoom-out 3 layers	0,803	0,819	0,868	0,925	0,739	0,649	0,816	0,837
Zoom-out all layers	0,841	0,850	0,894	0,950	0,769	0,700	0,869	0,914
No-ctxt Global BIC + XGB	0,660	0,713	0,741	0,777	0,539	0,515	0,590	0,731

the two last classes has colors quite similar to Trees and Road classes, respectively. The relative improvement of MCL over the baseline using SVM was of $36,64\%$, $45,83\%$ and $46,67\%$ for the three classes, respectively, while Zoom-out with just three layers achieved $40,03\%$, $38,84\%$ and $51,40\%$, respectively. On the other hand, when using XGBoost the increasing on the values of Kappa statistics was much smaller because XGBoost is capable of selecting the best features and thus leverage feature vectors that are less discriminative or even containing redundancy or noise, like those of No-ctxt. Once MCL and Zoom-out are features learned from data, which are usually much more discriminative, the work of XGBoost on selecting the best ones is softened. This becomes clear when one compares the results of MCL, Zoom-out and No-ctxt using XGBoost to those using SVM: No-ctxt performed much better with the former classifier while MCL and Zoom-out using three layers achieved top results with SVM, indicating that No-ctxt was benefited by the feature selection of XGBoost. In fact, the relative improvements of MCL over the baseline with XGBoost for the same three classes were $35,57\%$, $26,87\%$ and $33,44\%$, respectively, and Zoom-out using three layers achieved $37,05\%$, $26,10\%$ and $38,36\%$, respectively.

4.2.4 Comparison of the Proposed Methods

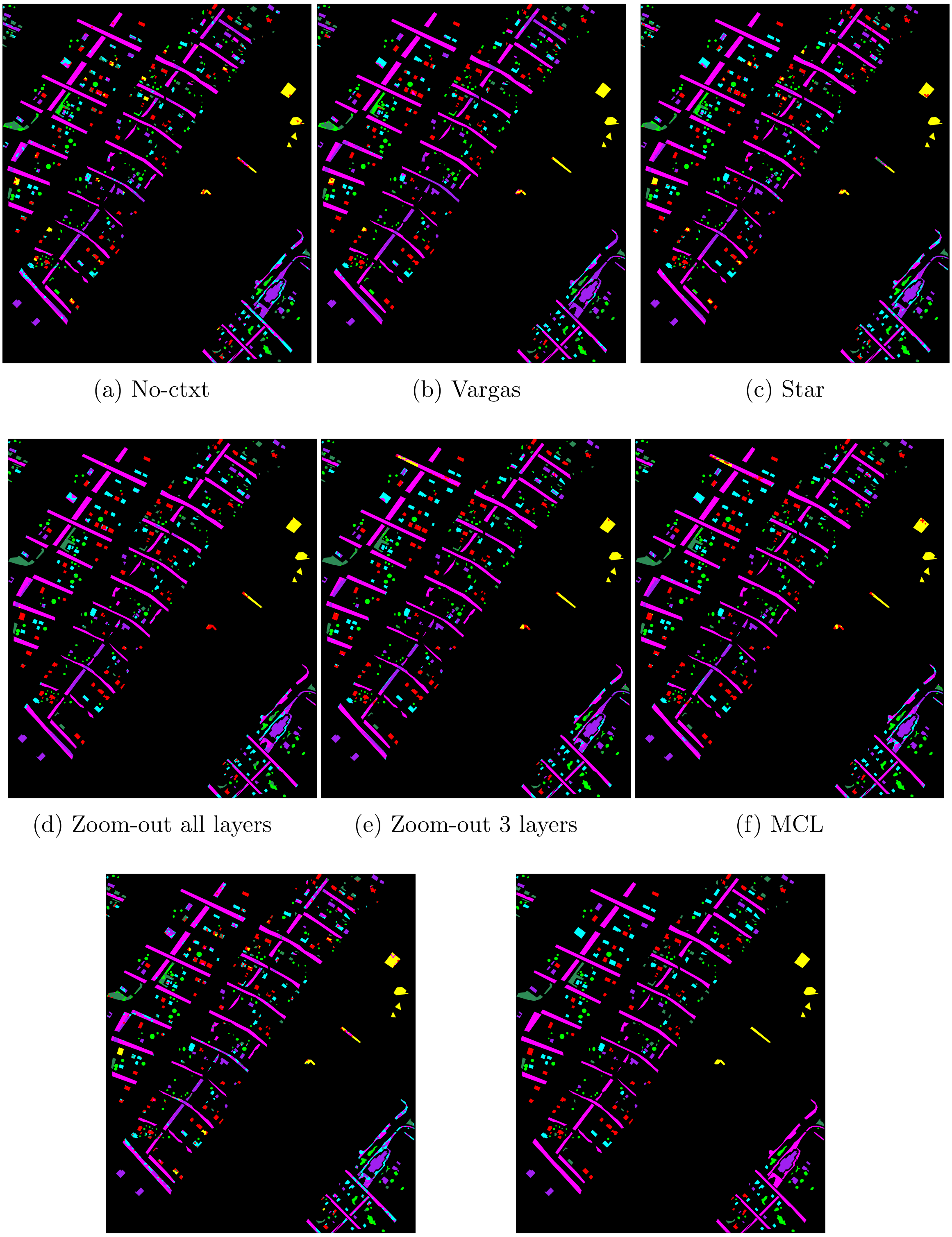
To sum up the experiments carried out so far, this subsection is dedicated to compare the proposed methods and the baselines. Table 4.22 presents the best results of each proposed method and the baselines and Figure 4.15 shows the respective generated maps.

Table 4.22: Comparison among the best results of each proposed method and baselines.

Method	Number of Superpixels	Classifier	Ovr. Acc.	Avg. Acc.	Kappa
No-ctxt Global BIC	30000	XGB	75,97%	93,13%	0,660
Vargas Global BIC Sum pooling	37500	SVM	78,83%	93,95%	0,702
Star Global BIC Max pooling QCCH Sum pooling	45000	XGB	82,86%	95,10%	0,751
Region + VWCM QCCH 3x3 cells 16 codewords	37500	XGB	79,90%	94,26%	0,713
Zoom-out 3 layers	45000	SVM	87,44%	96,41%	0,818
Zoom-out all layers	45000	XGB	89,17%	96,91%	0,841
MCL	45000	SVM	86,99%	96,28%	0,812

A first thing to notice is that average accuracy is always greater than overall accuracy, not only in Table 4.22 but also in all the experiments of this section. This is due to the low effectiveness of the methods on classes with large number of samples, mainly the class Roads which represents $19,79\%$ and $55,73\%$ of the training and test samples, respectively. In spite of the fact that all methods achieved reasonable results when taking just this class into account (indeed the average Kappa statistic of the results presented in Tables 4.13, 4.14, 4.17, 4.20 and 4.21 is 0,765), when even just about 10% of the samples of this class are misclassified it causes a huge impact on the overall accuracy, once $55,73\%$ of the training samples belong to this class.

Another important aspect is that all the top results of the methods using context are better than the baseline without it, confirming that context really improves classification. Moreover, one can see from Tables 4.22 that data-driven features were much more effective than the handcrafted ones, even when extracted using a pre-trained



(a) No-ctxt (b) Vargas (c) Star

(d) Zoom-out all layers (e) Zoom-out 3 layers (f) MCL

(g) VWCM (h) Ground Truth

Figure 4.15: Maps generated using the best configurations of each method.

ConvNet without fine-tuning: Kappa index of all methods that use ConvNets is regarded as almost perfect according to the reference on Table 4.8 while the handcrafted contextual descriptors is only substantial.

After extensively assessing the proposed methods, two questions are still to be answered:

1. Were the good results due to the discriminative features or a combination of them with better classifiers? and

2. How much does the segmentation limit the results? Or equivalently, how much error does SLICO introduce to the classification process?

In order to answer the first question, we carried out a experiment using the configurations of each method listed in Table 4.22 with a lazy classifier: *k-Nearest Neighbors* (kNN). To determine the best number of neighbors for kNN (the best k), a 5-fold Cross-validation was performed varying k from 1 to 15 in steps of 2. Once kNN does not require any training, instead assigns a class to a sample according to its distance to the others, it is possible to compare which method generates representations that result in the best class separability, i.e., samples belonging to the same class are close to each other and far from samples of other classes. The results of this experiment are reported in Table 4.23.

Observing Table 4.23, the first thing that becomes clear is that QCCH is not a good descriptor alone. If one compares the values of Kappa statistics achieved by the methods with kNN to those with other classifiers, the specific configurations of Star and VWCM used in this experiment had a relative decreasing of $41,34\%$ and $66,26\%$, respectively. Once the descriptor proposed by Vargas et al. [2015] is also composed of a vertex descriptor using BIC and did not suffer such large loss, one may infer that these results ratify the conclusions of the experiment of VWCM in Subsection 4.2.2 that showed that QCCH is not a good descriptor for small areas of the image. Therefore, the reasonable results for Star and VWCM reported in Table 4.22 may be attributed to the combination with BIC to describe the superpixel itself and XGBoost as classifier, which is able to leverage feature vectors containing noise and redundancy. Nevertheless, these results do not mean that all setting of Star and VWCM are so sensitive to the classifier chosen. MCL is also worse if compared to the other data-driven representations, having a relative decreasing of $8,35\%$, while Zoom-out using three and all layers had only $7,99\%$ and $5,73\%$ of loss, respectively.

Another result essential to highlight was achieved by the baseline Vargas: using kNN caused a reduction of only $1,57\%$ in its effectiveness (in terms of Kappa index), being more robust to the choice of classifier than all data-driven representations. Despite the method had not achieved greater values of Kappa statistics, it is a good

Table 4.23: Comparison among the best results of each proposed method and baselines using kNN classifier.

Method	Number of Superpixels	Classifier	Ovr. Acc.	Avg. Acc.	Kappa
No-ctxt Global BIC	30000	kNN	71,22%	91,78%	0,599
Vargas Global BIC Sum pooling	37500	kNN	78,05%	93,73%	0,691
Star Global BIC Max pooling QCCH Sum pooling	45000	kNN	59,20%	88,34%	0,441
Region + VWCM QCCH 3x3 cells 16 codewords	37500	kNN	42,49%	83,57%	0,241
Zoom-out 3 layers	45000	kNN	82,74%	95,07%	0,753
Zoom-out all layers	45000	kNN	85,81%	95,95%	0,793
MCL	45000	kNN	82,26%	94,93%	0,744

choice once it generates a very discriminative representation that is able to generate good land-cover maps even using weak classifiers. No-ctxt was reasonable, having a drop of only $9,21\%$.

The second question is of great relevance due to the difference between the classification and evaluation processes: the feature vectors used to train the classifier and to which a label is assigned to are computed from superpixels, and the map is built by painting all the pixels inside each superpixel with the color corresponding to the class assigned by the classifier to its respective feature vector. On the other hand, all the metrics used to assess the effectiveness of the proposed method (described in Subsection 4.1.5) are pixel-wisely computed. If the segmentation were perfect and therefore there were pixels from just one class inside each superpixel, there would be no problem at assigning only one predicted class to the superpixel and at the same time using its pixels to compute the metrics. In such situation, all the pixels inside a superpixel would belong to the same actual class and, thus, all of them would count either only as hits or misses. But as the segmentation generated by SLICO is not perfect, when a predicted class is assigned to all the pixels of a superpixel, some of them are correctly classified while other are misclassified just because they were included during the segmentation process inside a superpixel they should not belong to. Thereby, it is essential to measure this amount of error introduced by the segmentation.

Aiming at answering this question, we simulated a perfect classification taking the ground truth annotation of the test image as if it were a prediction of the classes used to build the map. Once the ground truth annotation is the reference to which the constructed map is compared to, there is not a better prediction than the ground truth. Therefore, the process to simulate the perfect classification begins by superimposing the segmentation generated by SLICO to the image corresponding to the ground truth annotation of the test image. Secondly, a majority vote is performed in order to decide which class is going to be assigned to each superpixel: each pixel votes for its respective actual class according to the ground truth and black pixels are ignored. By assigning to each superpixel the most voted class taking into account only the pixels within it, we are introducing the smallest possible error due to the segmentation, once the number of hits by counting the pixels correctly classified is maximized. Then a map is built in the same manner as the classification process: using the predictions made by the majority vote, the color corresponding to the class assigned to a specific superpixel is used to paint all pixels inside it. Finally, all metrics described in Subsection 4.1.5 are calculated comparing the generated map to the ground truth annotations and the results are presented in Table 4.24. We refer to the values achieved for each metric in a given segmentation scale as theoretical maximum values, once they are the maximum values that one might achieve for such metrics using the given segmentation in a perfect classification.

Table 4.24: Theoretical maximum values for each metric on grss_dfc_2014 dataset.

Number of Superpixels	Ovr. Acc.	Avg. Acc.	Kappa
30000	99,9795%	99,9942%	0.999687
37500	99,9848%	99,9957%	0.999768
45000	99,9879%	99,9966%	0.999816

Of course, there is no single segmentation scale that is capable of delineating all meaningful object on the RSI, once superpixel methods are prone to generate regions more homogeneous in shape and size than traditional segmentation methods and the objects within the image are in a wide range of sizes. This means that some small objects will be within superpixels that cover other objects or part of them, some will fit almost perfectly inside an entire superpixel and others will be oversegmented. Nevertheless, this is not only a problem of segmentation, but also of spatial resolution of the RSI: it is widely known in remote sensing that it is not suitable to use a sensor whose resolution is relatively coarse to distinguish smaller objects. Therefore, each RSI with a different resolution is better to address a classification problem whose set of classes contain objects that are not smaller than a specific size. In this sense, the choice for SLICO and the segmentation scales applied were successful, once they allowed for an almost perfect (as shown in Table 4.24) map generation that the features and classifiers used were not able to achieve.

4.3 Results on ISPRS Potsdam Dataset

Once the ISPRS Potsdam dataset has much more training data than the grss_dfc_2014 one (about 120000 against nearly 3000, respectively), it is not feasible to carry out many experiments to find out the best configurations of each method that fit better on the data. Thereby, only a few settings of each method which achieved top results were used. For this experiment, the XGBoost classifier was trained without grid search for MCL and Zoom-out with 3 layers once the high dimensionality of the feature vectors generated (4448 dimensions) combined with the large number of hyper-parameters of XGBoost to optimize would make this experiment infeasible. The results achieved so far are presented in Table 4.25.

Analyzing Table 4.25, it is possible to notice some differences on the behavior of the methods with relation to the grss_dfc_2014 dataset. For instance, the method MCL performed slightly better than Zoom-out with 3 layers on the ISPRS Potsdam dataset than on the previous one, even though the method is not significantly better from the statistical perspective. Such situation has not happened on the former dataset. Also important to mention is the fact that though the VWCM method achieved a Kappa statistic that can be regarded only as moderate according to the reference in Table 4.8, this dataset is much more challenging than the grss_dfc_2014 one due to the higher intra-class variance and the requirement for classifying the background class. Another aspect that must be highlighted is that No-ctxt achieved better results than Vargas and Star. Nevertheless, it is important to notice that not all top settings for Vargas, Star and VWCM were examined yet. Thereby, it is still possible that the results of Star and Vargas with XGBoost and VWCM using QCCH are better.

Table 4.25: Comparison of the proposed methods on ISPRS Potsdam dataset.

Method	Number of Superpixels	Classifier	Ovr. Acc.	Kappa
No-ctxt Global BIC	30000	XGB	65,65% ± 4,99%	0,496 ± 0,060
Vargas Global BIC Max pooling	30000	SVM Linear	50,10% ± 5,80%	0,275 ± 0,053
Star Global BIC Max pooling Unser Avg pooling	30000	SVM Linear	42,10% ± 6,40%	0,181 ± 0,049
Region + VWCM BIC 3x3 cells 16 codewords	30000	XGB	70,39% ± 4,61%	0,563 ± 0,062
Zoom-out 3 layers	30000	XGB	81,40% ± 3,2%	0,726 ± 0,042
MCL	30000	XGB	81,55% ± 3,1%	0,728 ± 0,041

4.4 Time Complexity Analysis

In order to compare the proposed methods, it is of great relevance comparing their computational efficiency. To do so, we have analyzed the time complexity of each of the three methods for extracting contextual features from RSIs. The time complexity function and order may provide an efficiency estimate that is independent of the implementation due to the abstraction of the underlying machine. Such characteristics are suitable for a theoretical analysis that focus on the algorithm itself, providing a fair comparison once many methods are leveraged by implementation tricks.

Once all the proposed methods are region-based, it is interesting to analyze how each of them behaves as the number of superpixels increases. Thereby, the time complexity is in function of the number of superpixels N.

Thinking of how Star works, one may divide its analysis in three steps: feature extraction, vertex descriptor composition and edge descriptor composition. Observing the first step, it is important to notice that the method is independent of the underlying low or mid-level representation chosen. Therefore, we are going to abstract how this underlying descriptor works and begin the analysis assuming that a feature vector is already assigned to each of the N superpixels, taking a constant time c_1 for each of them. In order to model the time complexity for the vertex composition step, two aspects that we have to highlight are that it depends on the number of neighbors of each superpixel and depends on how the composition operation works. Considering that the N superpixels have an average number of neighbors V, and the vertex composition operation takes a constant time c_2 to combine two feature vectors, we have that the complexity for this step is $c_2 * (V - 1) * N = c_2 * (VN - N)$. The same assumptions are true for the edge descriptor composition step: assuming a constant time c_3 for the edge composition operation, the complexity of this step is $c_3 * (VN - N)$. Bringing together the three steps, the time complexity function for the Star descriptor may be defined as:

$$\begin{aligned} f(N) &= c_1 N + c_2(VN - N) + c_3(VN - N) \\ &= c_1 N + c_2 VN - c_2 N + c_3 VN - c_3 N \\ &= (c_2 + c_3)VN + (c_1 - c_2 - c_3)N \end{aligned} \tag{4.14}$$

Omitting the constants, we may say that the time complexity order for the Star descriptor is $O(V * N)$. Nevertheless, in practice we have that $V << N$ (V is usually a number from 6 to 8 while N is greater than 20000) and the constants (mainly c_1) are not so insignificant.

Likewise, the analysis of the VWCM method may be separated into two steps: assignment and co-occurrences counting. Once the codebook composition is an offline process, it is not going to be considered in the time complexity. Thus, we are going to begin the analysis of the assignment step assuming that we already have a codebook of K visual words. The assignment step depends on the number of visual words inside each superpixel. Denoting the average number of codewords within the superpixels as C, we have that the complexity of this step is $K * C * N$ because all codewords must be tested in order to choose which one is going to be assigned to a superpixel. Thinking of the second step, one may realize that the analysis may be further divided into two parts: internal and within neighborhood co-occurrences counting. Once the internal co-occurrence relationship is symmetric, the number of other codewords to which each visual word inside a superpixel is compared follows an arithmetic progression with a common difference of -1 and the first term being $C - 1$. Thus, the complexity may be defined as $(C^2 - C)N/2$. On the other hand, the relationship of co-occurrence between neighbors is not symmetric and therefore the complexity of this part becomes $C^2 * V * N$ because every codeword of a superpixel is compared to all visual words inside each of the V neighbors. Joining the two steps, the complexity of VWCM may be defined as:

$$f(N) = KCN + \frac{(C^2 - C)N}{2} + C^2VN \tag{4.15}$$

In terms of complexity order, one may say that VWCM is $O(C^2 * V * N)$. However, it is important to mention that the number of visual words in the codebook has a strong influence on the overall efficiency of the method.

Once the ConvNet training is an offline process, we are going to left it out of the analysis, assuming that we already have a trained network. The efficiency of MCL basically depends on the ConvNet architecture chosen and the type of interpolation used for upsampling the patches. It is essential to notice that the number of operations performed is fixed for a given architecture, i.e., changes from one architecture to another but is the same for all patches forwarded in the same ConvNet. Therefore, we may abstract all the operations performed by a given ConvNet (including the upsampling) and denote it by the constant c_4. Thereby, the complexity function of the MCL method may be defined as:

$$f(N) = c_4N \tag{4.16}$$

Besides having the best time complexity order ($O(N)$), MCL benefits from the practical aspect of being trained on General Purpose Graphical Processing Units (GPGPUs). Considering its efficiency and accuracy, one may conclude that MCL is the best option among the methods evaluated in this work from both the practical and theoretical viewpoint.

Chapter 5

Conclusions and Future Work

This work has dealt with the problem of creating discriminative representations for superpixels of RSIs by encoding not only the visual appearance of objects, but also their context. In order to address the problem, three new methods to encode context were proposed and extensively evaluated: Star, which makes use of a RAG to represent the spatial relationship between a superpixel and its neighbors and then combines feature vectors computed from vertices and edges into just one representation that encodes both the visual appearance and the context of the superpixel; VWCM, which was inspired by the traditional GLCM, encodes the semantic context from the local neighborhood of a superpixel by counting co-occurrences of codewords only inside it and co-occurrences between the superpixel and each of its neighbors; and the last proposed method (MCL) exploits ConvNets 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.

The experiments revealed some interesting points. One of them is that low-level representations achieved better results than using BoVW with Star, what is totally opposite to the expected behavior. Another important aspect is that the top results were reached using as few visual words as 128 for Star and 16 for VWCM. Concerning the MCL, it was possible to notice that data-driven features performed better than the handcrafted ones, though the difference was not large. Taking Table 4.8 as a reference, the top results for all the proposed methods would be classified as substantial. A key finding with respect to the chosen superpixels method was that SLICO and the parameters used allowed for an almost perfect segmentation, introducing very small error.

Currently, more experiments are in progress on the ISPRS Potsdam dataset in order to assess the behavior of the proposed methods when mapping a larger area using images with higher resolution than the previous dataset, what increases the intra-class variance due to the heterogeneity in the appearance of objects. We also plan to evaluate MCL using other ConvNets, like VGG, as an end-to-end solution instead of employing other classifiers to generate the maps.

Bibliography

Abe, S. (2003). On invariance of support vector machines. In *Proceedings of the 4th international conference on intelligent data engineering and automated learning*.

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and S'usstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(11):2274–2282. ISSN 0162-8828.

Alpaydin, E. (2014). *Introduction to Machine Learning*. The MIT Press. ISBN 0262028182, 9780262028189.

Bar, M. (2004). Visual objects in context. *Nature reviews. Neuroscience*, 5(8):617.

Binaghi, E., Madella, P., Montesano, M. G., and Rampini, A. (1997). Fuzzy contextual classification of multisource remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 35(2):326–340. ISSN 0196-2892.

Blaschke, T. (2010). Object based image analysis for remote sensing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(1):2 – 16. ISSN 0924-2716.

Blaschke, T., Hay, G. J., Kelly, M., Lang, S., Hofmann, P., Addink, E., Feitosa, R. Q., van der Meer, F., van der Werff, H., van Coillie, F., and Tiede, D. (2014). Geographic object-based image analysis - towards a new paradigm. *ISPRS Journal of Photogrammetry and Remote Sensing*, 87:180 – 191. ISSN 0924-2716.

Blaschke, T., Lang, S., Lorup, E., Strobl, J., and Zeil, P. (2000). Object-oriented image processing in an integrated gis/remote sensing environment and perspectives for environmental applications. *Environmental information for planning, politics and the public*, 2:555--570.

Blaschke, T. and Strobl, J. (2001). What's wrong with pixels? some recent developments interfacing remote sensing and gis. *GIS–Zeitschrift f'ur Geoinformationssysteme*, 14(6):12–17.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees.* CRC press.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754.

Choi, M. J., Lim, J. J., Torralba, A., and Willsky, A. S. (2010). Exploiting hierarchical context on a large database of object categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273--297. ISSN 0885-6125.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21--27.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '05, pages 886--893, Washington, DC, USA. IEEE Computer Society.

de Andrade, E. F., Araújo, A. d. A., and dos Santos, J. A. (2015). A multiclass approach for land-cover mapping by using multiple data sensors. In *Iberoamerican Congress on Pattern Recognition (CIARP)*, pages 59--66. Springer.

dos Santos, J., Penatti, O., Torres, R. d. S., Gosselin, P.-H., Philipp-Foliguet, S., and Falcão, A. (2012). Improving texture description in remote sensing image multi-scale classification tasks by using visual words. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*, pages 3090--3093.

dos Santos, J. A., Penatti, O. A. B., and Torres, R. d. S. (2010). Evaluating the potential of texture and color descriptors for remote sensing image retrieval and classification. In *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*, Angers, France.

Duarte, F. (2009). *Planejamento urbano.* Ed. Ibpex. ISBN 9788599583418.

Fauvel, M., Chanussot, J., and Benediktsson, J. A. (2012). A spatial-spectral kernel-based approach for the classification of remote-sensing images. *Pattern Recognition*, 45(1):381 – 392. ISSN 0031-3203.

Fischler, M. A. and Elschlager, R. (1973). The representation and matching of pictorial structures. *IEEE Transactions on Computers*, C-22(1):67–92. ISSN 0018-9340.

Fisher, P. (1997). The pixel: A snare and a delusion. *International Journal of Remote Sensing*, 18(3):679–685.

Galleguillos, C. and Belongie, S. (2010). Context based object categorization: A critical survey. *Computer Vision and Image Understanding (CVIU)*, 114(6):712--722. ISSN 1077-3142.

Galleguillos, C., Rabinovich, A., and Belongie, S. (2008). Object categorization using co-occurrence, location and appearance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. ISSN 1063-6919.

Gonzalez, R. C. and Woods, R. E. (2006). *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA. ISBN 013168728X.

Hanson, A. R. and Riseman, E. M. (1978). VISIONS: A computer system for interpreting scenes. In Hanson, A. R. and Riseman, E. M., editors, *Computer Vision Systems*. Academic Press, New York.

Haralick, R. M., Shanmugam, K., and Dinstein, I. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621. ISSN 0018-9472.

Hay, G. J. and Castilla, G. (2008). *Geographic Object-Based Image Analysis (GEOBIA): A new name for a new discipline*, pages 75--89. Springer Berlin Heidelberg, Berlin, Heidelberg.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554--2558.

Hu, T., Yang, J., Li, X., and Gong, P. (2016). Mapping urban land use by using landsat images and open social data. *Remote Sensing*, 8(151). ISSN 2072-4292.

Huang, C. B. and Liu, Q. (2007). An orientation independent texture descriptor for image retrieval. In *2007 International Conference on Communications, Circuits and Systems*, pages 772–776.

Intergovernmental Committee on Surveying and Mapping (2017). Fundamentals of mapping. `http://www.icsm.gov.au/mapping/maps_thematic.html`. Accessed: 2017-09-02.

Ippoliti, E., Clementini, E., and Natali, S. (2012). Automatic generation of land use maps from land cover maps. In *AGILE International Conference on Geographic Information Science*.

Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010). Aggregating local descriptors into a compact image representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3304–3311. ISSN 1063-6919.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097--1105. Curran Associates, Inc.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159--174. ISSN 0006-341X.

Lim, J. J., Arbelaez, P., ArbelÃ¡ez, P., Gu, C., and Malik, J. (2009). Context by region ancestry. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1978–1985. ISSN 1550-5499.

Marceau, D. J., Howarth, P. J., Dubois, J. M., and Gratton, D. J. (1990). Evaluation of the grey-level co-occurrence matrix method for land-cover classification using spot imagery. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 28(4):513–519. ISSN 0196-2892.

Melgani, F. and Serpico, S. B. (2002). A statistical approach to the fusion of spectral and spatio-temporal contextual information for the classification of remote-sensing images. *Pattern Recognition Letters*, 23(9):1053 – 1061. ISSN 0167-8655.

Meneses, P. R. and de Almeida, T. (2012). *Introdução ao processamento de imagens de sensoriamento remoto*. Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq, Brasília, DF.

Moser, G., Serpico, S. B., and Benediktsson, J. A. (2013). Land-cover mapping by markov modeling of spatial-contextual information in very-high-resolution remote sensing images. *Proceedings of the IEEE*, 101(3):631–651. ISSN 0018-9219.

Mostajabi, M., Yadollahpour, P., and Shakhnarovich, G. (2015). Feedforward semantic segmentation with zoom-out features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3376--3385.

Mottaghi, R., Chen, X., Liu, X., Cho, N.-G., Lee, S.-W., Fidler, S., Urtasun, R., and Yuille, A. (2014). The role of context for object detection and semantic segmentation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 891–898.

Mou, L., Zhu, X., Vakalopoulou, M., Karantzalos, K., Paragios, N., Saux, B. L., Moser, G., and Tuia, D. (2017). Multitemporal very high resolution from space: Outcome of the 2016 ieee grss data fusion contest. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(8):3435–3447. ISSN 1939-1404.

Palmer, S. E. (1975). The effects of contextual scenes on the identification of objects. *Memory & Cognition*, 3(5):519--526.

Parikh, D., Zitnick, C. L., and Chen, T. (2008). From appearance to context-based recognition: Dense labeling in small images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. ISSN 1063-6919.

Parikh, D., Zitnick, C. L., and Chen, T. (2012). Exploring tiny images: The roles of appearance and contextual information for machine and human object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(10):1978–1991.

Pass, G., Zabih, R., and Miller, J. (1996). Comparing images using color coherence vectors. In *Proceedings of the Fourth ACM International Conference on Multimedia*, MULTIMEDIA '96, pages 65--73, New York, NY, USA. ACM.

Peirce, J. W. (2015). Understanding mid-level representations in visual processing. *Journal of Vision*, 15(7):5--5.

Perronnin, F. and Dance, C. (2007). Fisher kernels on visual vocabularies for image categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. ISSN 1063-6919.

Rabinovich, A., Vedaldi, A., Galleguillos, C., Wiewiora, E., and Belongie, S. (2007). Objects in context. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1–8. ISSN 1550-5499.

Santana, T. M. H. C., Machado, A. M. C., Araújo, A. A., and dos Santos, J. A. (2016). Star: A contextual description of superpixels for remote sensing image classication. In *Iberoamerican Congress on Pattern Recognition (CIARP)*.

Santana, T. M. H. C., Nogueira, K., Machado, A. M. C., and dos Santos, J. A. (2017). Deep contextual description of superpixels for aerial urban scenes classification. In *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE.

Shotton, J., Winn, J., Rother, C., and Criminisi, A. (2009). Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *International Journal of Computer Vision (IJCV)*, 81(1):2--23. ISSN 0920-5691.

Silva, F. B., Goldenstein, S., Tabbone, S., and Torres, R. d. S. (2013). Image classification based on bag of visual graphs. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 4312–4316.

Sivic, J. and Zisserman, A. (2003). Video google: a text retrieval approach to object matching in videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1470–1477 vol.2.

Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, 45(4):427--437. ISSN 0306-4573.

Stehling, R. O., Nascimento, M. A., and Falcão, A. X. (2002). A compact and efficient image retrieval approach based on border/interior pixel classification. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, CIKM '02, pages 102--109, New York, NY, USA. ACM.

Strat, T. M. and Fischler, M. A. (1991). Context-based vision: recognizing objects using information from both 2d and 3d imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 13(10):1050–1065. ISSN 0162-8828.

Stuckens, J., Coppin, P. R., and Bauer, M. E. (2000). Integrating contextual information with per-pixel classification for improved land cover classification. *Remote Sensing of Environment*, 71(3):282 – 296. ISSN 0034-4257.

Torralba, A. (2003). Contextual priming for object detection. *International Journal of Computer Vision (IJCV)*, 53(2):169–191. ISSN 0920-5691.

Torres, R. d. S. and Falcão, A. X. (2006). Content-based image retrieval: theory and applications. *Revista de Informática Teórica e Aplicada*, 13(2):161--185.

Tso, B. C. K. and Mather, P. M. (1999). Classification of multisource remote sensing imagery using a genetic algorithm and markov random fields. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 37(3):1255–1260. ISSN 0196-2892.

Unser, M. (1986). Sum and difference histograms for texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 8(1):118--125. ISSN 0162-8828.

Vargas, J. E., Falcão, A. X., dos Santos, J. A., Esquerdo, J. C. D. M., C., C. A., and Antunes, J. F. G. (2015). Contextual superpixel description for remote sensing image classification. In *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE.

Vasconcelos, L. C. d. S., Felix, G. D. N., and Ferreira, F. H. (2007). Aspectos gerais sobre região e o processo de urbanização brasileira. *Espacio y Desarrollo*, (19):161–178.

Wilkinson, G. G. (2005). Results and implications of a study of fifteen years of satellite image classification experiments. *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 43(3):433--440.

Wulder, M., Cranny, M., Dechka, J., White, J., et al. (2004). An illustrated methodology for land cover mapping of forests with landsat-7 etm+ data. *Canadian Forest Service, Pacific Forestry Centre.*

Yu, Q., Gong, P., Clinton, N., Biging, G., Kelly, M., and Schirokauer, D. (2006). Object-based detailed vegetation classification with airborne high spatial resolution remote sensing imagery. *Photogrammetric Engineering & Remote Sensing*, 72(7):799--811.

Zhang, Z. and Saligrama, V. (2017). Prism: Person reidentification via structured matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(3):499–512. ISSN 1051-8215.