International Doctorate School in
Information and Communication Technologies

Department of Information Engineering and Computer Science
University of Trento

ADVANCED CLASSIFICATION METHODS FOR UAV IMAGERY

Abdallah ZEGGADA

Advisor: Prof. Farid Melgani, University of Trento

Words cannot thank them enough, I would just extend profound thanks to

Particularly my supervisor Dr. Farid Melgani for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge.

Besides my advisor, I would like to thank Mesay, Satoru and Souad for their collaboration in this work.

My sincere thanks also goes to my dear friends as well as anyone who helped the cause of this work in any possible way.

Last but not the least, I would like to thank my family, especially my parents, for giving birth to me at the first place and supporting me spiritually throughout my life.

.

Abstract

The rapid technological advancement manifested lately in the remote sensing acquisition platforms has triggered many benefits in favor of automated territory control and monitoring. In particular, unmanned aerial vehicles (UAVs) technology has drawn a lot of attention, providing an efficient solution especially in real-time applications. This is mainly motivated by their capacity to collect extremely high resolution (EHR) data over inaccessible areas and limited coverage zones, thanks to their small size and rapidly deployable flight capability, notwithstanding their ease of use and affordability. The very high level of details of the data acquired via UAVs, however, in order to be properly availed, requires further treatment through suitable image processing and analysis approaches.

In this respect, the proposed methodological contributions in this thesis include: i) a complete processing chain which assists the Avalanche Search and Rescue (SAR) operations by scanning the UAV acquired images over the avalanche debris in order to detect victims buried under snow and their related objects in real time; ii) two multilabel deep learning strategies for coarsely describing extremely high resolution images in urban scenarios; iii) a novel multilabel conditional random fields classification framework that exploits simultaneously spatial contextual information and cross-correlation between labels; iv) a novel spatial and structured support vector machine for multilabel image classification by adding to the cost function of the structured support vector machine a term that enhances spatial smoothness within a one-step process. Conducted experiments on real UAV images are reported and discussed alongside suggestions for potential future improvements and research lines.

Contents:

Chapter 1. Introduction and Thesis Overview
1.1. Remote Sensing Platforms and Applications..... 2
1.2. Issues, Solutions and Thesis Organization..... 5
1.3. Multilabel Classification..... 6
1.4. References..... 8
Chapter 2. Deep Learning for Assisting Avalanche Search and Rescue Operations
2.1. Introduction..... 12
2.2. Methodology..... 14
2.2.1. Pre-processing..... 14
2.2.2. Feature Extraction..... 15
2.2.2.1. Convolutional Layer..... 16
2.2.2.2. Pooling Layer..... 17
2.2.2.3. Fully Connected Layer..... 17
2.2.3. Classifier..... 18
2.2.4. Post-Processing..... 19
2.3. Data and Experimental Setup..... 21
2.3.1. Dataset Description..... 21
2.3.2. Setup..... 23
2.4. Results and Discussions..... 24
2.4.1. Experiments without pre-processing..... 24
2.4.2. Experiments with pre-processing..... 26
2.4.3. Experiments with Markovian post-processing..... 29
2.4.4. Computation time..... 32
2.4.5. Conclusion..... 33
2.5. References 34
Chapter 3. Multilabel Deep Learning Strategies for Imagery Description
3.1. Introduction..... 38
3.2. Methodological Overview..... 39
3.2.1. Image Coarse Description..... 39
3.2.2. Multilabeling UAV Images with Convolutional Neural Networks..... 41
3.2.3. Otsu's Thresholding Algorithm..... 43
3.3. Multilabeling UAV Images with Autoencoders Neural Networks..... 44
3.3.1. Tile Representation strategies..... 45
3.3.1.1. Bag of Visual Words..... 45
3.3.1.2. Wavelet Transform..... 45
3.3.1.3. Histogram of oriented gradients..... 46
3.3.2. Autoencoder neural network..... 46
3.4. Experimental Results..... 48
3.4.1. Dataset Description and Experimental Setup..... 48
3.4.2. Results and Discussion..... 49
3.4.2.1 Results of the Convolutional Neural Networks based Method..... 49
3.4.2.2 Results of the Autoencoder Neural Networks based Method..... 51
3.5 Conclusion..... 56
3.6. References..... 57
Chapter 4. Multilabel Conditional Random Field Classification
4.1. Introduction 60
4.2. Proposed Method..... 62
4.2.1. Monolabel Classification with CRF..... 62
4.2.2. CRF for Multilabel Classification..... 63
4.3. Experimental Validation..... 66
4.3.1. Dataset Description and Experimental Setup..... 66
4.3.2. Evaluation metrics and experimental results..... 67

4.4. Conclusion..... 71
4.5. References..... 71
Chapter 5. Spatial and Structured SVM for Multilabel Image Classification
5.1. Introduction 74
5.2. Problem Formulation and Tools..... 77
5.2.1 Multilabel Classification for EHR Imagery..... 77
5.2.2 Tile Representation..... 78
5.2.3 Reviews of Structured SVM..... 79
5.2.3. Structured SVM with Spatial Embedding..... 81
5.3. Experimental Results..... 84
5.3.1. Dataset Description..... 84
5.3.2. Experimental Setup..... 85
5.3.3. Experimental Results..... 86
5.3.3.1 Results of Dataset..... 86
5.3.3.2. Results of Dataset..... 90
5.4. Conclusion..... 93
5.5. References..... 94
Chapter 6. Conclusions..... 98
6.1. References.....101

List of Tables:

- Table 2.1. HMM notations in accordance to our detection problem.
- Table 2.2. General description of experiments conducted.
- Table 2.3. Classification results for the first dataset (Experiments 1 - 3).
- Table 2.4. Classification results for the second dataset at resolution of 224x224 (Experiment 4).
- Table 2.5. Classification results for the second dataset at 640x480 and 1280x720 resolutions (Experiments 5 and 6).
- Table 2.6. Classification results for the second dataset at 1920x1080 and 3840x2160 resolutions (Experiments 7 and 8).
- Table 2.7. State transition matrix.
- Table 2.8. HMM detection result at resolution of 244x224.
- Table 2.9. HMM detection results at VGA and 720p resolutions.
- Table 2.10. HMM detection results at 1080p and 4K resolutions.
- Table 2.11. Detection speed (number of frames per second) for the second dataset.
- Table 3.1. Comparison of classification accuracies in terms of sensitivity (SENS) and specificity (SPEC) between the different implementations. Computational time per tile is also reported for each strategy.
- Table 3.2. Best threshold values yielded by the Otsu's method for each of the RBFNN output classes.
- Table 3.3. Comparison of classification accuracies in terms of sensitivity (SENS) and specificity (SPEC) between the different implementations. Computational time per tile is also reported for each strategy.
- Table 3.4. Best threshold values yielded by the Otsu's method for each of the AE-MLP based output classes.
- Table 3.5. Sensitivity (SENS) and specificity (SPEC) accuracy achieved for each class by the reference method , the CNN-RBFNN and WAV-BOW with and without the multilabeling layer (ML) classifiers on dataset 1 (Povo).
- Table 3.6. Sensitivity (SENS) and specificity (SPEC) accuracy achieved for each class by the reference method, the CNN-RBFNN and RGB-BOW with and without the multilabeling layer (ML) classifiers on dataset 1 (Civezzano).
- Table 4.1. Sensitivity (SENS), specificity (SPEC) and average (AVG) accuracies in percent obtained by the different classification methods on datasets 1 and 2.
- Table 4.2. class-by-class accuracy performances achieved by the three models on dataset 1 (Civezzano).
- Table 4.3. class-by-class accuracy performances achieved by the three models on dataset 2(Munich).
- Table 5.1. Number of class occurrence on dataset 1 (Civezzano).

- Table 5.2. Number of class occurrence on dataset 2 (Munich).
- Table 5.3. Classification accuracies of the three classifiers on dataset 1 (Civezzano).
- Table 5.4. Statistical comparison based on MCNEMAR's test between the three models on dataset 1 (Civezzano).
- Table 5.5. Classification accuracies of the three classifiers on dataset 2 (Munich).
- Table 5.6. Statistical comparison based on MCNEMAR's test between the three models on dataset 2 (Munich).
- Table 5.7. class-by-class accuracy performances achieved by the three models on dataset 1 (Civezzano).
- Table 5.8. class-by-class accuracy performances achieved by the three models on dataset 2 (Munich).
- Table 5.9. Sensitivity (SENS), specificity (SPEC) and average (AVG) accuracies in percent obtained by the different classification methods on datasets 1and 2.

List of Figures:

- Fig. 1.1. Example of imagery spatial resolution differences of three acquisition platforms covering the same area.
- (a) Satellite; (b) Airborne; (c) Unmanned aerial vehicle.
- Fig. 1.2. Example of unmanned aerial vehicles usages in different application environments.
- Fig. 2.1. Block diagram of the overall system.
- Fig. 2.2. Example of CNN architecture for object recognition.
- Fig. 2.3. An example of operation performed by the neurons at a spatial location of the input and the resulting activation maps.
- Fig. 2.4. State transition diagram (object pointed by a yellow arrow is a jacket used to simulate top half of buried victim).
- Fig. 2.5. Example of positive (top) and negative (bottom) images from the first dataset. Objects of interest (partially buried skis (left) and top half buried victim (right)) are marked with yellow circle.
- Fig. 2.6. Positive (left) and negative (right) frame snapshots from the second dataset. Objects of interest (skis, jacket to simulate bottom half buried victim, and ski pole) are marked by yellow circle.
- Fig. 2.7. Example of correctly classified negative (top left) and positive (top right), false positive object marked in yellow rectangle (bottom left), and false negative object marked by red rectangle (bottom right).
- Fig. 2.8. Example showing the pre-processing step. The image on top shows a frame being scanned by a sliding window while the image on the bottom highlights a region (marked by blue rectangle), centered around a window (marked by cyan rectangle) selected for further processing.
- Fig. 2.9. Snapshot of frames with correct positive (left) and negative (right) detection results at the VGA resolution from the second dataset. Regions of a frame containing an object are shown with green rectangle.
- Fig. 2.10. Examples of false positive (left) and false negative (right) frame snapshots at VGA resolution. Yellow arrows indicate false positive regions in a frame whereas red arrows show missed objects in a frame.
- Fig. 2.11. Example of positive change by HMM. Sequence of white and black squares on top indicate label of successive frames. White square indicates a frame has object of interest whereas black square indicates the opposite. The frame where the change happened is highlighted by red dotted rectangle and the corresponding frame in the bottom. The frame for which SVM made wrong decision is shown in bottom left (the object in the frame, skis in this case, is indicated by red arrow) whereas the same frame corrected by HMM is shown in the bottom right (the object in the frame is indicated by green arrow). Note that object is not localized since post-processing decision is made at the frame level.
- Fig. 2.12. Example of negative change by HMM. Sequence of white and black squares on top indicate label of successive frames. White squares indicate a frame has object of interest whereas black squares indicate the opposite. Frame where the change happened is

highlighted by red dotted rectangle. The frame for which SVM made the right decision, with the object localized in a green rectangle, is shown in the bottom left. The same frame for which HMM made wrong decision is shown in bottom right.

- Fig. 2.13 Bar graph showing the change in accuracy and detection rate as resolution increases.
- Fig. 3.1. Flow chart of the multilabel coarse classification framework.
- Fig. 3.2. Flow chart of the Inception module.
- Fig. 3.3. Global flowchart of the proposed classification scheme.
- Fig. 3.4. Graphical histogram illustration of the OTSU thresholding technique.
- Fig. 3.5. Flow chart of the multilabel classification method based on the autoencoder network.
- Fig. 3.6. Architecture of an AE network.
- Fig. 3.7. The effect of the encoding rate on the average accuracy for (a) dataset 1 (Povo) and (b) dataset 2 (Civezzano).
- Fig. 3.8. Example of two MLP output histograms and their decision threshold computed using OTSU's method for WAV-AE,(a) class Asphalt, (b) class Grass in dataset one (Povo).
- Fig. 3.9-1 Qualitative map results of the 13 object classes obtained by the coarse WAV-AE (ML) classification technique on one of the test images of dataset 1(Povo) along with its related ground truth and original image.
- Fig. 3.9-2 Qualitative map results of the 13 object classes obtained by the coarse GoogLeNet-RBFNN (ML) classification technique on one of the test images of dataset 1(Povo) along with its related ground truth and original image.
- Fig. 3.9-3 Qualitative map results of the 14 object classes obtained by the coarse Alex-RBFNN (ML) classification technique on one of the test images of dataset 2 (Civezzano) along with its related ground truth and original image.
- Fig. 4.1. Flow chart of the multilabel classification method
- Fig. 4.2. The correlation and traditional spatial neighboring information in the proposed ML-CRF.
- Fig. 4.3. Average accuracy versus spatial parameters λ_1, λ_2 achieved by the Full-ML-CRF method on (a) dataset 1 (Civezzano) and (b) dataset 2 (Munich).
- Fig. 4.4. Examples of multilabel classification maps obtained by the three classification methods (ML-Unary, ML-CRF, Full-ML-CRF) on two test images of dataset 2(Civezzano), along with their related ground truth and original image.
- Fig.5.1. General block diagram of the proposed framework for multilabel tile-wise description.
- Fig. 5.2. Output graph structure obtained for dataset 1 (Civezzano).
- Fig. 5.3. First-order neighborhood system. The black tile is the center i and the yellow tiles are the elements of neighborhood N_i.

- Fig. 5.4. (a) Ground truth of a test image of dataset 1 (Civezzano). (b, c, d) Classification results of the test image obtained by SVM(b), SSVM(c) and SSSVM(d). Each tile is colored according to the Hamming distance between the ground truth and prediction.
- Fig. 5.5. Multilabel classification map obtained by the SSSVM classifier on one of the test images of dataset 1 (Civezzano), along with its related ground truth and original image.
- Fig. 5.6. Output graph structure obtained for dataset 2 (Munich).
- Fig. 5.7. Average accuracy versus spatial parameter $\boldsymbol{\lambda}$ (of SSSVM model) for dataset 1 (Civezzano) (a) and dataset 2 (Munich) (b).
- Fig. 5.8. (a) Ground truth of a test image of dataset 2 (Munich). (b, c, d) Classification results of the test image obtained by SVM(b), SSVM(c) and SSSVM(d). Each tile is colored according to the Hamming distance between the ground truth and prediction.
- Fig. 5.9. Multilabel classification map obtained by the SSSVM classifier on one of the test images of dataset 2 (Munich), along with its related ground truth and original image.

Glossary:

UAV: unmanned aerial vehicle

VHR: very high resolution

EHR: extremely high resolution

ANN: artificial neural network

CNN: convolutional neural network

RGB: red, green, blue

SVM: support vector machine

HMM: hidden markov models

HOG: histogram of oriented gradients

RBFNN: radial basis function neural network

MLP: multilayer perceptron network

SAR: synthetic aperture radar

LIDAR: light detection and ranging

AE: autoencoder network

OTSU: Otsu's thresholding method

WAV: wavelets transform

BOW: bag of visual words

SENS: sensitivity

SPEC: specificity

AVG: average

SSVM: structured support vector machine

SSSVM: spatial and structured support vector machine

MRF: Markov random field

CRF: conditional random field

ML-CRF: multilabel conditional random field

Full-ML-CRF: full multilabel conditional random field

GAN: generative adversarial network

IOWA: induced ordered weighted averaging

Chapter I

Introduction and Thesis Overview

1.1. Remote Sensing Platforms and Applications

There exist several definitions for Remote Sensing; all of those definitions in the widest sense are concerned with information acquisition and measuring the reflected energy of areas and objects under monitoring without a direct contact. Remote Sensing can be divided into two categories, active and passive. Active remote sensing is when a signal is first emitted then the resulting reflected energy is analyzed. Active image sensors such as, satellites- and aircraft-based sensors emit their own energy (radiation pulse) which is transmitted to the object (i.e., earth surface) creating a backscatter that bounces and returns back to the emitting sensors again. This technique has shown great potential in collecting data whenever it is needed, day or night, without any time or atmospheric constraints, however it is energy demanding. Two common examples among the variety of existing active sensors that are widely used within the remote sensing community are Radars, and light detection and ranging (LiDAR). Radar and LiDAR are respectively, radio waves- and laser- based sensors used to collect information about the target under surveillance, such as distance, range, velocity, angle, and elevation. On the other hand, passive remote sensing simply acquires information using the energy that the target reflects (i.e., the sun, electromagnetic energy). It exploits the targets' own energy, which is not energy demanding but depends on the external sources.

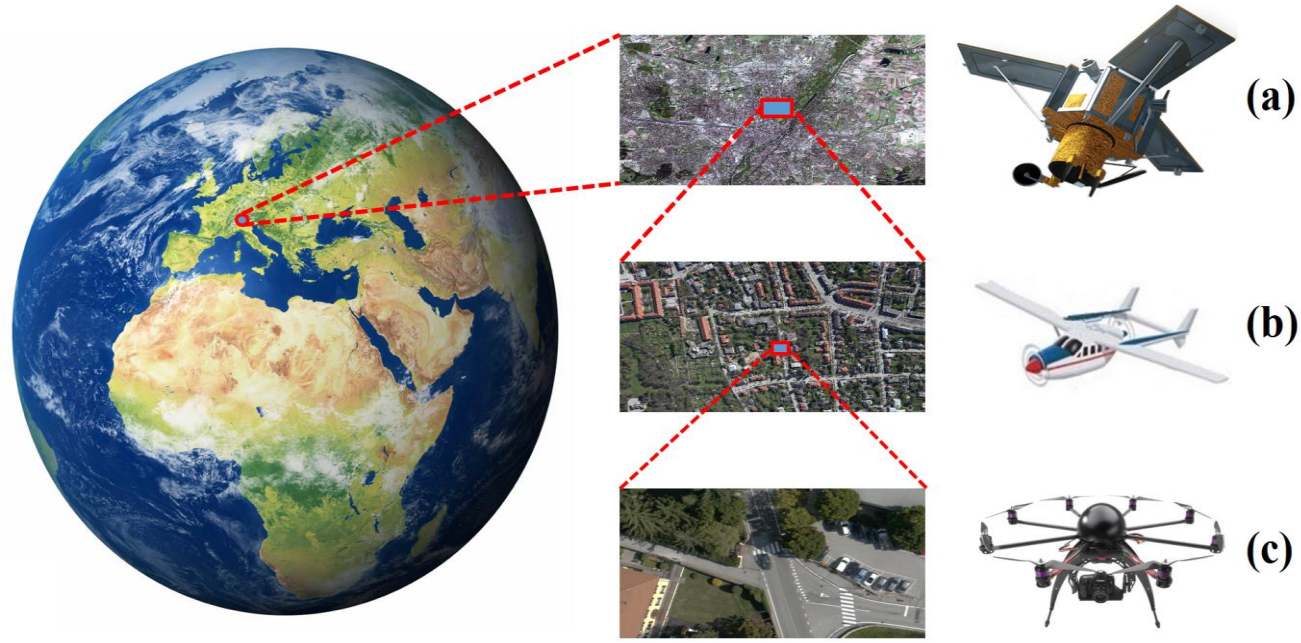


Fig. 1.1. Example of imagery spatial resolution differences of three acquisition platforms covering the same area. (a) Satellite; (b) Airborne; (c) Unmanned aerial vehicle.

Over the last few decades, satellites have reached a good level of technological progress in terms of spatial resolution (i.e. Landsat 8 with 15 meters of spatial resolution). Such acquired information from satellites have been used widely, and showed to be efficient in various applications such as (e.g., forestry, cartography, forestry, climate, geology) [1]. However since the lunch of IKONOS the first very high resolution (VHR) optical satellite with (82-centimeter resolution and multispectral imagery of 4 meters), and then the last generation of VHR (i.e.,

GeoEye-1 -2, QuickBird, WorldView-1 -2 & 3) a new level of spatial complexity has been exposed.

In particular, very high-resolution (VHR) satellites imagery has shown a remarkable performance and efficiency in several applications. For instance, in [2], using Multitemporal High-Resolution IKONOS and GeoEye-1 Satellites data, authors proposed a segmentation-based method exploiting support vector machines (SVMs) classifier to map urban ecological conditions, and to determine land cover changes in a dense urban core from 2000 to 2009. In [3], was presented an identification method for archaeological buried remains within a dense presence of vegetation using VHR Quickbird satellite data. In [4], authors introduced a bayesian classification framework of urban land use. It incorporates its open source data from very high resolution (VHR) GeoEye stereo satellite imagery. In [5], the authors presented an automatic moving vehicles information extraction framework from Single-Pass WorldView-2 VHR Imagery.

Similarly, airborne technologies equipped with VHR SAR and LiDAR sensors such as, helicopters, fixed wing aircrafts and single-rotor helicopters have shown to be very suitable for several urban landscape 3D modeling gaining a rapid growth in recent years. For instance, in [6], authors present a three dimensional reconstruction method for large multilayer interchange bridges using airborne light detection and ranging (LiDAR) data. In [7], authors put forth an image reconstruction algorithm for airborne downward-looking sparse linear array 3-D synthetic aperture radar (DLSLA 3-D SAR). Another Airborne LiDAR application for large urban environments was introduced in [8], particularly, a multiscale grid algorithm for the detection and reconstruction of building roofs. It derives benefits from making use of an iterative morphological interpolation exploiting a gradually increasing scale from large to small scales. Afterwards, the resulting building roofs features are segmented and reconstructed according to their elevation. In [9], it was introduced a woodland canopy reconstruction framework of digital terrain model (DTM) from an airborne E-SAR sensor. Another very interesting work has been presented in [10], where Paris et al. proposed a 3-dimentional model-based algorithm for tree top height estimation for high-spatial-density LiDAR data.

Notably, airborne sensors have proven to be very efficient in providing color/ infrared very high-resolution imagery. Outfitted with different sensors and technologies (i.e., camera, LIDAR), they are capable to acquire very accurate and high quality geometrical data of the observed areas. On the contrary, satellites imagery have a smaller resolution capability compared to airborne sensors. However, their coverage capacity is extremely large due to their very high altitude, saving the mosaicking and geocoding process that is required for aerial photography platforms when the scanned area is large (Fig. 1.1). Yet, satellites suffer from (i) cloud cover and (ii) specific fixed-timing acquisitions. As a result, airborne platforms are emerging as a potential strong alternative to conventional satellites acquisition technologies in both spectrally and spatially very high-resolution remote sensing imagery. Despite the afforested, airborne high cost and complicated flight procedures, in addition of their limited flight altitude, make of them a less appropriate acquisition platform for critical applications such as routine maintenance and emergency response.

By contrast, UAVs were originally developed for military purposes, where their mission plans are designed “on-demand” in prospect to survive in critical mission conditions. They are

characterized by low-cost, fast deployment and custom-made capacity, since they can be equipped with appropriate sensors according to the requirements of several distinct missions. Moreover, they play a complementary role in supporting satellites to cover inaccessible small areas, by taking advantage of their very detailed scanning capability at a sub meter / centimeter level i.e., extremely high image spatial resolution (EHR). Furthermore, UAVs grant the flexibility to operate within a swarm forming a complex acrobatic group flights that communicate and cooperate synchronously in mid-air. As to cope with their limited coverage scale, UAVs collect data in a timely manner providing a wider coverage capacity. Thanks to their smaller size and easier air traffic management compared to airborne platforms, UAVs stand out to be a favorable alternative to the traditional

Fig. 1.2. Example of unmanned aerial vehicles usages in different application environments.

field visit and ground surveying, and a very suitable acquisition platform to collect extremely high-resolution (EHR) data for several image classification and analysis applications such as, urban and environmental monitoring, precision farming, surveillance, and search and rescue missions (Fig. 1.2).

For instance, in mapping and cartography applications, Márquez *et al.* in [11], presented a framework that generates cadastral cartography for small urban/rural locations taking as input unmmaned aerial vehicles imagery. Pattern recognition algorithms were applied to add pictorial terrain components in the resulting cadastral plans. In [12], it was proposed a 3-D mapping system via a UAV flying from low altitude equipped with cameras, a laser scanner, an inertial measurement unit, and Global Positioning System (GPS). In [13], authors presented a novel range-dependent map-drift algorithm (RDMDA) developed for synthetic aperture radar (SAR) systems mounted on unmanned aerial vehicles (UAVs). One more interesting work in [14] put forth a stereo vision path planning assistant system for unmanned ground vehicle (UGV) in GPS-denied environments based on multiunmanned aerial vehicles. As for the inspection and public safety applications, a notable interest has been dedicated to UAVs platforms. In [15], a navigation prototype is developed to localize drone positions in indoor environments for industrial applications. Another automatic UAV inspection system is presented in [16]. This platform aims towards resource assessment and defect detection in large-scale photovoltaic systems over large geographical areas. Furthermore, in [17], Pinto *et al.* introduced an online inspection video monitoring system for industrial installations based on areial collaborative communications between small UAVs (i.e., UAV-to-UAV network). Concerning public safety, UAV aerial support provides a very adequate system to acquire information in unreachable areas. Aside from guaranteeing the safety of human operators from direct contact with danger in emergency

situations. For instance, [18] presented an autonomous UAV system equipped with thermal and digital imagery sensors for human body detection in disaster scenarios. The detected victim positions are geo-located within a map of points and then sent to the rescuing teams. [19] put forth a 3D UAV-based modeling for collapsed buildings rapid response situations in urban scenes. In [20], a semi-autonomous indoor fire-fighting to ensure firefighters safety is presented. Other works based on UAVs dealing with emergency and disaster fast response can be found in [21]-[22].

Moreover, UAVs have been finding their way in precision agricultural applications. By incorporating thermal, near-infrared and visible spectral bands to UAVs platform sensors, they serve as an alternative to traditional field visits to measure vegetation, health condition, and water stress indices, especially for large farmlands surveying. For instance, in [23] Katsigiannis *et al.* presented an autonomous multi-sensor UAV system that provides spectral information related to water management for pomegranate orchards. In [24], authors put forth a low cost multi-spectral vegetation classification platform of UAVs equipped with a set of exchangeable filters over a camera connected to a Raspberry Pi. The implemented prototype have shown to be able to distinguish between two types of vineyards and different species of plants. Authors in [25], presented a row and water front detection architecture combining UAV and thermal-infrared imagery for furrow irrigation monitoring. Still in precision farming, an interesting UAV-based network system for early stage disease detection was presented in [26]. In [27], authors presented an autonomous timely monitoring method for close range UAV citrus greening disease detection. It exploits depth-invariant machine learning models to distinguish between healthy and infected plant leaves. Very promising results have been yielded with validation accuracies up to 93%. Furthermore, plenty of other works involving UAVs have been undertaken in a wide range of areas, including but not limited to archaeology [28], radiation monitoring [29], ecological protection [30], and environmental monitoring in general [31],[32]. For further potential remote sensing applications of UAVs, we refer the reader to relevant state of the art.

1.2. Issues, Solutions and Thesis Organization

As with the evolution of UAVs technology which has known a dramatic increase for civilian applications over the last decade. UAVs have displayed a remarkable efficiency as a safer and job faster alternative to traditional field visit and ground surveying in urban scenarios. Such acquisition systems, despite their effectiveness, convey an extremely high-resolution (EHR) images with a very accurate geometrical analysis for the objects present in the scene, entailing a challenging huge amount of details to be processed and exploited (i.e., hundreds of spectral bands). This calls for the need to adopt new processing and analysis techniques that are capable to exploit the full potential of this huge amount of acquired information. Thus acquiring some main points to be dealt with, like:

- The first one recalls the fact that the more the resolution of the acquired data increases significantly, the most likely that monolabel processing methods become inadequate to meet satisfactory classification accuracies and needs. Indeed, in UAV scene description applications, typically an image frame tends to contain several objects with complex distribution structures (i.e., between-class similarity and within-class diversity) due to the

large amount of details of the extremely high resolution data. Indeed, most of the monolabel processing methods have been applied successfully within constrained environments with moderate resolutions and a limited number of classes of objects to be predicted. This calls for the need to design new processing methods suggesting the use of multilabeling approaches.

- The second one is that the objects in EHR imagery manifest two particular topologies of correlation, which could be exploited to enhance the recognition process. They are: i) the intrinsic correlation between objects (i.e., when a particular object is present, it is likely that another is present as well); ii) the spatial correlation between adjacent decisions, which may be provided by a given classifier.
- Third, as a matter of fact, most UAV-oriented applications require real-time processing overhead. Therefore, the respective processing algorithms are ought to be implemented to meet such requirement.

1.3. Multilabel Classification

As hinted earlier, usually imagery analysis and classification applications for data acquired over urban areas are composed by a list of objects, that when are put together they describe the conventional scene. In order to address this, we extend the interest into describing several classes at the same time. Therefore, multilabel approach presents an alternative to the single object description making the classification task more informative and generalized.

In particular, the scope of this dissertation is mainly focused on describing extremely high resolution images in urban scenarios. Consequently, we dedicated three chapters to deal with multilabel classification, which is a subject that has attracted a scarce attention with respect to monolabel (i.e., binary and multiclass) classification. In fact, most of the processing methods and frameworks based on statistical modeling are mainly designed for monolabel tasks. Particularly, within the remote sensing community, monolabel classification and object detection has drawn the attention of most of researchers generating the largest number of published papers. In contrast to multiclass classification where the labels are mutually exclusive (only one object class per sample), multilabel classification associates to a single sample one or more than one label simultaneously (i.e., a list of object classes). As a result, as the number of classes exponentially increases, so does the classification output space complexity. A common way to address the multilabel issue, is to handle each class separately (i.e., in a class specific manner) then the resulting output of all the classes together is the final outcome. Such approach is very time consuming due to the number of algorithms (i.e., classifiers) that would be called simultaneously. Another critical point to be highlighted in multilabel classification is the inter-class correlation information between labels and how to exploit them effectively through the classification process. Therefore, we will try to benefit from handling all the object classes together rather than separately in order to extract some interaction rules between labels.

The other challenging part of this task, which makes it not easily achievable, is the large number of variable and heterogeneous complex features with very similar spectral characteristics in extremely high resolution data. As hinted earlier, it is clearly essential that we must take into

consideration the capability of the proposed approach in terms of its ability to resolve both the multilabel classification and the EHR spectral feature complexities characterizing this task.

In this respect, having devoted this first chapter as an introductive part to cover the different aspects of the topic. The rest of the thesis is organized into five chapters, in which the next chapter presents an interesting public safety application, namely, a framework based on UAV imagery for assisting avalanche search and rescue operations, whereas in the remaining three chapters, we put forth three proposed multilabeling classification schemes in urban scenarios. In more details, in Chapter 2, we put forth a victim detection and localization framework to assist rescue teams in avalanche search and rescue (SAR) operations my means of a UAV equipped with digital cameras. The proposed framework consists of three steps, 1) a pre-processing step to select regions of interest within the acquired images. 2) we use convolutional neural networks (CNN) for feature extraction and a Support Vector Machine (SVM) on top of it for classification. 3) a post-processing step based on a Hidden Markov Model is used to improve the prediction output of the SVM classifier.

In Chapter 3, we propose a tile-based pipeline that takes advantage of a tile-based coarse description technique providing global results for the considered EHR images. Considering the conventional pixel-based and segment-based descriptors that may raise the problem of intra-class variability particularly when dealing with the multilabel object detection. Coarse description strategy does not aim to assign to each single pixel or pattern descriptor a label, but it simply describes a query image or the specific investigated tile within the image by the list of objects present in it. In this context, two Deep Neural Networks (DNNs) architectures have been investigated namely, convolutional neural networks [33], and autoencoders [34] which have become one of the most promising and fast growing techniques within the machine learning community in the last few years. Moreover, for the multilabeling requirements, we introduce a multilabel layer that has been integrated on top of the proposed architectures to increase the obtained results.

Chapter 4, proposes a novel method that aims to address the multilabel classification problem by combining two types of information, namely, the spatial correlation between adjacent tiles as well as the interclass correlation between all class labels, we define this method as full multilabel conditional random field (Full-ML-CRF) model. This Full-ML-CRF method reformulates both the interclass and spatial correlation information as an energy minimization problem based on the conventional conditional random fields model. Iterated Conditional Modes (ICM) algorithm is adopted for the optimization problem. In particular, after the subdivions of the query images into a grid of equal tiles, posterior probability predictions are generated for each tile by means of an opportune multilabel classification method. Afterwards, we feed the resulting predictions (i.e., classification label maps) to the Full-ML-CRF to improve the obtained prediction results. Such approach is able to bridge the gap up to a reasonable extent, between the likely high semantic content of the UAV-grabbed images and their spectral information. Furthermore, it enhances the spatial and the interclass smoothness of the resulting label maps of the multilabel classification framework. In Chapter 5, we put forth a novel multilabel classification approach based on a structural Support Vector Machine (SSVM) classifier. Unlike state-of-the-art

techniques, which typically perform the multilabel classification task by separating the spectral features and the spatial contextual information into two different processing methods. We propose here a completely different alternative scheme called Spatial and Structured Support Vector Machine (SSSVM). Aiming at expanding the coarse tile-based multilabel spectral features classification scheme to incorporate the spatial information within the recognition process, we propose to merge both information, spectral and contextual within the same cost function. The resulting framework operates as an extension to the conventional Structured Support Vector Machine (SSVM) by integrating the structured output of the SVM and the spatial information simultaneously during the training phase. Finally, Chapter 6 draws final conclusions of the discussed methods and put forward some open issues and potential ameliorations for future developments.

This dissertation has been written supposing that the Reader is familiar with the basic concepts regarding the image processing, remote sensing and pattern recognition fields. Otherwise, the Reader is recommended to consult the references which are available at the end of each chapter of this dissertation. They are useful to give a complete and well-structured overview about the topics discussed throughout the manuscript. The following chapters have been written in such a way to be independent between each other to give to the Readers the possibility to read only the chapter/s of interest, without loss of information.

1.4. REFERENCES

[1] M. A. Wulder, J. C. White, S. N. Goward, J. G. Masek, J. R. Irons, M. Herold, W. B. Cohen, T. R. Loveland and C. E. Woodcock, “Landsat continuity: Issues and opportunities for land cover monitoring,” Remote Sensing of Environment, vol. 112, no. 3, pp. 955-969, 2008.

[2] J. Haas and Y. Ban, "Mapping and Monitoring Urban Ecosystem Services Using Multitemporal High-Resolution Satellite Data," in *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 2, pp. 669-680, Feb. 2017.

[3] R. Lasaponara and N. Masini, "Identification of archaeological buried remains based on the normalized difference vegetation index (NDVI) from Quickbird satellite data," in *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 3, pp. 325-328, July 2006.

[4] M. Li, K. M. de Beurs, A. Stein and W. Bijker, "Incorporating Open Source Data for Bayesian Classification of Urban Land Use From VHR Stereo Images," in *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. PP, no. 99, pp. 1-14.

[5] B. Salehi, Y. Zhang and M. Zhong, "Automatic Moving Vehicles Information Extraction From Single-Pass WorldView-2 Imagery," in *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 5, no. 1, pp. 135-145, Feb. 2012.

[6] L. Cheng, Y. Wu, Y. Wang, L. Zhong, Y. Chen and M. Li, "Three-Dimensional Reconstruction of Large Multilayer Interchange Bridge Using Airborne LiDAR Data," in *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 2, pp. 691-708, Feb. 2015.

[7] X. Peng, W. Tan, W. Hong, C. Jiang, Q. Bao and Y. Wang, "Airborne DLSLA 3-D SAR Image Reconstruction by Combination of Polar Formatting and L_1 Regularization," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 1, pp. 213-226, Jan. 2016.

[8] Y. Chen, L. Cheng, M. Li, J. Wang, L. Tong and K. Yang, "Multiscale Grid Method for Detection and Reconstruction of Building Roofs from Airborne LiDAR Data," in *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 10, pp. 4081-4094, Oct. 2014.

[9] C. S. Rowland and H. Balzter, "Data Fusion for Reconstruction of a DTM, Under a Woodland Canopy, From Airborne L-band InSAR," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 5, pp. 1154-1163, May 2007.

[10] C. Paris and L. Bruzzone, "A Three-Dimensional Model-Based Approach to the Estimation of the Tree Top Height by Fusing Low-Density LiDAR Data and Very High Resolution Optical Images," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 1, pp. 467-480, Jan. 2015.

[11] E. S. Márquez and A. V. Soto, "Cadastral stereo plotting production from unmmaned aerial vehicles imagery," *2017 First IEEE International Symposium of Geoscience and Remote Sensing (GRSS-CHILE)*, Valdivia, 2017, pp. 1-9.

[12] M. Nagai, T. Chen, R. Shibasaki, H. Kumagai and A. Ahmed, "UAV-Borne 3-D Mapping System by Multisensor Integration," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 3, pp. 701-708, March 2009.

[13] L. Zhang, M. Hu, G. Wang and H. Wang, "Range-Dependent Map-Drift Algorithm for Focusing UAV SAR Imagery," in *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 8, pp. 1158-1162, Aug. 2016.

[14] J. H. Kim, J. w. Kwon and J. Seo, "Multi-UAV-based stereo vision system without GPS for ground obstacle mapping to assist path planning of UGV," in *Electronics Letters*, vol. 50, no. 20, pp. 1431-1432, September 25 2014.

[15] K. J. Wu, T. S. Gregory, J. Moore, B. Hooper, D. Lewis and Z. T. H. Tse, "Development of an indoor guidance system for unmanned aerial vehicles with power industry applications," in *IET Radar, Sonar & Navigation*, vol. 11, no. 1, pp. 212-218, 1 2017.

[16] X. Li, Q. Yang, Z. Chen, X. Luo and W. Yan, "Visible defects detection based on UAV-based inspection in large-scale photovoltaic systems," in *IET Renewable Power Generation*, vol. 11, no. 10, pp. 1234-1244, 8 16 2017.

[17] L. R. Pinto, A. Moreira, L. Almeida and A. Rowe, "Characterizing Multihop Aerial Networks of COTS Multirotors," in *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 898-906, April 2017.

[18] P. Rudol and P. Doherty, "Human Body Detection and Geolocalization for UAV Search and Rescue Missions Using Color and Thermal Imagery," *2008 IEEE Aerospace Conference*, Big Sky, MT, 2008, pp. 1-8.

[19] S. Verykokou, A. Doulamis, G. Athanasiou, C. Ioannidis and A. Amditis, "UAV-based 3D modelling of disaster scenes for Urban Search and Rescue," *2016 IEEE International Conference on Imaging Systems and Techniques (IST)*, Chania, 2016, pp. 106-111.

[20] A. Imdoukh, A. Shaker, A. Al-Toukhy, D. Kablaoui and M. El-Abd, "Semi-autonomous indoor firefighting UAV," *2017 18th International Conference on Advanced Robotics (ICAR)*, Hong Kong, 2017, pp. 310-315.

[21] M. Abdelkader, M. Shaqura, M. Ghommem, N. Collier, V. Calo and C. Claudel, "Optimal multi-agent path planning for fast inverse modeling in UAV-based flood sensing applications," *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, Orlando, FL, 2014, pp. 64-71.

[22] C. Corrado and K. Panetta, "Data fusion and unmanned aerial vehicles (UAVs) for first responders," *2017 IEEE International Symposium on Technologies for Homeland Security (HST)*, Waltham, MA, 2017, pp. 1-6.

[23] P. Katsigiannis, L. Misopolinos, V. Liakopoulos, T. K. Alexandridis and G. Zalidis, "An autonomous multi-sensor UAV system for reduced-input precision agriculture applications," *2016 24th Mediterranean Conference on Control and Automation (MED)*, Athens, 2016, pp. 60-64.

[24] J. Natividade, J. Prado and L. Marques, "Low-cost multi-spectral vegetation classification using an Unmanned Aerial Vehicle," *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, Coimbra, 2017, pp. 336-342.

[25] D. Long, C. McCarthy and T. Jensen, "Row and water front detection from UAV thermal-infrared imagery for furrow irrigation monitoring," *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Banff, AB, 2016, pp. 300-305.

[26] P. Menendez-Aponte, C. Garcia, D. Freese, S. Defterli and Y. Xu, "Software and Hardware Architectures in Cooperative Aerial and Ground Robots for Agricultural Disease Detection," *2016 International Conference on Collaboration Technologies and Systems (CTS)*, Orlando, FL, 2016, pp. 354-358.

[27] S. K. Sarkar, J. Das, R. Ehsani and V. Kumar, "Towards autonomous phytopathology: Outcomes and challenges of citrus greening disease detection through close-range remote sensing," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, 2016, pp. 5143-5148.

[28] R. Saleri *et al*., "UAV photogrammetry for archaeological survey: The Theaters area of Pompeii," *2013 Digital Heritage International Congress (DigitalHeritage)*, Marseille, 2013, pp. 497-502.

[29] G. V. Prado and M. A. Y. Medina, "Design and Implementation of a Non-ionizing Radiation Measuring System Evaluated with an Unmanned Aerial Vehicle," *2015 Asia-Pacific Conference on Computer Aided System Engineering*, Quito, 2015, pp. 52-57.

[30] N. Li, D. Zhou, F. Duan, S. Wang and Y. Cui, "Application of unmanned airship image system and processing techniques for identifying of fresh water wetlands at a community scale," *2010 18th International Conference on Geoinformatics*, Beijing, 2010, pp. 1-5.

[31] Y. Lu, D. Macias, Z. S. Dean, N. R. Kreger and P. K. Wong*, "A UAV-Mounted Whole Cell Biosensor System for Environmental Monitoring Applications," in *IEEE Transactions on NanoBioscience*, vol. 14, no. 8, pp. 811-817, Dec. 2015.

[32] R. Ke, S. Kim, Z. Li and Y. Wang, "Motion-vector clustering for traffic speed detection from UAV video," *2015 IEEE First International Smart Cities Conference (ISC2)*, Guadalajara, 2015, pp. 1-5.

[33] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov 1998.

[34] L. Deng, and D. Yu, "Deep Learning: Methods and Applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197-387, 2014.

Chapter II

Deep Learning for Assisting Avalanche Search and Rescue Operations

Abstract*– Following an avalanche, one of the factors that affect victims chance of survival is the speed at which they get located and dugout. Rescue teams use techniques like trained rescue dogs and electronic transceivers to locate victims. However, the amount of resource required and time to deploy rescue teams are major bottlenecks to increase victim's chance of survival. Advances in the field of Unmanned Aerial Vehicles (UAVs) have enabled the use of flying robots equipped with sensors like optical cameras to assess damages caused by natural or manmade disasters and locate victims in the debris. In this chapter, we propose to assist avalanche search and rescue (SAR) operations with UAVs fitted with vision cameras. The sequence of images of the avalanche debris captured by the UAV is processed with a pre-trained Convolutional Neural Network (CNN) to extract discriminative features. A trained linear Support Vector Machine (SVM) is integrated at the top of the CNN to detect objects of interest. Moreover, we introduce a pre-processing method to increase the detection rate and a post-processing method based on a Hidden Markov Model to improve prediction performance of the classifier. Experimental results conducted on two different datasets at different levels of resolution show that detection performance increases with an increase in resolution while the computation time increases. Additionally, they also suggest that a significant decrease in processing time can be achieved thanks to the pre-processing step.*

2.1. Introduction

An avalanche, a large mass of snow detached from a mountain slope and sliding suddenly downward, kills more than one hundred fifty people worldwide [1] every year. According to the Swiss institute for snow and avalanche research, more than 90 percent of avalanche fatalities are occurred in uncontrolled terrain, like for example during off-piste skiing and snowboarding, [2]. Backcountry avalanches are mostly triggered by skiers or snowmobilers. Though it is rare, they can also be triggered naturally due to an increased load from a snow fall, metamorphic changes in snow pack, rock fall, and icefall. The enormous amount of snow carried at a high speed can cause a significant destruction to life as well as property. In areas where avalanches pose significant threat to people and infrastructure, preventive measures like snow fences, artificial barriers and explosives, to dispose avalanche potential snow packs, are taken to prevent and lessen their obstruction power.

Several factors account for the victims' survival. For example, victims can collide with obstacles while carried away by avalanches or fall over a cliff in the avalanches path and get physically injured. Once the avalanche stops, it settles like a rock and body movement is nearly impossible. Victims chance of survival depends on the degree of burial, presence of clear airway, and severity of physical injuries. Additionally, duration of burial is also a factor for victims' survival. According to statistics, 93 percent of victims survive if dugout within fifteen minutes of complete burial. Survival chance drops fast after the first fifteen minutes of complete burial. A "complete burial" is defined as where snow covers victims' head and chest; otherwise the term partial burial applies, [3]. Therefore, avalanche SAR operation is time critical.

Avalanche SAR teams use various ways to locate victims. For example, trained avalanche rescue dogs are used to locate victims by searching for pools of human scent rising up from the snow pack. Though dogs can be useful in locating victims not equipped with electronic transceivers, the number of dogs required and the time to deploy are constraints. If victims are equipped with electronic transceivers like ARVA (Appareil de Recherche de Victime d'Avalanche) part of skiers can immediately start searching for a missing member. But such

transceivers are powered by batteries and require experience to use. RECCO rescue system is an alternative to transceivers where one or more passive reflectors are embedded into clothes, boots, helmets, etc. worn by skiers and a detector is used by rescuers to locate the victims. Once area of burial is identified, a probe can be used to localize the victim and estimate the depth of snow to be shoveled. Additionally, an organized probe line can also be used to locate victims not equipped with electronic transceivers or if locating with the transceivers fails. But such technique requires significant man power and is a slow process. Recent advances in the field of UAVs have enabled the use of flying robots equipped with ARVA transceivers and other sensors to assist post-avalanche SAR operations, [4–6]. This has allowed to reduce search time and to search in areas that are difficult to reach and dangerous for rescuers.

In the literature, there are active remote sensing methods proposed to assist post-avalanche SAR operation. For example, the authors in [7] have shown that it is possible to detect victims buried under snow by using a Ground Penetrating Radar (GPR). Since human body has a high dielectric permittivity relative to snow, a GPR can uniquely image human body buried under snow and differentiate it from other man-made and natural objects. With the advent of satellite navigational system, Jan S, *et.al* [8], studied the degree to which a GPS signal can penetrate through the snow and be detected by a commercial receiver, hence a potential additional tool for quick and precise localization of buried victims. Following the work in [8], the authors in [9] also studied the performance of low cost High Sensitivity GPS (HSGPS) receivers available in the market for use in post-avalanche SAR operation. In a more recent work, Victor *et.al* [10] studied the feasibility of 4G-LTE signals to assist SAR operations for avalanche buried victims and presented a proof of concept that using a small UAV equipped with sensors that can detect cellphone signals, it is possible to detect victim's cellphone buried up to seven feet deep. Though there has been no research published documenting the use of vision based methods, a type of passive remote sensing methods, specifically for post-avalanche SAR operation, it is possible to find papers that propose to support SAR operations in general with image analysis techniques. Rudol *et.al.*, [11], proposed to assist wilderness SAR operation with videos collected using a UAV with an onboard thermal and color cameras. In their experiment, the thermal image is used to find regions with possible human body and corresponding regions in the color image are further analyzed by an object detector that combines Haar feature extractor with cascade of boosted classifiers. Because of partial occlusion and variable pose of victims, the authors in [12] demonstrated models that decompose complex appearance of humans into multiple parts, [13–15], are more suited than monolithic models to detect victims laying on the ground from aerial images captured by UAV. Furthermore, they have also shown that integrating prior scale information from inertial sensors of the UAV helps to reduce false positives and a better performance can be obtained by combining complementary outputs of multiple detectors.

In recent years, civilian remote sensing applications are greatly benefiting from the development of smaller and cost effective UAVs. Some of the applications include: detecting and counting of cars or other objects from aerial images captured by UAVs [16–18], to assess impact of man-made or natural disaster for humanitarian action, and vegetation mapping and monitoring. In general, they are rapid, efficient, and effective systems to acquire extremely high resolution (EHR) images. Additionally, their portability and easiness to deploy makes them well suited for

applications like post-avalanche SAR operation. According to [19] out of 1886 people by avalanche in Switzerland between 1981 and 1998, 39% of the victims were buried with no visible parts while the rest are partially buried or stayed completely unburied on the surface. Moreover, chance of complete burial can be reduced if avalanche balloons are used. With this statistics, we present a method that utilizes UAVs equipped with vision sensors to scan the avalanche debris and further process the acquired data with image processing techniques to detect avalanche victims and objects related to the victims in near-real time.

Organization of this chapter is as follows: the overall block diagram of the system along with the description of each block is presented in the next section. Datasets used and experimental setup are presented in section 3. Experimental results are presented in section 4 and the last section, section 5, is dedicated to conclusion and further development.

2.2. Methodology

In this section we present a pre-processing method, partially based on image segmentation technique, to filter areas of interest from a video frame followed by an image representation method based on Convolutional Neural Networks (CNNs or ConvNets) and train a Support Vector Machine (SVM) classifier to detect objects. Furthermore, we present a post-processing method based on Hidden Markov Models (HMMs) to take advantage of the correlation between successive video frames to improve decision of the classifier. Block diagram of the overall system is shown in Fig 2.1.

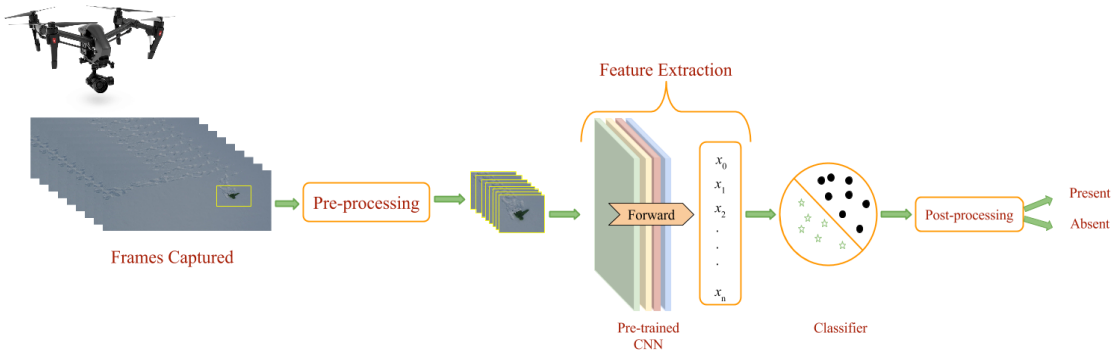


Fig. 2.1. Block diagram of the overall system

2.2.1. Pre-processing

If we consider post-avalanche areas, they are covered by snow and hence mostly white. Assuming objects of interest will have different color than snow, applying image segmentation methods will allow us to separate a frame into regions of snow and other objects. Then, these potential regions of objects are further processed by the next steps. This step allows us to process only regions of a frame and in some cases to skip or filter frames with no potential object regions, thereby providing a better localization of objects and a desirable reduced computation time. In the

pre-processing step, a frame will be scanned with a sliding window and each window will be checked for a color different than snow by thresholding saturation component of the window in the HSV color space. We have adopted the thresholding scheme proposed in [20]:

$$th_{sat}(V) = 1.0 - \frac{0.8V}{255}, \quad (1)$$

where *V* represents the value of the intensity component. We decide that a pixel corresponds to an object if the value of the saturation component *S* is greater or equal than $th_{sat}(V)$. In such a case, the window is said to contain an object.

2.2.2. Feature Extraction

Feature extraction is the process of mapping image pixels or groups of pixels into a suitable feature space. The choice of an appropriate feature extractor strongly affects the performance of the classifier. In the literature, one can find several feature extraction methods proposed for object detection in images or videos. Haar, Scale Invariant Feature Transform (SIFT), and Histogram of Gradients (HOG) are some of the most widely used methods to generate image descriptors. In recent years, the availability of large real world datasets like ImageNet [21] and high performance computing devices have enabled the capability to train deep and improved neural network architectures like ConvNets. These classifiers have significantly improved object detection and classification performances. Beside training CNNs to learn features for a classification task, using pre-trained CNN architectures as a generic feature extractor and training classifiers like SVM has outperformed the performance results obtained by using hand designed feature extractors like SIFT and HOG [22,23].

CNNs are regular feedforward neural networks where each neuron accepts inputs from neurons in the previous layer and perform operations such as multiplication of the input with the network weights and non-linear transformation. Unlike regular neural networks, a neuron in a CNN is only connected to a small number of neurons in the previous layer that are called local receptive fields. Moreover, neurons in a layer are arranged in three dimensions: width, height, and depth. CNNs are primarily designed to encode spatial information available in images and make the network more suited to image focused tasks [24]. Regular neural networks struggle from computational complexity and overfitting with an increase in the size of the input. In contrast, CNNs overcome this problem through weight sharing. Weight sharing is a mechanism by which neurons in a ConvNet are constrained in a depth slice and use the same learned weights and bias in the spatial dimension. These set of learned weights are called filters or kernels. A typical CNN architecture (Fig. 2.2) is a cascade of layers mainly made from three types of layers: the convolutional, pooling, and fully connected layers.

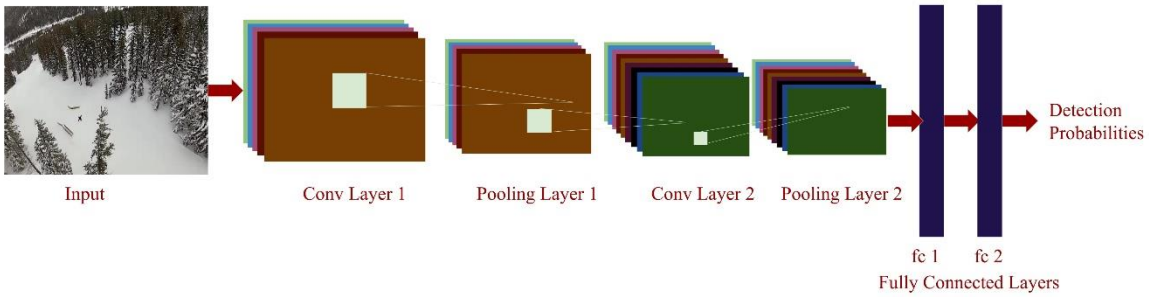


Fig. 2.2. Example of CNN architecture for object recognition

2.2.2.1. Convolutional Layer

The convolutional layer is the main building block of a ConvNet that contains a set of learnable filters. These filters are small spatially (along the height and width dimension) and extend fully in the depth dimension. Through training, the network learns these filters that activate neurons when they see a specific feature at a spatial position of the input. The convolution layer performs a 2-D convolution of the input with a filter and produce a 2-D output called activation map (Fig. 2.3). Several filters can be used in a single convolutional layer and the activation maps of each filter are stacked to form the output of this layer, which is an input to the next layer. The size of the output is controlled by three parameters: depth, stride, and zero padding. The depth parameter controls the number of filters in a convolutional layer. Stride is used to control the extent of overlap between adjacent receptive fields and has impact on the spatial dimension of the output volume. Zero padding is used to specify the number of zeros that need to be padded on the border of the input, which allows to preserve input spatial dimension at the output. Although there are other types of non-linear activation functions, such as the sigmoid and tanh , the most commonly used activation function in ConvNets is the rectified linear unit (ReLu) [25] that thresholds the input at zero. They are simple to implement and their non-saturating form accelerates the convergence of stochastic gradient descent [26].

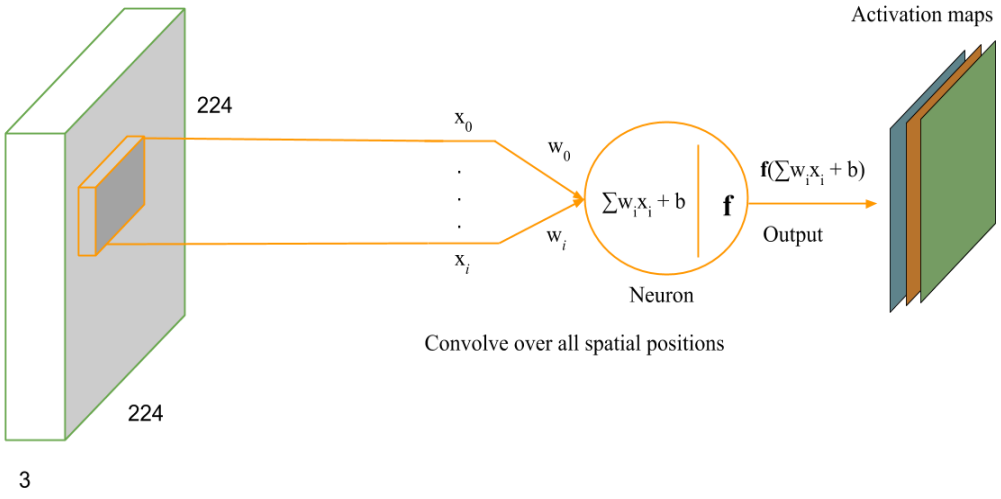


Fig. 2.3. An example of operation performed by the neurons at a spatial location of the input and the resulting activation maps.

2.2.2.2. Pooling Layer

In addition to weight sharing, CNNs use pooling layers to control overfitting. A pooling layer performs down sampling of the input in the spatial dimensions. Similar to convolutional layers, it also has stride and filter size parameters that control the spatial size of the output. Each element in the output activation map corresponds to the aggregate statistics of the input at the corresponding spatial position. In addition to control overfitting, pooling layers help to achieve spatial invariance [27]. The most commonly used pooling operations in CNNs are the max pooling, which computes maximum response of a given patch, the average pooling, which computes average response of a given patch, and the sub sampling (Equation 2) [27]

$$a_j = \tanh(\beta \sum_{N \times N} a_i^{n \times n} + b) \tag{2}$$

which computes the average over a patch of size $n \times n$, multiply it with a trainable parameter β, add a trainable bias b, and applies a non-linear function.

2.2.2.3. Fully Connected Layer

This layer is a regular multi-layer perceptron (MLP), where a neuron is connected to all neurons in the previous layer, that is used for classification. Once the network is setup the weights and biases are learned by using variants of the gradient descent algorithm. The algorithm requires to compute the derivative of a training loss with respect to the network parameters using the backpropagation algorithm. In the context of classification, the cross-entropy loss function is used in combination with the softmax classifier.

Training deep CNN architectures from scratch requires to have sufficient training data, high computing power, and sometimes months of work. Often researches release pre-trained models along with their paper. These models can be adapted to specific tasks either by fine tuning that is using the network parameters as initialization and re-train with the new dataset or as a fixed feature extractor for the recognition task. Which type of transfer learning to use depends on the size of data at hand and its affinity with the original dataset (exploited by the pre-trained model) [28]. In this work, we will make use of the publicly available trained CNN named GoogLeNet. It is trained for image classification task with ImageNet ILSVRC2014 [29] challenge and ranked first. The challenge involved classifying images into one of thousand leaf node categories in the ImageNet hierarchy. ILSVRC dataset contains about 1.2 million images for training, 50,000 for validation, and 100,000 images for testing. The network is 27 layers' deep including the pooling layers. Each convolutional layer contains 64 to 1024 filters of size 1x1 to 7x7 and they use RELU activation function. Max pooling kernels of size 3x3 and an average pooling kernel of size 7x7 are used at different layers of the network. The input layer takes a color image of size 224x224. Beside the classification performance achieved by the network, design of the deep architecture considered the power and memory usage of mobile and embedded platforms so that it could be put to real world use at a reasonable cost. We refer the reader to [30] for detailed description of the model.

2.2.3. Classifier

The next step after feature extraction is to train a classifier suited for the task at hand. The choice of the classifier should take into account dimensionality of the feature space, the number of training samples available and any other requirements of the application. Motivated by their effectiveness in hyperdimensional classification problems, we will adopt the SVM classifier in this work. Introduced by Vapnik and Chervonenkis, SVMs are supervised learning models used to analyze data for classification and regression analysis. The main objective of such models is to find an optimal hyperplane or set of hyperplanes (in multiclass object discrimination problems) that separates a given dataset. They have been applied to a wide range of classification and regression tasks, [31–33].

Consider a binary classification problem with N training samples in a d-dimensional feature space $x_i \epsilon\, \Re^d$ $(i = 1, 2, 3, \dots, N)$with corresponding labels $y_i \epsilon \{-1, +1\}$. There is an optimal hyperplane defined by a vector $\underline{w} \epsilon \Re^d$ normal to the plane and a bias $b \epsilon \Re$ that minimizes the cost function [34] given by:

$$\psi(\underline{w}, \xi) = \frac{1}{2}\|\underline{w}\|^2 + C\sum_{i=1}^{N}\xi_i \tag{3}$$

subject to the following constraints:

$$\begin{cases} y_i(\underline{w} \cdot \phi(x_i) + b) \geq 1 - \xi_i, & i = 1,2,3, \dots, N \\ \xi_i \geq 0, & i = 1,2,3, \dots, N \end{cases} \tag{4}$$

The cost function in equation 3 combines both margin maximization (separation between the two classes) and error minimization (penalizing wrongly classified samples) in order to account for non separability in real data. The slack variables (ξ_i's) are used to take into account non separable data while C is a regularization parameter that allows to control the penalty assigned to errors. Though initially designed for linearly separable data, SVMs were later extended to nonlinear patterns by using kernel tricks. A kernel function aims at transforming the original data into a new higher dimensional space using kernel functions ($\phi(.)$'s) and classification (or regression) is performed in the transformed space. Membership decision is made based on the sign of a discriminant function $f(x)$ associated with the hyperplane. Mathematically,

$$\hat{y} = sign\{f(x)\}, \text{ where} \tag{5}$$
$$f(x) = \underline{w} \cdot \phi(x) + b,$$

2.2.4. Post-Processing

In a video sequence, it can be reasonably expected that the change in content of successive frames is small. Therefore, it is highly likely for an object to appear in consecutive frames. With this in mind, we propose to resort to hidden markov models to improve decision of the classifier for a frame at time t based on the previous frame decisions. HMMs are statistical Markov models useful to characterize systems where unobserved internal state governs the external observations we make. They have been applied to a wide range of applications like human activity recognition from sequential images, bioinformatics, speech recognition, computational and molecular biology, etc., [35,36].

Consider a system with N distinct states, $S = \{s_1, s_2, s_3, \dots, s_N\}$, and M distinct observation symbols, $V = \{v_1, v_2, v_3, \dots, v_M\}$, per state. Given the following parameters,

1. State transition matrix, $A = [a_{ij}]$: probability that the system will be in state s_j at time t given the previous state is s_i.

$$a_{ij} = \Pr\left(q_t = s_j \middle| q_{t-1} = s_i\right), \qquad 1 \le i, j \le N \tag{6}$$

   where q_t is the state at time t.
2. Initial state probability, π: state of the system at time $t = 0$

$$\pi = \Pr(q_0 = s_i) \tag{7}$$

3. Observation symbol probability distribution in state s_j, $B = [b_j(k)]$

$$b_j(k) = Pr\left(x_t = v_k \middle| q_t = s_j\right), \qquad 1 \le j \le N \quad 1 \le k \le M \tag{8}$$

   where x_t is the observation at time t

and given also the two main HMM assumptions, i.e., first order Markov assumption (a state at time t only depends on a state at time $t-1$) and the independence assumption (output observation at time t is only dependent on a state at time t), there are three basic problems that need to be solved in the development of a HMM methodology. These are:

1) Evaluation problem: the objective of this problem is to calculate the probability of an observation sequence, $O = o_1, o_2, \dots, o_T$, given model parameters $\lambda = (A, B, \pi)$, i.e. $P(O|\lambda)$. Besides, it can also be viewed as a way of evaluating how the model can predict the given observation sequence.
2) Decoding problem: it deals with finding the optimal state sequence, $S = s_1, s_2, \dots, s_T$, that best explains a given observation sequence, $O = o_1, o_2, \dots, o_T$, given model parameters λ.
3) Learning problem: it consists in estimating model parameters, $\lambda = (A, B, \pi)$, from a given training data (supervised or unsupervised) to maximize $P(O|\lambda)$.

For our detection problem, we have two hidden states, $S = \{s_1, s_2\}$, namely the presence and absence of an object in a frame (see Table 2.1). The observation variables, x, are image descriptors and our objective will be to maximize the instantaneous posteriori probability (the probability that maximizes the decision of a frame at time t given all the previous observations). Mathematically,

$$q_t^* = \underset{1 \le i \le 2}{argmax}\, P(q_t = s_i | o_1, o_2, \dots, o_t, \lambda) \tag{9}$$

Table 2.1. HMM notations in accordance to our detection problem

s_1	$y = '-1'$
s_2	$y = '+1'$
o_t	x_t (image aquired at time *t*)
y_t	$\hat{y}$ (equation 4)

The state diagram is shown in Fig. 2.4. There exists an efficient dynamic programming algorithm called the forward algorithm, [36], to compute the probabilities. The algorithm consists of the following two steps:

1. *Prediction step*: predict the current state given all the previous observations

$$P(q_t | x_{t-1}, x_{t-2}, \dots, x_1) = \sum_{s_{t-1}} P(q_t | q_{t-1})\, P(q_{t-1} | x_{t-1}, x_{t-2}, \dots, x_1) \tag{10}$$

2. *Update step*: update the prediction based on the current observation

$$P(q_t | x_t, x_{t-1}, \dots, x_1) = \frac{P(x_t | q_t) P(q_t | x_{t-1}, x_{t-2}, \dots, x_1)}{\sum_{x_t} P(x_t | q_t)\, P(q_t | x_{t-1}, x_{t-2}, \dots, x_1)} \tag{11}$$

using Bayes probability theorem

$$P(x_t | q_t) = \frac{P(q_t | x_t) P(x_t)}{P(q_t)} \tag{12}$$

substituting equation 12 into 11, we obtain

$$P(q_t | x_t, x_{t-1}, \dots, x_1) = \frac{P(q_t | x_t) P(q_t | x_{t-1}, x_{t-2}, \dots, x_1)}{\sum_{x_t} P(q_t | x_t)\, P(q_t | x_{t-1}, x_{t-2}, \dots, x_1)} \tag{13}$$

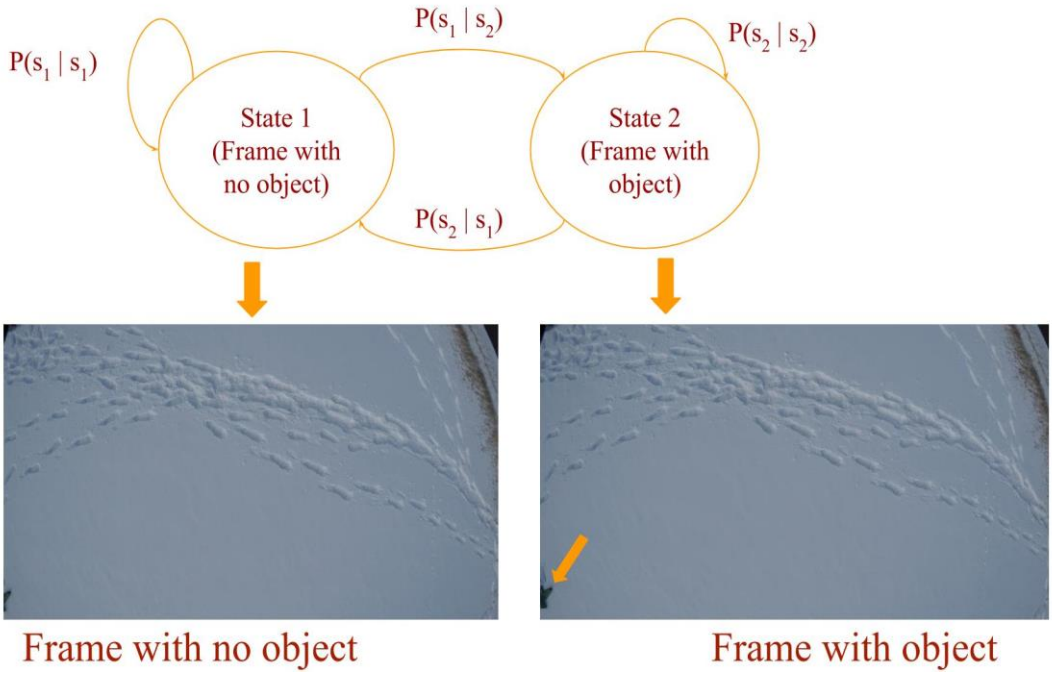


Fig. 2.4. State transition diagram (object pointed by a yellow arrow is a jacket used to simulate top half of buried victim)

The posterior probability, $P(q_t|x_t)$, is obtained by converting SVM classifier decision into a probability using the Platt scaling method, [37]. Platt scaling is a way of transforming outputs of a discriminative classification model (like SVM) into a probability distribution over the classes. Given a discriminant function, $f(x)$, of a classifier, the method works by fitting a logistic regression model to the classifier scores. Mathematically,

$$P(y = 1|x) = \frac{1}{1 + exp(Af(x) + B)} \tag{14}$$

where the parameters A and B are fitted using the maximum likelihood estimation method from a training set by minimizing the cross-entropy error function.

2.3. Data and Experimental Setup

2.3.1. Dataset Description

For this work, we have used two datasets. The first one was compiled by extracting successive frames from different videos of ski areas captured by UAVs freely available on the web. We edited the frames by placing objects of interest like body parts, backpacks, skis, etc. This dataset has a total of 270 frames, of which 165 were used for the training set and the rest for the test set. We have 59 and 52 positive samples in the training and test sets, respectively. Resolution of the images is 1280x720. An example of positive and negative images is shown in Fig. 2.5.

Fig. 2.5. Example of positive (top) and negative (bottom) images from the first dataset. Objects of interest (partially buried skis (left) and top half buried victim (right)) are marked with yellow circle.

The second dataset is recorded on a mountain close to the city of Trento using a GoPro camera mounted on a CyberFed "Pinocchio" hexacopter. It consists of five videos of different durations recorded in 4K resolution (3840x2160) at a rate of 25 frames per second. For convenience, let us call each video as video 1, video 2…, up to video 5. Videos 1, 2, 3, and 4 are recorded at a height in the range of 2 to 10 meters while video 5 is recorded at a relatively higher height, which is between 20 and 40 meters. The first two videos were recorded with the camera at 45° tip angle while the others were captured with the camera pointing strait to the nadir. For this dataset, training set images are extracted from videos 1 and 2 and the rest are used for the test set. Sample frame snapshots are shown in Fig. 2.6.

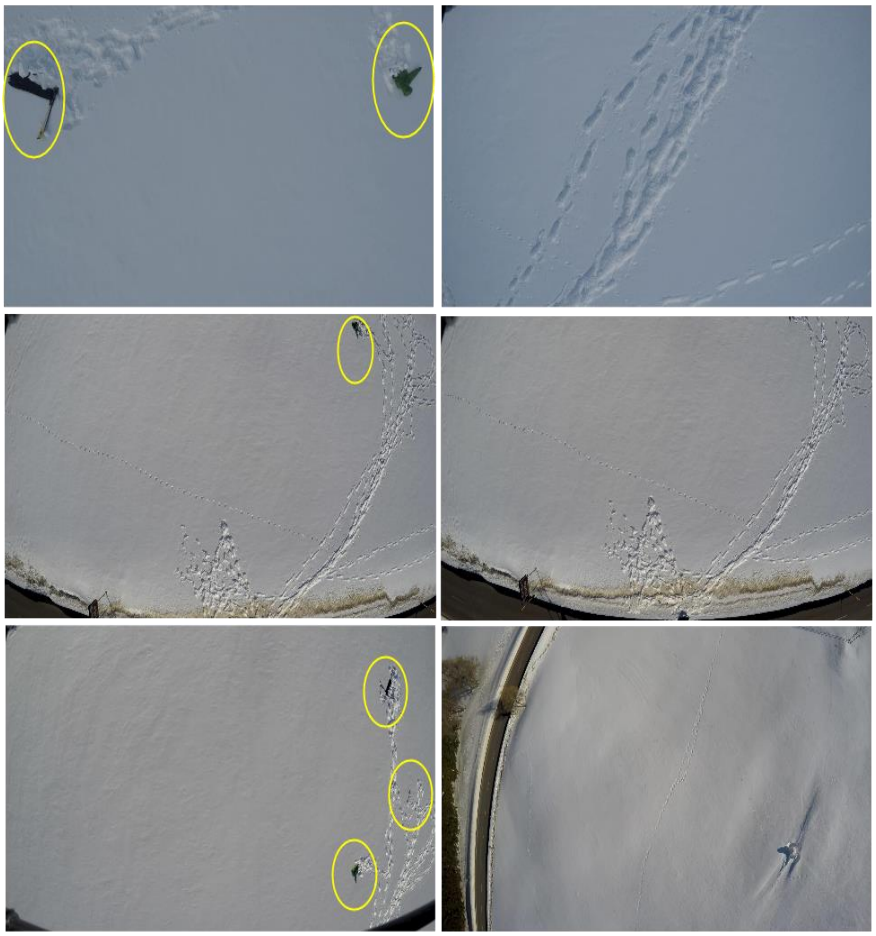


Fig. 2.6. Positive (left) and negative (right) frame snapshots from the second dataset. Objects of interest (skis, jacket to simulate bottom half buried victim, and ski pole) are marked by yellow circle.

2.3.2. Setup

As explained earlier, since our dataset is small and objects of interest are among the thousand classes onto which GoogleNet is trained, we have used the network as a feature extractor. For this purpose, we removed the classification layer (layer 25) of the network. A forward propagation of zero center normalized image of size 224x224 through the network outputs a vector of image descriptor with 1024 elements.

Moreover, since processing time is critical to our problem and data is distributed in a high dimensional space, we train linear SVM for the task of classification. Both training and test features are scaled to have a unit length (equation 14) and the choice of best C (regularization factor) is performed with a grid search of values in the range of 2^{-15} to 2^{5} using two fold cross validation.

$$x' = \frac{x}{||x||} \tag{15}$$

We have used the MatConvNet library [38] to operate on the pre-trained model and LibSVM library [39] to train SVM. All the experiments were conducted on a standard desktop computer with clock speed of 3GHz and 8GB RAM.

2.4. Results and Discussions

In this section, we report experimental results obtained for both datasets. General information about all experiments conducted can be found in Table 2.2. Accuracy, probability of true positives (P_{TP}), and probability of false alarm (P_{FA}) are the performance metrics used. P_{TP} and P_{FA} are calculated as follows:

$$P_{TP} = \frac{\sum \#\ of\ positive\ samples\ correctly\ classified}{\sum \#\ of\ positive\ samples} \tag{16}$$

$$P_{FA} = \frac{\sum \#\ of\ negative\ samples\ classified\ as\ positive}{\sum \#\ of\ negative\ samples} \tag{17}$$

Table 2.2. General description of experiments conducted

		Original image resolution	Resized to	Pre-processing
First dataset	**Experiment 1**	1280 × 720	224 × 224	No
	Experiment 2	1280 × 720	672 × 448, then 6 tiles (224 × 224)	No
	Experiment 3	1280 × 720	1120 × 672, then 15 tiles (224 × 224)	No
Second dataset	**Experiment 4**	3840 × 2160	224 × 224	No
	Experiment 5	3840 × 2160	640 × 480	Yes
	Experiment 6	3840 × 2160	1280 × 720	Yes
	Experiment 7	3840 × 2160	1920 × 1080	Yes
	Experiment 8	3840 × 2160	No resizing	Yes

2.4.1. Experiments without pre-processing

For the first dataset, we conducted three separate experiments at different resolutions. The first experiment is conducted by resizing both training and test frames to an input size, 224×224, of the pre-trained model and extracting the features. In the second experiment, each frame is divided into six tiles each of 224x224 size after resizing to 672x448 (close to VGA). While in the third experiment, fifteen tiles of size 224x224 are generated from each frame after resizing to 1120x672 (close to the original resolution). The results are reported in Table 2.3.

Table 2.3. Classification results for the first dataset (Experiments 1 - 3).

	Accuracy (%)	P_{TP}	P_{FA}
Experiment 1	65.71	0.8462	0.5283
Experiment 2	94.29	0.6346	0.1095
Experiment 3	97.59	0.8065	0.0152

From Table 2.3, it is clear that the overall accuracy increases and P_{FA} decreases with an increase in resolution. Contrarily, P_{TP} decreases as for the second and third experiments with respect to the first and it increases for the third experiment with respect to the second. We believe that the reason for having a high P_{TP} in the first experiment is because we are considering the whole frame, which contains unwanted objects like poles, trees, lift lines, etc. In the first experiment we have high P_{FA} because the whole frame is resized to 224x224. The resizing makes objects of interest become insignificant with respect to the surrounding and thus forces the classifier to learn not only objects of interest but also the surrounding. On the other hand, second and third experiments have small P_{FA} and increased P_{TP} due to tiling, which makes objects of interest in a tile to become more significant with respect to the surrounding and the classifier is able to better discriminate objects of interest from the background. Some qualitative results are shown in Fig. 2.7.

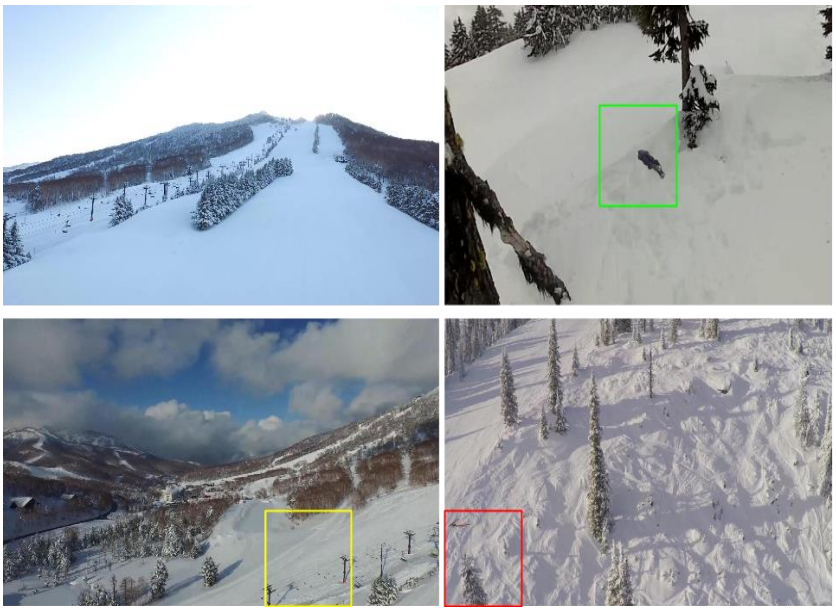


Fig. 2.7. Example of correctly classified negative (top left) and positive (top right), false positive object marked in yellow rectangle (bottom left), and false negative object marked by red rectangle (bottom right).

For the second dataset, the first experiment (Experiment 4 in Table 2.2) we conducted is by down sampling each frame to a size of 224x224. For this experiment, the training set is made up of 4000 frames, of which 2000 are positive samples, extracted from the first two videos. From the results in Table 2.4, video 3 has high accuracy and very low P_{FA} as compared to the other test videos. This is mainly due to the nature of the video. Almost all frames are either snow (white) or objects of interest on top of snow. So, down sampling the frames will not affect visibility of objects of interest. On the other hand, frames from videos 4 and 5 contain background objects like cars,

trees, etc. Additionally, video 5 is recorded at a higher height. For the reasons mentioned above, down sampling a frame to 224x224 results in higher insignificance of objects of interest with respect to the background and hence a high P_{FA}.

Table 2.4. Classification results for the second dataset at resolution of 224×224 (Experiment 4).

	Accuracy (%)	P_{TP}	P_{FA}
Video 3	84.34	0.8386	0.1470
Video 4	36.25	0.9405	0.7445
Video 5	44.13	0.4311	0.5472

2.4.2. Experiments with pre-processing

Next, we conducted four separate experiments at resolutions of 640x480, 1280x720, 1920x1080, and 3840x2160, respectively. Since the number of frames in this dataset is large, tiling each frame and labeling each tile is time consuming. Alternatively, we composed a training set with 3000, of which 1500 are positive, image crops of size 224x224 from the first two videos at the original resolution and trained a linear SVM. During the test phase, each frame is scanned with a sliding window of size 80x80 and if a window passes the threshold, a crop of size 224x224 centered on the window is taken for further processing with the next steps. An example of this process is shown in Fig. 2.8.

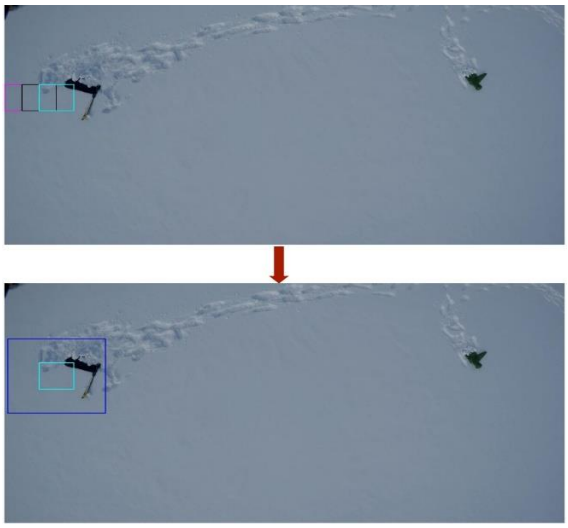


Fig. 2.8. Example showing the pre-processing step. The image on top shows a frame being scanned by a sliding window while the image on the bottom highlights a region (marked by blue rectangle), centered around a window (marked by cyan rectangle) selected for further processing.

As seen from the results in Tables 2.5 and 2.6, for video 3 (experiments 5 to 8), the overall accuracy increases with an increase in resolution as compared to the results obtained in experiment 4. An exception is at the VGA resolution, where there is a decrease in accuracy due to loss of details in down sampling. As expected, the probability of false alarm (P_{FA}) drops significantly with an increase in resolution. On the other hand, P_{TP} has decreased with respect to the result obtained in experiment 4. But it started to increase as resolution is improved, yielding a significant

increase at 4K resolution (experiment 8). We believe that the decrease is due to the difference in the training sets used for experiment 4 and experiments 5 to 8 while the increase is due to a more detailed information available with an increase in resolution.

Similarly, for video 4, the overall accuracy improves significantly as compared to the results obtained in experiment 4. But it starts to drop, as compared to the result at VGA resolution (experiment 5), with an increase in resolution. In experiment 4 we have a high P_{FA}, but it decreases significantly as resolution is improved. However, as we go from VGA (experiment 5) to 4K (experiment 8) resolution, there is an increase in P_{FA}. This is because of objects or part of objects in the background that have similarity with objects of interest, thus incurring the classifier in more wrong decisions. Moreover, the increase in P_{FA} has a negative impact on the overall accuracy. Though initially we have a decrease in P_{TP} at the VGA resolution with respect to the result obtained in experiment 4, there is an increase and stability in the rest of the experiments.

Table 2.5. Classification results for the second dataset at 640×480 and 1280×720 resolutions (Experiments 5 and 6)

	Experiment 5			Experiment 6		
	Accuracy (%)	P_{TP}	P_{FA}	Accuracy (%)	P_{TP}	P_{FA}
Video 3	78.95	0.6383	0.0020	88.40	0.8061	0.0080
Video 4	96.93	0.8452	0.0080	93.31	0.9940	0.0078
Video 5	62.72	0.3352	0.0409	67.72	0.4259	0.0373

Table 2.6. Classification results for the second dataset at 1920×1080 and 3840×2160 resolutions (Experiments 7 and 8)

	Experiment 7			Experiment 8		
	Accuracy(%)	P_{TP}	P_{FA}	Accuracy(%)	P_{TP}	P_{FA}
Video 3	90.01	0.8333	0.0080	94	0.9084	0.0164
Video 4	77.32	0.9940	0.2687	70.63	0.9940	0.3480
Video 5	74.34	0.5723	0.0620	78.93	0.7087	0.1191

For video 5, we have a significant increase in the overall accuracy as resolution increases. P_{TP} initially decreases at VGA resolution (experiment 5) with respect to the results obtained in experiment 4, but it starts to increase as resolution increases. Moreover, we have less P_{TP} as compared to other videos because of the height at which the video is captured. Similar to the other videos, P_{FA} drops significantly with an increase in resolution. But there is also a slight increase in experiments 5 to 8 due to similar reasons mentioned for video 4. Some qualitative results are shown in Fig. 2.9 and 2.10.

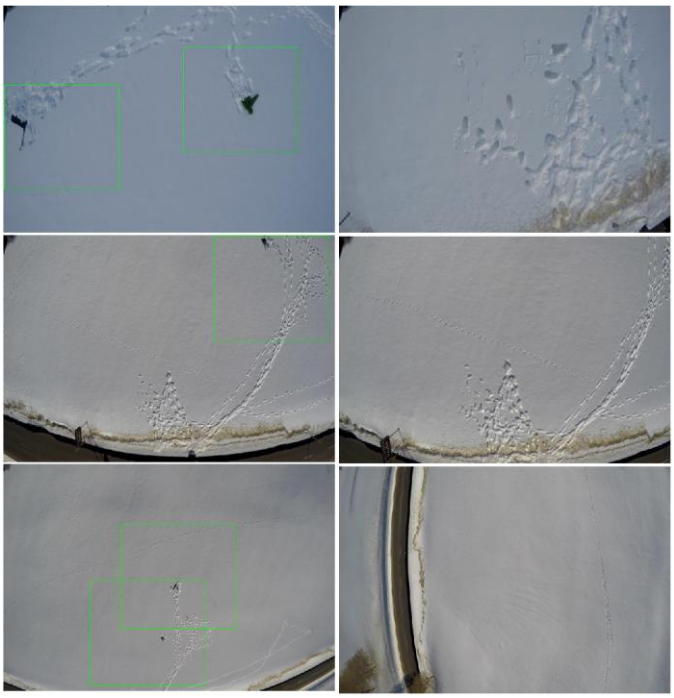


Fig. 2.9. Snapshot of frames with correct positive (left) and negative (right) detection results at the VGA resolution from the second dataset. Regions of a frame containing an object are shown with green rectangle.

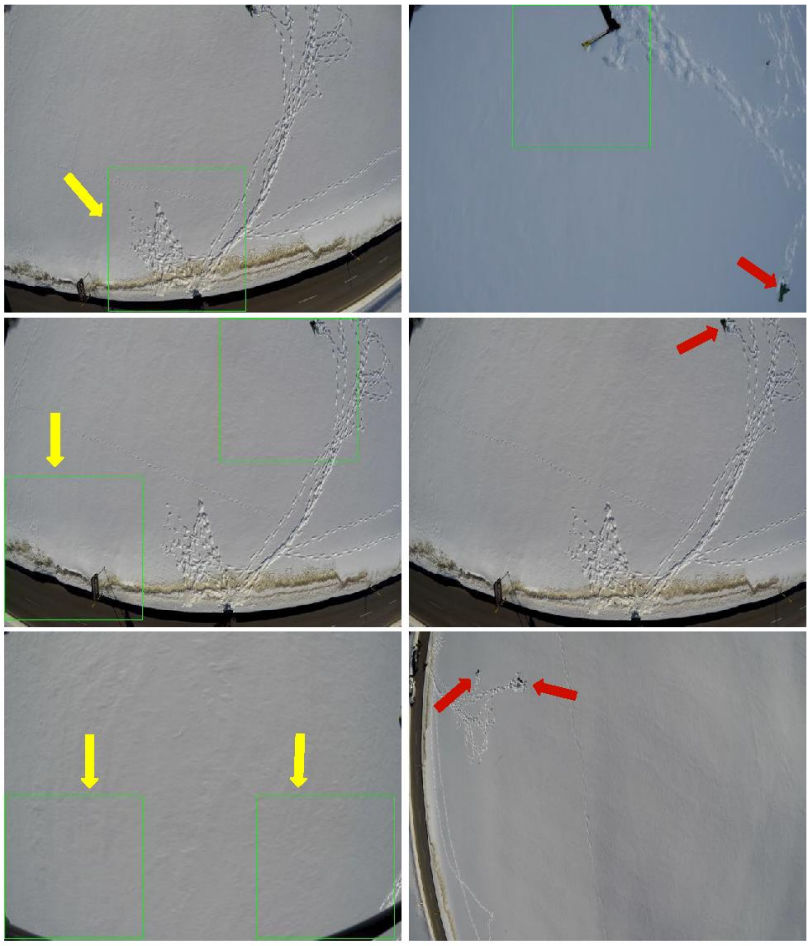


Fig. 2.10. Examples of false positive (left) and false negative (right) frame snapshots at VGA resolution. Yellow arrows indicate false positive regions in a frame whereas red arrows show missed objects in a frame.

2.4.3. Experiments with Markovian post-processing

In the previous experiments, decisions are made separately for each frame. But in a video sequence, there is correlation between successive frames and performance can be further improved by embedding this information in the decision making process. As described in the previous methodological section, we have used HMMs to opportunely exploit this information. Model parameters, prior distribution, transition matrix and observation probability distribution are calculated as follows:

- We have initialized prior distribution in such a way that the probability there is no object in the initial frame is high. For such purpose, we fixed this prior probability value to 0.9.
- The state transition matrix (Table 2.7) is calculated from the available labeled frames.
- Instead of the observation probability distribution, we use the posterior probability by converting SVM discriminant function value into a probability value using the Platt's method and use it in the modified equation of the forward algorithm mentioned in Section 2.2.

The effect of post-processing on the prediction performance can be positive or negative. Indeed, it can correct wrong predictions made by the classifier (positive change) or change the correct prediction made by the classifier into a wrong prediction (negative change). Moreover, these positive or negative changes occur between successive frames where there is transition from one state to the other in the prediction of the classifier. For example, consider two successive frames, at time t and $t-1$. If the decision of the SVM at time t is different than the decision made by HMM for the frame at time $t-1$, because of the small state transition probabilities it is highly likely for the HMM to remain in the same state for the current frame thereby changing decision of the SVM. Depending on the original label of the frame, this change can be either positive or negative. Therefore, prediction performance of the system can either increase if there are more positive changes than negative changes or decrease if there are more negative changes than the positive ones.

The results in Tables 2.8-10 show for video 3 the impact of HMM is not that significant in improving P_{FA}. On the other hand, P_{TP} improves by more than 2% at the VGA resolution. For video 4, since the number of positive frames is very small an increase or decrease in P_{TP} does not affect the overall accuracy. For example, P_{TP} increases by 6% in the first experiment and decreases by approximately 10% at the VGA resolution, but the effect on the overall accuracy is very small. With an increase in resolution P_{FA} gets improved and accuracy increases by more than 5%. Though post-processing has negative effect on the accuracy for video 5, we can see from the results that as resolution increases, P_{FA} drops and, consequently, the difference between the accuracies (achieved with and without post-processing) decreases. In general, it is possible to see that the gain of post-processing depends on the goodness of the classifier. When P_{TP} is high and P_{FA} is low, prediction performance gets improved or remains the same. In all other cases, the impact on prediction performance, especially on the overall accuracy, depends on the ratio of positive and negative frames. Example of positive and negative changes made by HMM are given in Fig. 2. 11 and 2.12.

Table 2.7. State transition matrix

	Current frame with no object	Current frame with object
Previous frame with no object	0.9977	0.0023
Previous frame with object	0.0023	0.9977

Table 2.8. HMM detection result at resolution of 224×224

	Accuracy (%)	P_{TP}	P_{FA}
Video 3	84.95	0.8450	0.1440
Video 4	36.06	1	0.7580
Video 5	42.64	0.4120	0.5570

Table 2.9. HMM detection results at VGA and 720p resolutions

	640×480			1280×720		
	Accuracy (%)	P_{TP}	P_{FA}	Accuracy (%)	P_{TP}	P_{FA}
Video 3	80.52	0.6642	0.0010	88.70	0.8090	0.0051
Video 4	96	0.7440	0.0010	95.26	0.9880	0.0517
Video 5	59.7	0.2768	0.0340	65.47	0.3712	0.0299

Table 2.10. HMM detection results at 1080p and 4K resolutions

	1920×1080			3840×2160		
	Accuracy (%)	P_{TP}	P_{FA}	Accuracy (%)	P_{TP}	P_{FA}
Video 3	89.39	0.8211	0.0056	93.29	0.8910	0.0091
Video 4	82.89	0.99	0.2033	72.86	0.99	0.3226
Video 5	72.80	0.5178	0.0330	77.45	0.6179	0.0463

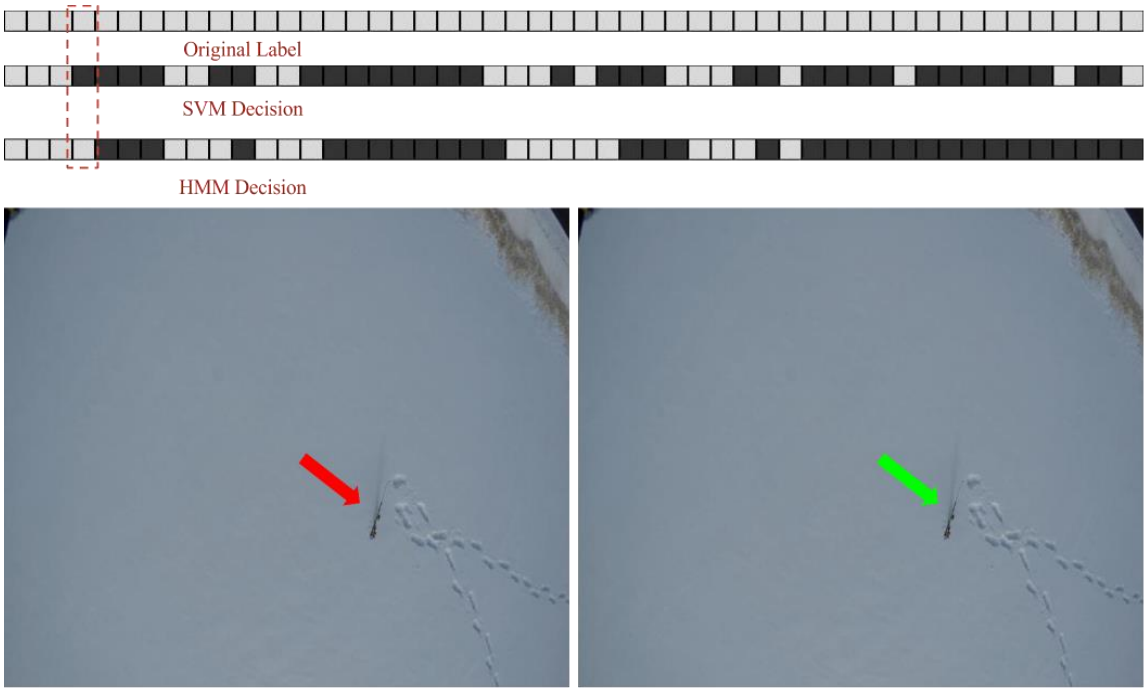


Fig. 2.11. Example of positive change by HMM. Sequence of white and black squares on top indicate label of successive frames. White square indicates a frame has object of interest whereas black square indicates the opposite. The frame where the change happened is highlighted by red dotted rectangle and the corresponding frame in the bottom. The frame for which SVM made wrong decision is shown in bottom left (the object in the frame, skis in this case, is indicated by red arrow) whereas the same frame corrected by HMM is shown in the bottom right (the object in the frame is indicated by green arrow). Note that object is not localized since post-processing decision is made at the frame level.

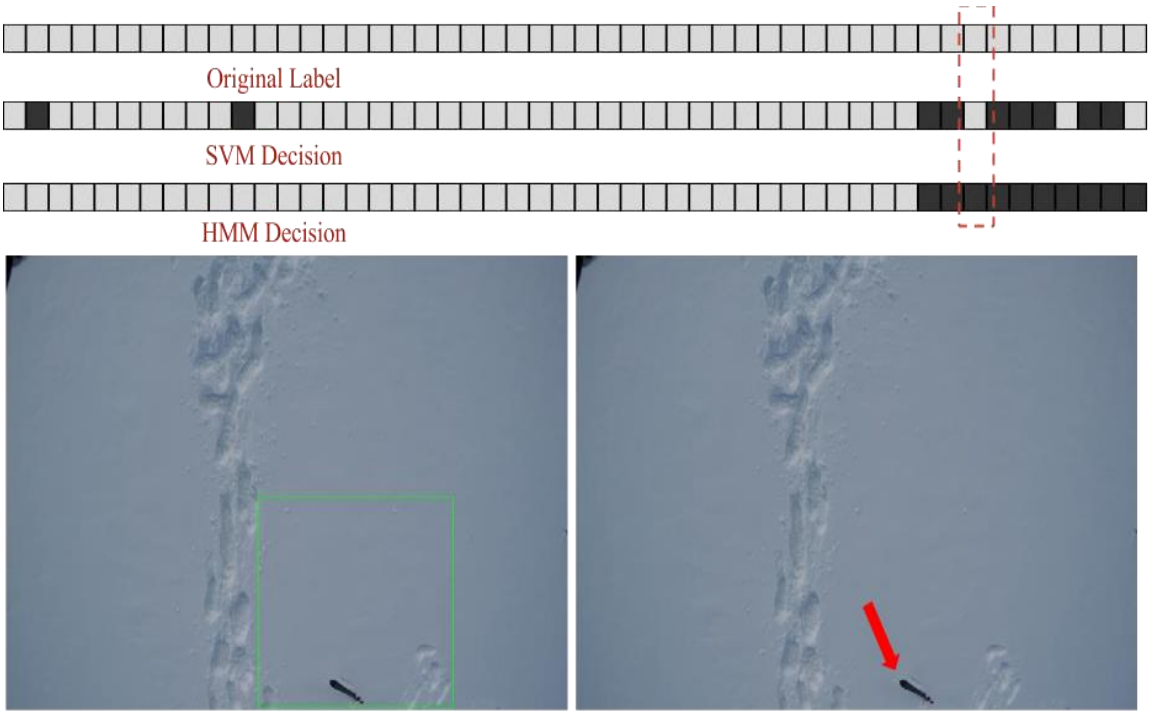


Fig. 2.12. Example of negative change by HMM. Sequence of white and black squares on top indicate label of successive frames. White squares indicate a frame has object of interest whereas black squares indicate the opposite. Frame where the change happened is highlighted by red dotted rectangle. The frame for which SVM made the right decision, with the object localized in a green rectangle, is shown in the bottom left. The same frame for which HMM made wrong decision is shown in bottom right.

2.4.4 Computation time

The processing time required to extract CNN features and perform the prediction for an input image of size 224x224 is 0.185 seconds. For both the first and second datasets, detection at a resolution of 224x224 can be done at a rate of 5.4 frames per second. For the first dataset, since we used tiling to do detection at higher resolutions, the processing time is the product of the number of tiles per frame with the processing time required for a single tile (0.185 seconds). Therefore, at near VGA and full resolutions detection rates are 0.9 and 0.36 frames per second, respectively. For the second dataset, since we have the pre-processing step, we only extract features and perform prediction on frames that pass this step. Additionally, there can be more than one crop of size 224x224 from a single frame. The average processing time is reported in Table 2. XI. The advantage of pre-processing as compared to the tiling approach is twofold. First, it allows to reduce processing time and, second, it provides better localization of objects with in a frame.

In general from the experimental results obtained, it emerges that working at a higher resolution provides a significant improvement on prediction performance at a cost of increased processing time. The bar graph in Fig. 2.13 shows the average accuracy and processing time for the second dataset.

Table 2.11. Detection speed (number of frames per second) for the second dataset

	Video 3	Video 4	Video 5
224 × 224	5.4	5.4	5.4
640 × 480	3.63	1.8	2.88
1280 × 720	2.25	1.15	1.65
1920 × 1080	1.48	0.86	0.98
3840 × 2160	0.41	0.32	0.24

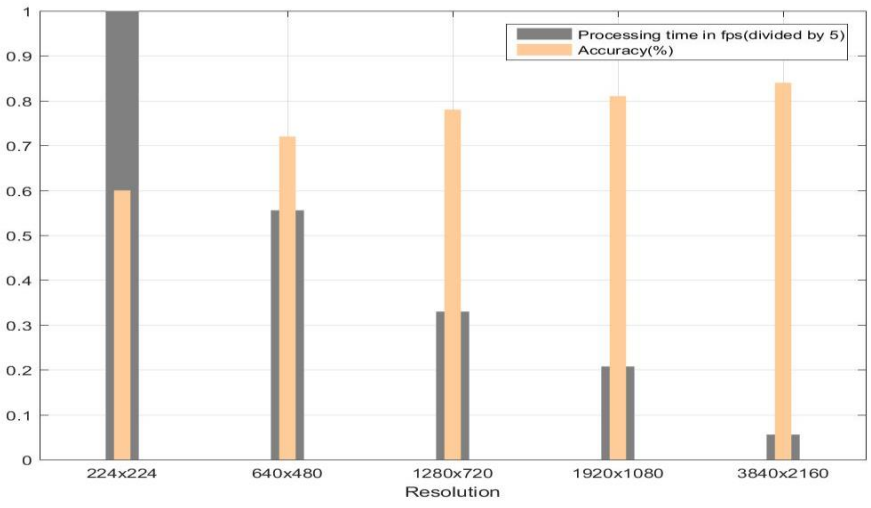


Fig. 2.13 Bar graph showing the change in accuracy and detection rate as resolution increases

2.4.5 Conclusion

In this chapter, we have presented a method to support avalanche SAR operation using UAVs equipped with vision cameras. The UAVs are used to acquire EHR images of an avalanche debris and the acquired image is processed by a system composed of a pre-processing method to select regions of interest within the image, a pre-trained CNN to extract suitable image descriptors, a trained linear SVM classifier for object detection and a post-processing method based on HMM to further improve detection results of the classifier.

From the experimental results, it is clear that improved resolution results in an increase in prediction performance. This is mainly due to the availability of a more detailed information at a higher resolution which enables the decision system to better discriminate objects of interest from the background. Contrarily, we have also seen an increase in false alarm because of background objects or part of objects that exhibit similarity with the objects of interest. Though the computation time increases with an increase in resolution, it is possible to assert that, except at full resolution, the processing time is acceptable for such kind of applications. Additionally, as seen from experimental results of video 5, the height at which frames are acquired is also an important factor which impacts on the prediction performance and the results obtained with the other test videos suggest that scanning the debris at a lower altitude is preferable for a better detection performance. Finally, the choice of resolution to perform detection should be done according to a tradeoff between accuracy and processing time.

Operational scenarios of the proposed method are two. In the first one, the data are transmitted in real time to the ground station where the processing is performed in order to alert the operator when objects of interest are detected while the UAV (or a swarm of UAVs) performs the scans of the avalanche areas. In this scenario, problems of communication links between the drone and the ground station need to be beforehand resolved. In the second scenario, the processing is performed onboard the UAV. This allows to reduce considerably the amount of information to be sent toward the ground station, which in this case can be reduced to simple flag information whenever a frame containing objects of interest is detected. The drawback is the processing capabilities which are reduced with respect to those of a ground station. Work is in progress for an onboard implementation. Moreover, though we have used videos captured at 45° in our experiments, we expect the acquisition to be performed at nadir.

Acknowledgment:

This chapter was carried out within the framework of a project entitled ‘Drones for Finding Avalanche-Buried’ funded by the University of Trento, Italy.

2.5 REFERENCES

[1] Society, N. G. Avalanche Facts, Avalanche Information, Avalanche Videos, Avalanche Photos - National Geographic http://environment.nationalgeographic.com/environment/natural-disasters/avalanche-profile/ (accessed Dec 26, 2016).

[2] SLF, W. Long-term statistics http://www.slf.ch/praevention/lawinenunfaelle/lawinenstatistik/-index_EN (accessed Nov 3, 2016).

[3] Brugger, H.; Etter, H. J.; Zweifel, B.; Mair, P.; Hohlrieder, M.; Ellerton, J.; Elsensohn, F.; Boyd, J.; Sumann, G.; Falk, M. The impact of avalanche rescue devices on survival. *Resuscitation* 2007, *75*, 476–483.

[4] Drones can save lives in search and rescue mission after avalanche http://www.aerialtronics.com/2015/03/drones-can-save-lives-in-search-and-rescue-mission-after-avalanche/ (accessed Oct 26, 2016).

[5] European Satellite Navigation Competition http://www.esnc.info/index.php?anzeige=switzerland10.html (accessed Oct 26, 2016).

[6] First onsite test for the UNIBO drone supporting search and rescue in avalanches http://www.unibo.it/en/research/first-onsite-test-for-the-unibo-drone-supporting-search-and-rescue-in-avalanches (accessed Oct 26, 2016).

[7] Fruehauf, F.; Heilig, A.; Schneebeli, M.; Fellin, W.; Scherzer, O. Experiments and Algorithms to Detect Snow Avalanche Victims Using Airborne Ground-Penetrating Radar. *IEEE Trans. Geosci. Remote Sens.* 2009, *47*, 2240–2251.

[8] Stepanek, J.; Claypool, D. W. GPS signal reception under snow cover: A pilot study establishing the potential usefulness of GPS in avalanche search and rescue operations. *Wilderness Environ. Med.* 1997, *8*, 101–104.

[9] Schleppe, J. B.; Lachapelle, G. GPS tracking performance under avalanche deposited snow. In *Proceedings of the institute of navigation GNSS2006 conference, Fort Worth, Texas, September*; Citeseer, 2006.

[10] Wolfe, V.; Frobe, W.; Shrinivasan, V.; Hsieh, T. Y. Detecting and locating cell phone signals from avalanche victims using unmanned aerial vehicles. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*; 2015; pp. 704–713.

[11] Rudol, P.; Doherty, P. Human Body Detection and Geolocalization for UAV Search and Rescue Missions Using Color and Thermal Imagery. In *2008 IEEE Aerospace Conference*; 2008; pp. 1–8.

[12] Andriluka, M.; Schnitzspan, P.; Meyer, J.; Kohlbrecher, S.; Petersen, K.; Stryk, O. von; Roth, S.; Schiele, B. Vision based victim detection from unmanned aerial vehicles. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*; 2010; pp. 1740–1747.

[13] Felzenszwalb, P.; McAllester, D.; Ramanan, D. A discriminatively trained, multiscale, deformable part model. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*; IEEE, 2008; pp. 1–8.

[14] Andriluka, M.; Roth, S.; Schiele, B. Pictorial structures revisited: People detection and articulated pose estimation. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*; IEEE, 2009; pp. 1014–1021.

[15] Bourdev, L.; Malik, J. Poselets: Body part detectors trained using 3d human pose annotations. In *2009 IEEE 12th International Conference on Computer Vision*; IEEE, 2009; pp. 1365–1372.

[16] Moranduzzo, T.; Melgani, F.; Bazi, Y.; Alajlan, N. A fast object detector based on high-order gradients and Gaussian process regression for UAV images. *Int. J. Remote Sens.* 2015, *36*, 2713–2733.

[17] Moranduzzo, T.; Melgani, F. Automatic Car Counting Method for Unmanned Aerial Vehicle Images. *IEEE Trans. Geosci. Remote Sens.* 2014, *52*, 1635–1647.

[18] Moranduzzo, T.; Melgani, F. Detecting Cars in UAV Images With a Catalog-Based Approach. *IEEE Trans. Geosci. Remote Sens.* 2014, *52*, 6356–6367.

[19] Brugger, H.; Falk, M. Analysis of Avalanche Safety Equipment for Backcountry Skiers.

[20] Sural, S.; Qian, G.; Pramanik, S. Segmentation and histogram generation using the HSV color space for image retrieval. In *Image Processing. 2002. Proceedings. 2002 International Conference on*; IEEE, 2002; Vol. 2, p. II–589.

[21] Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*; IEEE, 2009; pp. 248–255.

[22] Sharif Razavian, A.; Azizpour, H.; Sullivan, J.; Carlsson, S. CNN features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*; 2014; pp. 806–813.

[23] Donahue, J.; Jia, Y.; Vinyals, O.; Hoffman, J.; Zhang, N.; Tzeng, E.; Darrell, T. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In *ICML*; 2014; pp. 647–655.

[24] O'Shea, K.; Nash, R. An Introduction to Convolutional Neural Networks. *ArXiv Prepr. ArXiv151108458* 2015.

[25] Nair, V.; Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*; 2010; pp. 807–814.

[26] Krizhevsky, A.; Sutskever, I.; Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*; 2012; pp. 1097–1105.

[27] Scherer, D.; Müller, A.; Behnke, S. Evaluation of pooling operations in convolutional architectures for object recognition. In *International Conference on Artificial Neural Networks*; Springer, 2010; pp. 92–101.

[28] CS231n Convolutional Neural Networks for Visual Recognition http://cs231n.github.io/transfer-learning/ (accessed Dec 27, 2016).

[29] CS231n Convolutional Neural Networks for Visual Recognition http://cs231n.github.io/transfer-learning/ (accessed Dec 27, 2016).

[30] Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; 2015; pp. 1–9.

[31] Douha, L.; Benoudjit, N.; Douak, F.; Melgani, F. Support Vector Regression in Spectrophotometry: An Experimental Study. *Crit. Rev. Anal. Chem.* 2012, *42*, 214–219.

[32] Pasolli, E.; Melgani, F.; Tuia, D.; Pacifici, F.; Emery, W. J. SVM Active Learning Approach for Image Classification Using Spatial Information. *IEEE Trans. Geosci. Remote Sens.* 2014, *52*, 2217–2233.

[33] Segata, N.; Pasolli, E.; Melgani, F.; Blanzieri, E. Local SVM approaches for fast and accurate classification of remote-sensing images. *Int. J. Remote Sens.* 2012, *33*, 6186–6201.

[34] Vapnik, V. N. *Statistical Learning Theory*; 1 edition.; Wiley-Interscience: New York, 1998.

[35] Kim, E.; Helal, S.; Cook, D. Human Activity Recognition and Pattern Discovery. *IEEE Pervasive Comput.* 2010, *9*, 48–53.

[36] Rabiner, L. R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 1989, *77*, 257–286.

[37] Platt, J.; others Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Adv. Large Margin Classif.* 1999, *10*, 61–74.

[38] Vedaldi, A.; Lenc, K. MatConvNet: Convolutional Neural Networks for MATLAB. In *Proceedings of the 23rd ACM International Conference on Multimedia*; MM '15; ACM: New York, NY, USA, 2015; pp. 689–692.

[39] Chang, C.-C.; Lin, C.-J. LIBSVM: A Library for Support Vector Machines. *ACM Trans Intell Syst Technol* 2011, *2*, 27:1–27:27.

Chapter III

Multilabel Deep Learning Strategies for Imagery Description

Abstract *– This chapter faces the problem of multilabeling unmanned aerial vehicle (UAV) imagery, typically characterized by a high level of information content, by proposing two novel methods based on two deep learning networks, namely, convolutional neural network (CNN) and Autoencoder (AE) network. They are exploited as means to yield a powerful description of the query image. The two proposed methods start by subdividing a given query image into a set of equal tiles, which are successively processed and analyzed separately. Precisely, each tile is described by extracting opportune features that are exploited to perform the multilabel classification task in order to derive the list of objects present in it. In particular, the first method performs the tiles feature extraction process using a pre-trained CNN. Then, it classifies the resulting features by means of a Radial Basis Function Neural Network (RBFNN). Differently, the second method carries out the tiles feature extraction by taking advantage from using three different handcrafted feature descriptors namely, the bag of visual words (BOW), the wavelets transform, and the Histogram of oriented Gradients (HoG). Followed by a further transformation through a learning step of an autoencoder (AE) model. This last provides new features of reduced dimensionality, exploited to feed a multilayer perceptron (MLP) classifier. Furthermore, for both proposed methods, a multilabeling layer composed of customized thresholding operations is integrated on the top of their whole architectures to improve the obtained outcomes. From the conducted experiments on two different EHR UAV image datasets, it comes out that the proposed methods yield very interesting classification accuracies compared to the state-of-the-art.*

3.1. Introduction

The ever increasing interest witnessed in the acquisition and development of Unmanned Aerial Vehicles (UAVs), commonly known as drones in the past few years, has paved the way to a very promising and effective technology. Since 2005, the number of countries that have acquired drones doubled from 40 to more than 75 [1]. UAVs have proven their effectiveness in collecting data over unreachable areas and limited coverage zones due to their small size and fast deployment. Moreover, their custom-made capacity allows them to collect information with a very high level of detail, leading to extremely high resolution (EHR) images. UAVs were mainly created for military usage. However, in the last decade, they have being exploited in numerous civilian applications as well. For instance, in [2], a real time algorithm is introduced for classification, object detection and tracking from thermal UAV images acquired over the surface of the ocean. In [3], the authors present a UAV cloud system disaster surveillance system to reduce natural or man-made damages. In [4], Shaodan *et al.* introduce an unsupervised classification method for UAV images to detect earthquake triggered on rural houses. Furthermore, several works dealing with vehicle detection can be found in [5]-[6]. Authors in [7], present a visual surveillance system for tracking moving objects in video sequences acquired by means of UAVs using Lucas-Kanade optical flow and Continuously Adaptive Mean-Shift (CAMshift) techniques. In [8], a texture-based (i.e., energy, correlation, mean intensity and lacunarity) classification method using Minkovski distance as a method of comparison was presented. In addition, UAVs have been used with promising results in various applications such as in the agricultural sector. In particular, in [9], the authors proposed an automatic method for palm tree detection using Scale-invariant Feature Transform (SIFT) features and extreme learning machine (ELM) classifier. Moreover, Senthilnath *et al.* [10] describe a spectral-spatial method for the detection of tomatoes on UAV

images, exploiting three different spectral clustering methods with spatial segmentation and morphological operations applied on the target image.

In spite of the efforts being dedicated to UAV imagery classification and analysis within the remote sensing community, there is still a plenty of room for improvement. Indeed, as the spatial resolution increases, so does the need for new methods to process images with such high level of detail and rich information content, where traditional ways of classification such as pixel-based and segment-based descriptors may raise the problem of intra-class variability especially when dealing with several classes at the same time. Moreover, they dramatically increase the computational needs. Indeed, this makes the analysis of UAV imagery particularly challenging. In this chapter, we deal with the problem of multilabel classification of EHR images acquired by means of UAVs using a coarse description approach. That is, instead of attributing a label to each individual spatial entity or segment region descriptor as in the traditional monolabel classification, we describe the considered entity by a list of object classes present in it. This approach first subdivides the image into a grid of equal tiles. Then using some specific tile representation and an opportune classification tool, to each tile, a vector of labels is assigned representing the object classes that are possibly present in it. Such a multilabel classification approach was first introduced in [11] for describing UAV images over urban areas with interesting results. In particular, the multilabel implementation derives benefits from exploiting local feature descriptors, such as Scale Invariant Feature Transform (SIFT), and Histogram of Oriented Gradients (HoG), combined with a Bag Of visual Words (BOW) compact representation. Recently, the computer vision community has reported a very promising generation of neural networks architectures, called Deep Neural Networks, They have shown their capability to overcome traditional classification methods in very complex vision tasks [13]-[14]-[15]. In this chapter, we propose two alternative techniques to the new classification problem raised in [11]. The idea behind the development of the first method lies in : i) representing tiles with CNN features; and ii) substituting the matching paradigm adopted in [11] with a multilabel classification model based on a Radial Basis Function Neural Network (RBFNN) .

In the second method, a faster classification paradigm is put forth by using a suitable tile representation, followed by a feature learning step based on an AutoEncoder (AE) network. Once extracted, the produced features are fed to a multilayer perceptron (MLP) network, which acts as classifier for our multilabeling task.

Finally, for both proposed methods, we introduce a multilabeling layer, relying on a set of simple thresholding operations [16] integrated on the top of their architectures to enhance further the obtained results. The remaining part of this chapter recalls the coarse scene description. Then, details the proposed multilabel classification methods. Experiments are conducted on two real UAV image datasets acquired over urban areas to investigate the effectiveness the proposed method, including a comparison with the state-of-the-art.

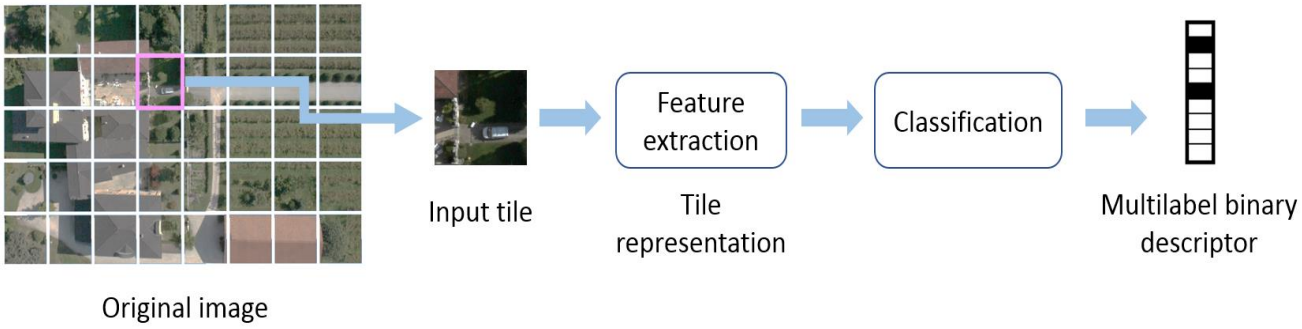


Fig. 3.1. Flow chart of the multilabel coarse classification framework.

3.2. Methodological Overview

3.2.1. Image Coarse Description

Pixel based image analysis methods are very time consuming and have demonstrated a limitation in processing VHR images in complex urban environments compared to object-/ segment- based methods, which allow producing higher classification accuracies, and better thematic maps [17]. However, these descriptor methods (i.e., object-/ segment- based) may raise the problem of intra-class variability when dealing with several classes at the same time especially for extremely high resolution imagery where spatial resolution increases up to 2 cm [11]. Moreover, these methods are not meant to deal with multilabel object classification since they assign to each pixel / segment entity a single label, which increases the number of algorithms (i.e., classifiers) invoked simultaneously for each class. Differently from these techniques, coarse description strategy does not aim to assign to each single pixel or pattern descriptor a single label, but it simply describes a query image or a specific investigated tile within the image by the list of objects present in it. This list indicates the presence/absence state of different objects of interest. Such approach exhibits the advantage of considerably simplifying the complexity of the multilabeling process requirements by jointly handling the co-occurring objects within a unique entity, providing a better perception of the considered multilabeled scene. However, objects normally manifest a sense of semantic dependency. In other words, some objects tend to appear along with other objects, which suggests adopting a classification model that handles such consistency. In greater detail, let us consider a three-channel RGB extremely high resolution (EHR) image (I) acquired by means of UAVs. We start by subdividing it into a grid of tiles of equal sizes. The size of each tile is defined according to the spatial resolution of I and the expected sizes of objects that one aims at recognizing. The multiclass tile-based approach is composed of two main stages: 1) a suitable tile representation strategy; and 2) a tile classification/matching method. A query tile is labeled either with a binary vector of the most similar tile present in the training library using a matching strategy, where the closest tile in the feature space from the training library has likely the same list of objects of the query tile, or as proposed in this chapter it is labeled by means of a classification paradigm. To this point, each tile is "coarsely" described by the subset of classes present in it (Fig. 3.1).

3.2.2. Multilabeling UAV Images with Convolutional Neural Networks

The first proposed strategy starts with the extraction of features using the GoogLeNet pre-trained CNN [13]. CNN is a feedforward hierarchical neural network implementing a set of convolutional and subsampling operations, followed by a softmax classifier. The main underlying idea behind convolutional neural network is to look for a pattern supposed to be invariant to spatial translations. The major difference between convolutional networks and a Multi-Layer Perceptron (MLP) is that, unlike MLP, the internal order of inputs and hidden units is relevant in convolutional networks, and each unit is connected with a specific spatial position of the input image. CNN architecture starts with a convolutional layer that is usually a 3 dimensional volume of neurons. It consists of a set of feature detectors which refer to a convolution with a mask that is consistently convolved across the width and height of the input layer in order to extract robust features against noise and translation. The repetition of this mask all over the input layer allows to share the same weights all over the input layer which reduces the number of free parameters learned. In fact, this simplifies the computational requirements of training the network on large datasets building much efficient and powerful networks. After the convolutional layer, several operations are performed such as the activation function and subsampling, called also pooling. This last forms a nonlinear down-sampling layer that reduces the spatial size, and thus the number of parameters to be computed for the next layer. The most common pooling technique is Max-Pooling. It divides the input into a set of non-overlapping blocks, and assigns to each block its maximum value. In order to increase sparseness, an elementwise activation function Rectified Linear Units (ReLU) layer can be applied after any convolutional layer. The ReLU layer deals also with the vanishing gradient problem in the error backpropagation phase. Thereafter, a set of convolutional and subsampling layers, comes the classification layer. It is a fully connected layer that has full connections to all activation units in the previous layers with a loss function (e.g., softmax).

Training a deep network usually requires a huge number of training images to avoid overfitting. Nevertheless, one may reasonably tackle this issue by transfer learning, which in our case consists of exploiting the weights of a model that is pertained on a large dataset and making use of them while developing our dataset-specific classifier. CNN features computed with pre-trained networks have been used in many computer vision tasks, and have shown good results [18]-[19]. One of the publicly available pre-trained CNN is GoogLeNet. It was first trained over ILSVRC2014 dataset, which contains over 1.2 million images, with a classification challenge of 1000 different classes, thus making GoogLeNet a promising candidate for generating powerful discrimination features. GoogLeNet is a 22-layers deep network excluding pooling layers, with a softmax loss layer as a classifier. The size of its receptive field is 224×224 of three channels (RGB) with zero mean. A rectified linear activation (ReLU) is used in all its convolutional layers. GoogLeNet generates a feature vector of size equal to 1024. The most important component characterizing GoogLeNet network is what is called Inception modules, which are modules with a wise local sparse structure of dense components (e.g., Convolution, Pooling, Softmax). They cluster the correlation statistics of the previous layer output into group of units (Fig. 3.2). The major benefit of this inception layers is their efficient reduction of dimensionality as well as computational requirements. GoogLeNet is based on 9 inception modules. The width of inception modules ranges from 256 filters (in early modules) to 1024 in top inception modules.

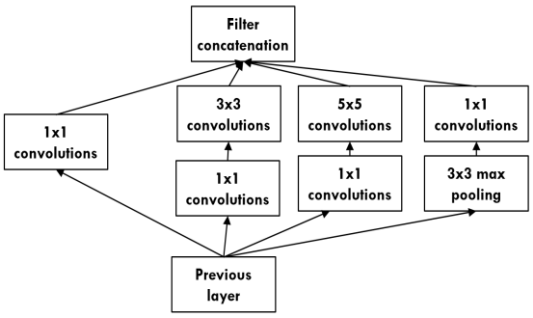


Fig. 3.2. Flow chart of the Inception module.

Since GoogLeNet is not directly adapted to multilabel classification tasks, in this work, we substitute the softmax classifier with a Radial Basis Function neural network (RBFNN), which is a classifier that can fit our multilabeling requirement. Indeed, we will violate the principle that the sum of output targets should be equal to one (i.e., just one of them is active) ruling traditional (mono-label) classification problems. We will look at the outputs of the RBFNN no more as posteriors but as indicators of presence/absence of the corresponding object. This means that during the training phase the classifier will model which objects are present/absent in each training tile. During the prediction phase, the model will provide for each object a quantity (indicator) $f_i(X)$ from which we will need to infer the presence of absence of the considered object. Since during training, the values used to indicate the presence or the absence of an object are set to $f_i(X)$=1 or 0, respectively, during the prediction an intuitive decision mechanism is "the object is present if $f_i(X)\geq 0.5$, otherwise it is absent". We propose to substitute this intuitive decision rule by integrating a multilabeling layer, which will be part of the architecture (Fig. 3.3). In particular, each indicator $f_i(X)$ will be viewed as a feature along which two hypotheses H_0 (absence) and H_1 (presence) are defined. The problem of discrimination between H_0 and H_1 can be seen as a simple thresholding problem. In the literature, there exist several algorithms for computing the best threshold between two classes. In the following, we briefly introduce a simple and fast algorithm, called Otsu's method [16], which will be exploited in this chapter.

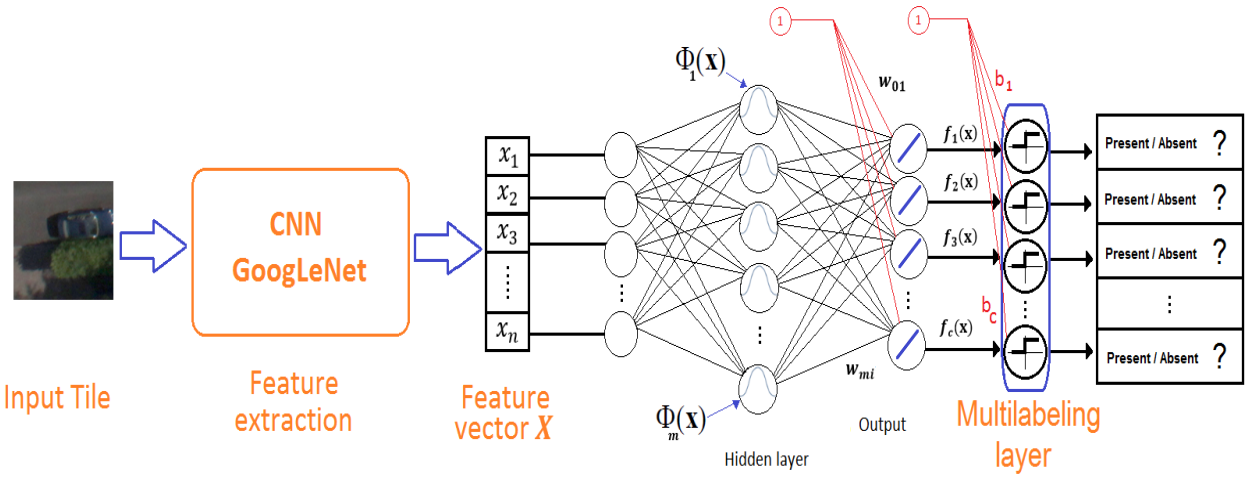


Fig. 3.3. Global flowchart of the proposed classification scheme.

3.2.3. Otsu's Thresholding Algorithm

This algorithm is an unsupervised method, which finds a decision threshold $\widehat{th}$ between two hypothesis (classes) H_0 (absence of object), H_1 (presence of object) based on a discriminant criterion aiming at maximizing the separability between the two classes and thus minimizing their intra-class variance (Fig. 3.4). Let $f_i(X)$ be an output function (feature) of a deep network architecture represented by a 1D histogram composed of M bins.

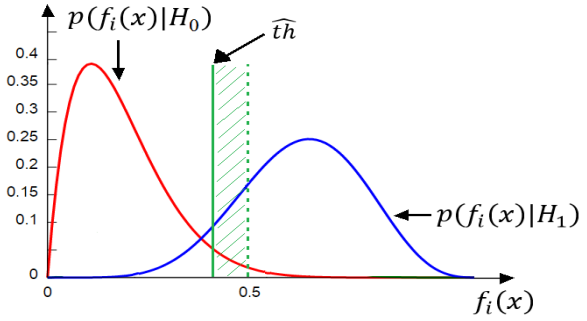


Fig. 3.4. Graphical histogram illustration of the OTSU thresholding technique

This last is transformed by normalization into a probability function $p(f_i(X))$. Let us assume that along $f_i(X)$ just two classes lie, namely H_0 and H_1. We are interested in finding a threshold value t that best separates the two classes. For a given threshold value t, the prior probabilities of H_0 and H_1 can be computed as follows:

$$\mathrm{P}(H_0(t)) = \sum_{i=0}^{t-1} p(i) \qquad (1)$$

$$\mathrm{P}(H_1(t)) = \sum_{i=t}^{M-1} p(i) \qquad (2)$$

The main idea behind Otsu's method is to select the threshold $\widehat{th}$ that minimizes the intra-class variance of the two classes H_0, H_1 which is but the weighted sum of variances of each cluster defined as:

$$\sigma_W^2(t) = \mathrm{P}(H_0(t))\,\sigma_0^2(t) + \mathrm{P}(H_1(t))\sigma_1^2(t) \qquad (3)$$

where $\sigma_1^2(t)$ and $\sigma_0^2(t)$ are the variance the pixels above and below the threshold t (thus an approximation of the variance of the classes H_0 and H_1)., respectively. Alternatively, we may express the minimization process in terms of the between-class variance $\sigma_B^2(t)$, which is defined as the subtraction of the within-class variance $\sigma_W^2(t)$ from the total variance of their combined distribution σ^2 given by:

$$\begin{aligned}\sigma_B^2(t) &= \sigma^2 - \sigma_W^2(t) \\ &= \mathrm{P}(H_0(t))(\mu_0(t) - \mu_T)^2 + \mathrm{P}(H_1(t))(\mu_1(t) - \mu_T)^2 \\ &= \mathrm{P}(H_0(t))\mathrm{P}(H_1(t))\,[\mu_0(t) - \mu_1(t)]^2 \qquad (4)\end{aligned}$$

where the class means are:

$$\mu_0(t) = \sum_{i=0}^{t-1} ip(i)/\mathrm{P}(H_0) \quad (5)$$

$$\mu_1(t) = \sum_{i=t}^{M-1} ip(i)/\mathrm{P}(H_1) \quad (6)$$

and the total mean level is defined as:

$$\mu_T = \sum_{i=0}^{M-1} ip(i) \quad (7)$$

It can easily be verified that:

$$\mathrm{P}(H_0) + \mathrm{P}(H_1) = 1, \qquad \mathrm{P}(H_0)\mu_0 + \mathrm{P}(H_1)\mu_1 = \mu_\mathrm{T} \quad (8)$$

The best threshold $\widehat{th}$ that minimizes the within-class variance σ_W^2 is selected as:

$$\widehat{th} = max_{0 \le t \le M-1} \ \sigma_B^2(t) \quad (9)$$

For every bin t (candidate value for the best threshold) of the histogram, we thus compute the between-classes variance $\sigma_B^2(t)$ and we choose the optimum threshold $\widehat{th}$ that maximizes it. This process is repeated for each output $f_i(X)$ of the RBFN network in order to find the best threshold value for each object and therefore to complete the training of the multilabeling layer. This means that: 1) for each output a histogram needs to be generated; and 2) Otsu's algorithm is applied on each histogram to estimate the best decision threshold for the corresponding class.

3.3. Multilabeling UAV Images with Autoencoders Neural Networks

The main underlying reason for this second approach is to improve the computational performance by proposing a fast and compact process chain that satisfies real-time application standards. Taking into consideration the fact that these algorithms are opted for computational devices mounted on UAVs, which are characterized by their limited processing capacity. The use of deep CNNs has a computational time drawback, due to the depth of their architectures (i.e., numerous computations performed through several layers) resulting in more processing requirements. As described earlier, this proposed tile-based paradigm starts with the extraction of features. Since we are working with extremely high resolution (EHR) imagery, each tile is characterized by a high level of detail and rich information content. In order to extract a compact signature that describes efficiently each tile, we resort to three different tile representation strategies, namely the bag of visual words (BOW), the wavelets transform, and the Histogram of oriented Gradients (HoG) representations. In order to further boost their representation capability, we exploit autoencoder neural networks. Indeed, we feed the extracted features to an autoencoder network, which constructs new learned features from the initially extracted features (i.e., RGB_BW, HoG, wavelets transform) (Fig. 3.5). The next step of our classification framework consists in adding a multilayer perceptron (MLP) network as a classifier, which fits our multilabeling requirement. Indeed, the MLP can handle simultaneously multiple outputs, which may characterize each tile of the image. Each title will be labeled with a binary vector of multiple predefined classes which correspond to the number of MLP outputs. Moreover, we will implement a simple refining mechanism at the output of the MLP so that to further boost the multilabeling

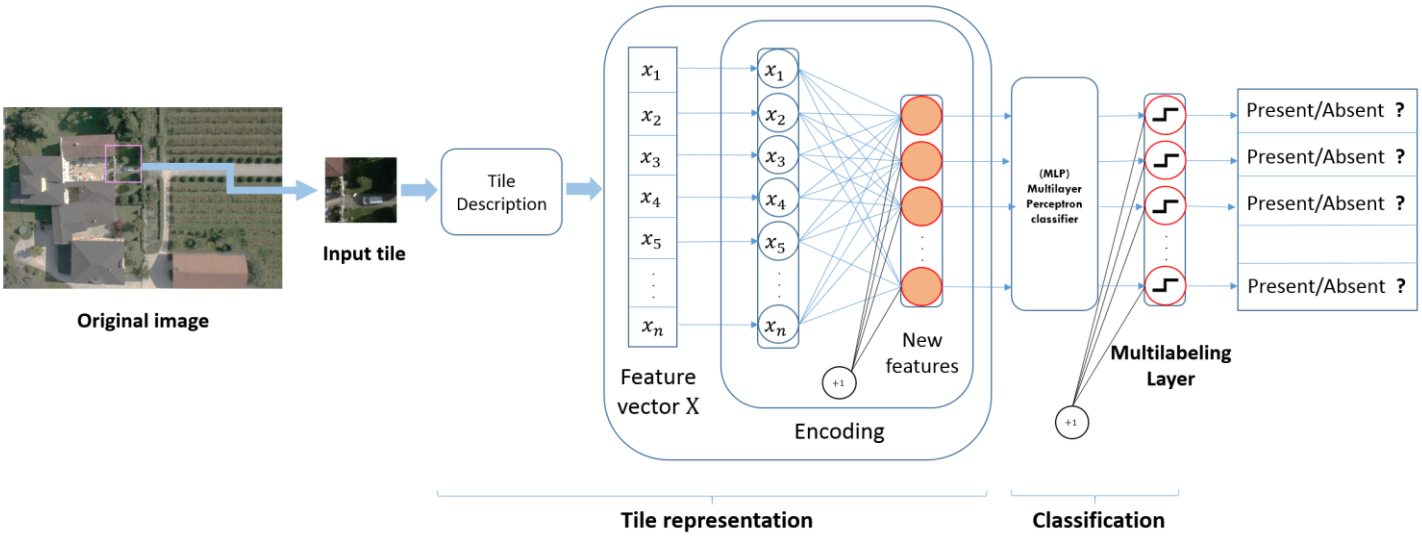


Fig. 3.5. Flow chart of the multilabel classification method based on the autoencoder network.

accuracy. It is based on a thresholding operation applied on the output of the MLP. The value of the threshold is estimated empirically i.e., by maximizing the average accuracy on the validation samples. The tile representation strategies adopted in this chapter are outlined in the following.

3.3.1. Tile Representation strategies

3.3.1.1. Bag of Visual Words

The bag of visual words is a well-known paradigm that has been widely applied in the computer vision field. The main reason behind using BOW model is to reformulate our tile image representation into a compact and finite number of features. The details outlining the BOW model starts from the pixel values (RGB) that represent the spectral features of our image tiles. First, we project all pixels of our training tile images into a 3D space of RGB features, forming a cloud of points. Next, we apply the *K*-means clustering method on the resulting spectral values which produce a set of *C* cluster centroids (*i.e.*, words) forming what is called a codebook. Each learned cluster K_i (i=1, 2,…, C) represents a word of this codebook. Finally, we map the RGB features of all the image tiles into a histogram whose length equals the number of the learned centroids (*i.e.*, *C*) by assigning each pixel of a given tile to the closest centroid (word) of the built codebook. The generated histogram presents a compact signature codifying the number of occurrences of each word in the image tile.

3.3.1.2. Wavelet Transform

In order to overcome the limitation of Fourier transform in extracting and analyzing the features of a signal in both time and frequency domains simultaneously, an alternative tool was introduced, namely the wavelet transform. It highlights the local regions over a range of scales instead of the frequency to capture both time and frequency information of a signal. In particular, in image processing, discrete wavelet transform (DWT) provides a multi-scale analysis transform framework that captures both spatial and frequency features at different resolution levels, where it decomposes the image into an independent approximation and detailed coefficients using low-pass

and high-pass finite impulse response (FIR) filters, respectively [20]. This decomposition process is performed recursively at each resolution level until a certain iteration number is reached. Since the wavelet decomposition is defined by some filter banks, and in consideration of the large number of filters that can be found. A problem to solve is the selection of the filter banks corresponding to a wavelet that can efficiently represent our UAVs tile images. In order to cope with this issue, we will exploit a well-known family of DWT orthogonal wavelets with compact support, namely Daubechies wavelets [21].

3.3.1.3. Histogram of oriented gradients

Histogram of oriented gradients (HoG) is an image descriptor that has gained a sound reputation in the computer vision and remote sensing communities. It extracts structural information (i.e., shape) in a localized section of an image, based on edge directions and the distribution of intensity gradients. This feature descriptor was initially introduced in [22] for pedestrian detection, then it has proven its effectiveness for detecting various complex classes of objects in both static and video images. The HoG features descriptor is obtained by subdividing the query image into a set of overlapping regions called cells. Then for each cell, a histogram of gradient directions is computed. The resulting histograms are then concatenated to form the descriptors of the image. Finally, a set of larger regions than the cells called blocks, are formed out of the resulting normalized histogram descriptors, in order to deal with the invariance to illumination and shadowing changes.

3.3.2. Autoencoder neural network

In recent years, Deep Neural Networks (DNNs) have become a topic of ongoing interest in the machine learning community, and have shown a great potential in many complex computer vision tasks. One of these Deep Neural Networks architectures is the Autoencoder [23]. It consists of a three-layer feedforward neural network (i.e., input, hidden, output) composed of a two-step process (i.e., encoding and decoding) (Fig. 3.6). The Autoencoder can reduce the dimensionality of a set of data by learning nonlinearly transformed features (through the hidden layer) (Fig. 3.6).

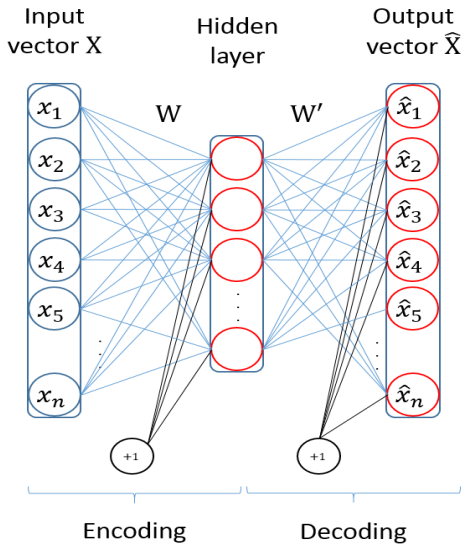


Fig. 3.6. Architecture of an AE network.

Alike the multilayer perceptron, the Autoencoder has the same number of inputs nodes in the output layer, since it aims at reconstructing the input vector at the output by passing it through the hidden layer that captures effective representation features with little redundancy. Usually (but not necessarily) the hidden layer contains fewer nodes than the input layer in order to generate compact and less sparse features.

Let $x \in \mathcal{R}^n$ be an input vector, and W, b the weight matrix and the bias of the input vector, respectively.
The AE learns the nonlinear function h_1 such that:

$$s = h_1(Wx + b)$$

usually, an element-wise activation function (i.e., sigmoid) is applied, which is formulated as:

$$h_1(z) = \frac{1}{1 + e^{-z}}$$

After the encoding phase comes the decoding phase, where one reconstructs the output $\hat{x}$ that has the same dimension of the input vector x, defined as:

$$\hat{x} = h_2(W's + b')$$

A loss function (*e.g.*, squared error) is used to compute the error between the input vector x and its reconstruction $\hat{x}$ as follows:

$$L(x, \hat{x}) = \|x - \hat{x}\|^2$$

$$= \left\|x - h_2(W'\left(h_1(Wx + b)\right) + b'\right\|^2$$

The AE estimates its parameters (*i.e.*, W, W′, b, b′) by minimizing the cost function, namely the squared error between the input vector **x** and its reconstruction $\hat{\mathbf{x}}$:

$$\underset{\mathbf{W},\mathbf{W}',\mathbf{b},\mathbf{b}'}{\operatorname{argmin}} [L(\mathbf{x}, \hat{\mathbf{x}})]$$

In the training phase, the weights and biases are first initialized randomly. In order to compute their best values, the back propagation algorithm is adopted to update them iteratively. Finally, the reconstruction layer is removed including its parameters (W', b') to obtain the optimal values of the construction weights and the biases (*i.e.*,W, b) of the encoding phase which will allow to generate the new reduced feature vector. In a successive step, the new AE-generated features are fed into a classifier to get the final prediction results. It is noteworthy that the AE-generated features can come from any of the previously mentioned type of features (RGB bag of visual words, wavelet, HoG features) or other kinds of descriptors not considered in this chapter (e.g., local binary pattern, Gabor features. etc…). Turning back to the classifier, we will make use of a multilayer perceptron (MLP) with a single hidden layer and sigmoid activation functions. We have chosen the MLP network as a classifier since it fits our multilabel requirements (i.e., need to handle simultaneously multiple outputs). Moreover, due to its relative simple architecture, an MLP with a single hidden layer does not need many parameter tunings. As the AE, the MLP training is based on the back propagation algorithm.

3.4. Experimental Results

In this section, we evaluate the classification performances of the proposed methods on two real datasets of UAV images acquired over two different locations.

3.4.1. Dataset Description and Experimental Setup

In the context of theses proposed frameworks, we used two different datasets of UAV images. The first set of images was taken over the Faculty of Science of the University of Trento, Povo, (Italy) Nadir acquisition on the 3rd October 2011 at 12:00 am. The second set of images was acquired near the city of Civezzano (Italy) at different off-nadir angles, on the 17th October 2012. Both Acquisitions were performed with a picture camera Canon EOS 550D characterized by a CMOS APS-C sensor with 18 megapixels. The UAV images are characterized by three channels (RGB) with a spatial resolution of approximately 2 cm. The image size is 5184×3456 pixels and the radiometric resolution is 8 bits for both datasets. The first dataset is composed of 9 images, subdivided into three groups.

Training set: 2 images are selected as training images. We extracted randomly 1000 tiles of size 224×224 from both images. The two training images were chosen from the overall set in such a way they contain all predefined classes of objects, which are 'Asphalt', 'Grass', 'Tree', 'Vineyard', 'Pedestrian Crossing', 'Person', 'Car', 'Roof 1', 'Roof 2', 'Solar Panel', 'Building Facade', 'Soil' and 'Shadow'.

Validation set: Just one image belongs to this set. It was used to determine the free parameters for both strategies, namely, the RBFNN's number of hidden nodes, centers of activation functions and their desctributions for the first strategy. The best number of epochs for the reconstruction phase (feature learning) in the autoencoder, along with the number of hidden nodes in the MLP classifier for the second strategy. In addition of the best threshold values yielded by the Otsu's method for each of the RBFNN/MLP output classes.

Test set: it is composed of 7 images. We subdivided each test image into a non-overlapping grid of equal tiles of 224×224 pixels as explained in the previous section.

The second dataset is composed by 10 images, subdivided into two groups.

Training set: 3 images are selected as training. We extracted randomly 1000 tiles of size 224×224 from the three images. The three training images were chosen from the overall set in such a way they contain all predefined classes of objects, which are 'Asphalt', 'Grass', 'Tree', 'Vineyard', 'Low Vegetation', 'Car', 'Roof 1', 'Roof 2', 'Roof 3, 'Solar Panel', 'Building Facade', ' Soil', 'Gravel', and 'Rocks'.

Test set: it is composed of 7 images. We subdivided each test image into a non-overlapping grid of equal tiles of 224×224 pixels. For both training sets, we rotated each tile randomly with one of the following four angle values: 0°, 90°, 180°, and 270°. Given the image resolution of 2 cm, the tile size covers 4.5×4.5 meters. Regarding the accuracy evaluation, we adopted the sensitivity and specificity metrics in order to compare our methods with a reference one [11]. This

latter consists in using RGB and HoG features combined with a Bag Of visual Words (BOW) for compact tile representation, and the chi-squared measure distance (χ^2) as a matching strategy (each test tile is labeled with the same binary vector of the most similar tile present in the training tiles library). All the experiments were conducted on an Intel Xeon E3-1246 CPU @ 3.5 GHz with 32 GB RAM using a Matlab platform.

3.4.2. Results and Discussion

3.4.2.1. Results of the Convolutional Neural Networks based Method

As for comparing the proposed first strategy with other state-of-the-art CNN-based models, we investigate also the AlexNet architecture [24] (Caffe version), which is another very popular pre-trained model. It contains five convolutional layers with three fully-connected layers. AlexNet generates a feature vector of size equal to 4096. In order to deal with our multilabeling requirement, we substitute the original multinomial logistic regression (Softmax) in AlexNet and GoogLeNet with a Multilabel Logistic Regression (MLR) prediction model, which consists of as many binary logistic classifiers as the number of labels (classes). In order to enrich further the comparative study, we added two other strategies, which are based on feeding the CNN features (i.e., AlexNet, GoogLeNet) to multiclass (one-against-all) Support Vector Machines (SVMs), implemented with both linear and radial basis function (RBF) kernels. The quantitative comparison results are summarized in Table 3.1. As shown, the combinations of CNN features (i.e., AlexNet, GoogLeNet) with RBFNN classifier achieved in general better results in terms of average accuracy than the combination of the same features with the other three classifiers (i.e., Multilabel Logistic Regression, linear SVM, and SVM with the Gaussian RBF kernel). GoogLeNet-RBFNN and AlexNet-RBFNN score 79.3% and 78.4% of average accuracy respectively for dataset 1, and 77.4% and 76.9% of average accuracy for dataset 2. Furthermore, this combination strategy overcomes the reference method [11] in dataset 1 with an increment of around 3% for GoogLeNet and 2% for AlexNet (in terms of average accuracy), and slightly overcomes it in dataset 2 with an increment of 0.8% and 0.3 % of average accuracy for GoogLeNet

Table 3.1. Comparison of classification accuracies in terms of sensitivity (SENS) and specificity (SPEC) between the different implementations. Computational time per tile is also reported for each strategy.

	Dataset 1(Povo)						Dataset 2(Civezzano)					
	Accuracy(%)			Time(ms)			Accuracy(%)			Time(ms)		
METHOD	Spec	Sens	Average	Tile Representation	Classification / Matching	Total	Spec	Sens	Average	Tile Representation	Classification / Matching	Total
HCR-B [11]	92.2	60.7	76.4	**32**	7	**39**	91.9	61.4	76.6	**32**	7	**39**
GoogLeNet (MLR)	92.7	60.8	76.8	90	**0.3**	90	92.9	58.7	75.8	90	**0.3**	90
AlexNet(MLR)	94.5	60.4	77.5	40	0.8	41	94.0	58.0	76.0	40	0.9	41
GoogLeNet -RBFNN	95.4	63.1	79.3	90	2	92	**96.1**	58.7	77.4	90	2	92
AlexNet -RBFNN	96.2	60.6	78.4	40	5	45	95.4	58.3	76.9	40	5	45
GoogLeNet _SVM(linear)	**96.4**	59.0	77.7	90	41	131	**96.1**	58.1	77.1	90	53	143
GoogLeNet_SVM(RBF)	96.3	58.6	77.5	90	39	129	95.1	53.6	74.4	90	51	141
AlexNet -SVM(linear)	95.5	54.5	75.0	40	131	171	95.2	52.3	73.7	40	143	183
AlexNet -SVM(RBF)	95.7	52.5	74.1	40	124	164	95.2	52.3	73.8	40	144	184
GoogLeNet -RBFNN (ML)	90.3	**75.1**	**82.7**	90	2	92	92.6	**68.6**	**80.6**	90	2	92
AlexNet -RBFNN(ML)	93.0	**70.5**	**81.8**	40	5	45	93.2	**66.1**	**79.7**	40	5	45

Table 3.2. Best threshold values yielded by the Otsu's method for each of the RBFNN output classes.

	Bias values													
class	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}	b_{13}	b_{14}
GoogLeNet (Dataset 1/Povo)	0.38	0.45	0.50	0.26	0.36	0.33	0.42	0.46	0.32	0.34	0.37	0.38	0.38	---
AlexNet (Dataset 1/Povo)	0.41	0.43	0.44	0.20	0.38	0.15	0.46	0.43	0.32	0.36	0.37	0.38	0.38	---
GoogLeNet (Dataset 2/Civezzano)	0.49	0.44	0.39	0.26	0.27	0.43	0.34	0.34	0.38	0.38	0.35	0.36	0.35	0.37
AlexNet (Dataset 2/Civezzano)	0.44	0.47	0.43	0.34	0.36	0.37	0.41	0.43	0.32	0.44	0.45	0.42	0.27	0.41

and AlexNet, respectively. This stresses the promising discriminative capabilities of the deep and hierarchical feature representation process implemented by GoogLeNet and AlexNet.

A further refinement of the results has been possible thanks to the addition of a multilabeling layer of step functions at the top of the RBFNN. As explained earlier, the parameter (i.e., bias) of these functions is estimated by applying the very fast Otsu's algorithm on the output of the training RBFNN. In particular, for each RBFNN $f_i(X)$ output, we constructed a histogram of 30 bins covering the range from -1.5 to +1.5 from the available training tiles. The obtained bias values of the multilabeling layer are reported in Table 3.2, which shows that the values range from 0.14 to 0.49 depending on the class, stressing thus the importance of customizing the threshold value to each kind of object. The final results on the test tiles of datasets 1 and 2 are reported in Table 3.1. As can be seen, a significant boost of accuracy was possible by adding the multilabeling layer on top of the RBFNN outputs. In particular, there is a clear average accuracy improvement that comes at the cost of some loss in the specificity. For instance, for GoogLeNet-RBFNN scheme in dataset 1, despite the decrease in specificity of roughly 5 % from 95.4% to 90.3%, there is a higher gain of almost 12 % in terms of sensitivity from 63.1% to 75.1%. A same improvement can be noticed for all CNN feature combinations with RBFNN in both datasets. The multilabeling layer exhibits the advantage that it reduces the risk of missing objects (estimated threshold values are all less than 0.5) resulting thus in a globally better prediction. The obtained results clearly illustrate the usefulness of exploiting the distributions of the RBFNN outputs from the training tiles for customizing the decision process and therefore improving the results of the multilabel classification task.

In terms of processing time required per single tile (see Table 3.1), the reference method [11] reports to be 6 millisecond faster than the proposed AlexNet-RBFNN architecture (39 against 45 milliseconds, respectively) and shows about 2 times faster than GoogLeNet-RBFNN (39 against 92 milliseconds, respectively). Indeed, CNN networks, in particular GoogLeNet, perform much more computations due to the large number of layers and blocks composing each layer compared to the HCR-B strategy which uses fast local feature descriptors. As expected, AlexNet-RBFNN turned out to be two times faster than GoogLeNet-RBFNN. In fact, GoogLeNet architecture is deeper than that of AlexNet's, resulting in more processing needs.

Table 3.3. Comparison of classification accuracies in terms of sensitivity (SENS) and specificity (SPEC) between the different implementations. Computational time per tile is also reported for each strategy.

	Dataset 1(Povo)						Dataset 2(Civezzano)					
	Accuracy(%)			Time(ms)			Accuracy(%)			Time(ms)		
METHOD	**Spec**	**Sens**	**Average**	**Tile Representation**	**Classification / Matching**	**Total**	**Spec**	**Sens**	**Average**	**Tile Representation**	**Classification / Matching**	**Total**
HoG	93.7	34.4	64.1	**5**	0.1	5	94.3	42.1	68.2	5	0.1	5
WAV	94.7	39.7	67.2	9	0.1	9	94.6	50.9	72.8	9	0.1	9
RGB-BOW	**96.5**	50.1	73.3	25	0.1	25	**96.2**	53.5	74.9	25	0.1	25
HoG-AE	94.2	41.5	67.8	**5**	0.1	5	93.4	45.7	69.6	5	0.1	5
WAV-AE	94.0	44.6	69.3	9	0.1	9	94.8	53.7	74.3	9	0.1	9
RGB-BOW-AE	94.4	56.6	75.5	25	0.1	25	92.8	60.9	76.9	25	0.1	25
HoG-AE(ML)	77.4	76.6	77.0	**5**	**0.1**	**5**	79.9	69.0	74.5	**5**	**0.1**	**5**
WAV-AE(ML)	84.0	**79.9**	**82.0**	9	0.1	9	89.6	65.1	77.4	9	0.1	9
RGB-BOW-AE(ML)	86.0	74.4	80.2	25	0.1	25	84.6	**76.7**	**80.7**	25	0.1	25
HCR-B [11]	92.2	60.7	76.4	32	7	39	91.9	61.4	76.6	32	7	39
GoogLeNet-RBFNN (ML)	90.3	75.1	**82.7**	90	2	92	92.6	68.6	**80.6**	90	2	92
AlexNet -RBFNN(ML)	93.0	70.5	81.8	40	5	45	93.2	66.1	79.7	40	5	45

3.4.2.2. Results of the Autoencoder Neural Networks based Method

Before evaluating the performance of the second proposed classification method, we discuss briefly the set of free parameters characterizing the three feature extractors exploited in this scheme. For the RGB-BW features, we have fixed the number *C* of clustering centroids to 200 in order to adequately represent the large color variations in EHR images. For the HoG features, we have set its parameters for generating the histograms as follows: 9 for the number of bins and 56 as the size of cells, which results in a feature descriptor of size 576. With regard to the wavelet features, we benefit from using the approximation part of the Daubechies wavelet filter coefficients of order 2 as extracted features with four decomposition levels. This results in an approximation image of size 14×14 and 3 channels (RGB). The input image is finally converted into a vector of length 588.

In consideration of the outputs of the MLP classifier which are real values and in order to force each output value into one of the two states of the corresponding object (present/absent), a thresholding step must take place. For this purpose, we investigate the effectiveness of integrating the multilabeling layer based on Otsu's method [16] (as explained earlier) on top of the MLP classifier compared to the standard 0.5 threshold. In Table 3.3, we first report the results of the three different typologies of features without applying an AE learning step. In particular, we first interpret the effect of feeding directly the features to the MLP classifiers using the naïve 0.5 threshold. In general, the accuracies expose some limitations, except for RGB-BOW, which exhibits an average accuracy of 73.3% and 74.9% for dataset 1 and 2, respectively, and WAV which scores 72.8% in dataset 2. We reasonably expect that the inclusion of an AE network, which can act as a nonlinear feature reduction means to get strong but still effective features with high compactness, may improve further these accuracies. Indeed, the dimensionality of the feature spaces induced by BoW, HoG and WAV is relatively high (200, 576 and 588, respectively).

As for the configuration of the size of the AE hidden layer, Fig. 3.7 (a) and 3.7(b) illustrate the average of the sensitivity and specificity accuracies achieved by the three feature representations successively transformed through AE learning step with the multilabeling layer (ML) in datasets 1 and 2. In particular, we have reduced the size of the input feature vectors into 5%, 10%, 20%, 30%, 40% and 50 % of its original size. In other words, the features are exposed to a reduction rate of 95%, 90%, 80%, 70%, 60% and 50% respectively. As can be observed in Fig. 3.7, there are some fluctuations of the average accuracy to the features reduction rate (i.e., size of AE hidden layer) which are not that high. Therefore, we adopt 80% features reduction rate in the remaining reported experiments.

The detailed accuracies of these results in terms of sensitivity (SENS) and specificity (SPEC) (along with their average) are reported in Table 3.3. By analyzing the obtained results, one can observe that a significant improvement of the average accuracy for the three tile representation strategies was possible by including an AE learning step with the conventional 0.5 threshold in both datasets. This clearly illustrates the effectiveness of exploiting the AE feature transformation. A further improvement of these obtained results has been yielded thanks to the addition of the multilabeling layer at the top of the MLP.

The best accuracies achieved in dataset 1 and 2 were by the WAV-AE(ML) and the RGB-BOW-AE(ML) strategies respectively. They have scored 84.0% of specificity and 79.9 % of sensitivity for the WAV-AE(ML) in dataset 1 and 84.6% of specificity and 76.7% of sensitivity for the RGB-BOW-AE(ML) in dataset 2. These strategies overcome the state-of-the-art method [11] set with an increment of around 6% for dataset 1 and 4% of average accuracy for dataset 2. The obtained bias values of the multilabeling layer are reported in Table 3.4. The values range from 0.11 up to 0.39, pointing out the importance of adjusting the threshold values of each class on the resulting classification accuracies.

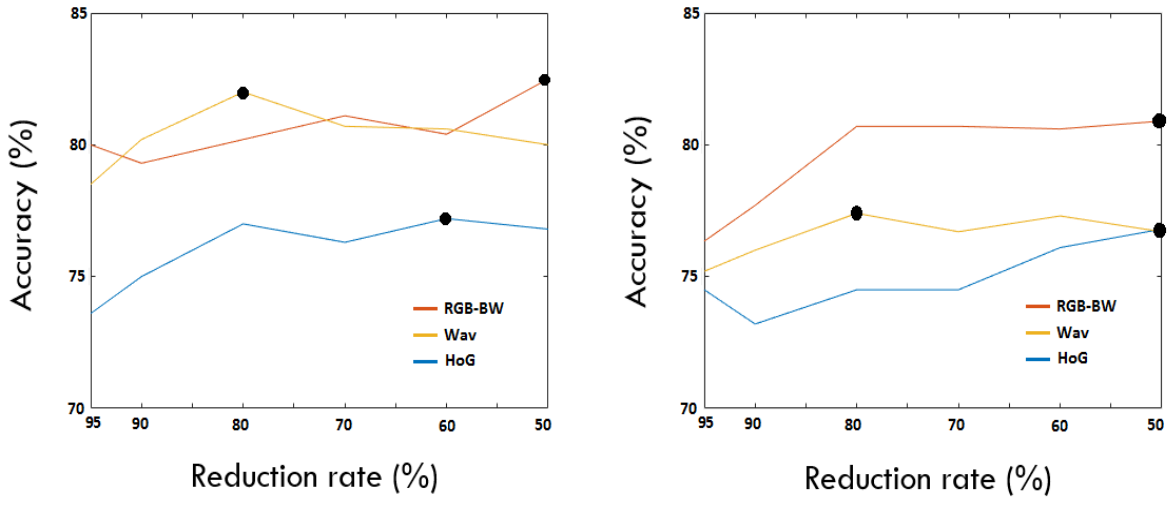


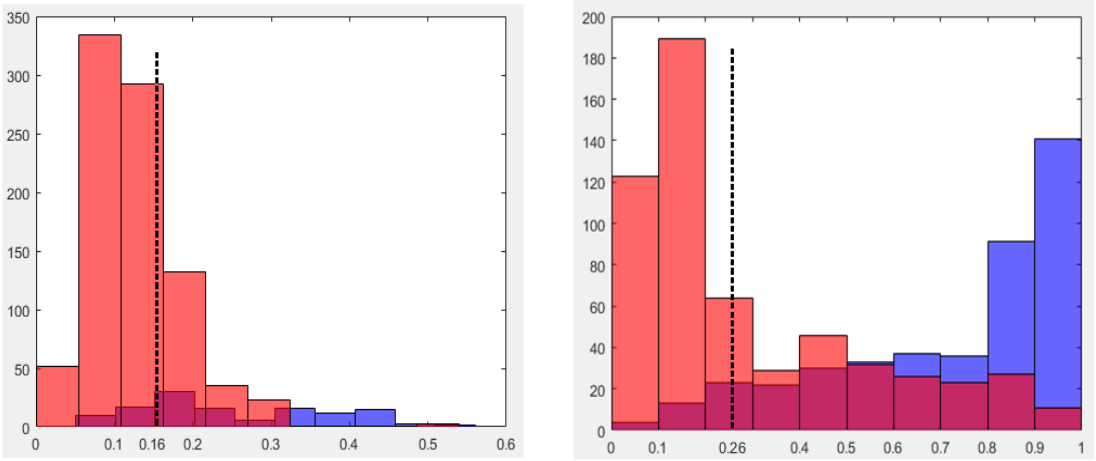
Fig. 3.7. The effect of the encoding rate on the average accuracy for (a) dataset 1 (Povo) and (b) dataset 2 (Civezzano).

Table 3.4. Best threshold values yielded by the Otsu's method for each of the AE-MLP based output classes.

	Bias values													
class	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}	b_{13}	b_{14}
HoG-AE (Dataset 1/Povo)	0.13	0.16	0.15	0.19	0.20	0.17	0.18	0.15	0.20	0.29	0.13	0.13	0.18	---
WAV-AE (Dataset 1/Povo)	0.11	0.11	0.35	0.38	0.10	0.33	0.32	0.24	0.23	0.17	0.11	0.29	0.21	---
RGB-BW-AE (Dataset 1/Povo)	0.17	0.26	0.22	0.24	0.19	0.31	0.22	0.16	0.24	0.22	0.11	0.23	0.26	---
HoG-AE (Dataset 2/Civezzano)	0.17	0.26	0.15	0.29	0.22	0.27	0.27	0.26	0.27	0.30	0.24	0.28	0.15	0.25
WAV-AE (Dataset 2/Civezzano)	0.29	0.30	0.33	0.30	0.22	0.30	0.30	0.29	0.30	0.30	0.30	0.30	0.29	0.30
RGB-BW-AE (Dataset 2/Civezzano)	0.21	0.12	0.19	0.16	0.31	0.21	0.21	0.23	0.17	0.30	0.31	0.17	0.26	0.37

Fig. 3.8 depicts an example of two MLP output histograms (the two first classes in dataset 1) and their decision threshold computed using OTSU's method (Fig. 3.5 (a) class Asphalt, Fig. 3.5 (b) Grass). As can be seen from the two histograms that the threshold values computed using OTSU's method show a better separable capability than the default threshold (th=0.5).

For the sake of further comparison, we considered the CNN-RBFNN (ML) methods detailed in section 3.2.2. Observing the results in Table 3.3, it comes out that, in overall terms, both RGB-BOW-AE (ML) and WAV-AE (ML) strategies respectively in dataset 1 and 2 perform almost equivalently compared to GoogLeNet-RBFNN (ML) accuracies. The sensitivity and specificity accuracies of each class of the compared strategies i.e., GoogLeNet-RBFNN (ML), RGB-BOW-AE (ML) and WAV-AE (ML) of both dataset 1 and 2 are detailed in Tables 3.5 and 3.6 respectively.



(a) (b)

Fig. 3.8. Example of two MLP output histograms and their decision threshold computed using OTSU's method for WAV-AE,(a) class Asphalt, (b) class Grass in dataset one (Povo).

Table 3.5. Sensitivity (SENS) and specificity (SPEC) accuracy achieved for each class by the reference method , the CNN-RBFNN and WAV-BOW with and without the multilabeling layer (ML) classifiers on dataset 1 (Povo).

		Asphalt	Grass	Tree	P.cross	Car	Person	Roof1	Roof2	B.Facade	Vineyard	S.Panel	Soil	Shadow
HCR-B [11]	**SENS**	55.9	70.3	75.3	**36.2**	27.9	0	50	63.6	21.7	52.3	**100**	**66.5**	46.8
	SPEC	93.6	81.5	61.7	98.4	97.3	98.8	98.6	86.6	96.1	92.2	99.0	86.9	91.4
GoogLeNet-RBFNN	**SENS**	47.0	80.9	68.4	0	53.4	0	53.0	60.6	4.3	71.2	33.3	51.9	44.4
	SPEC	**96.7**	72.6	**83.3**	100	98.9	100	**99.4**	**96.3**	99.3	93.9	99.6	88.6	95.9
GoogLeNet-RBFNN (ML)	**SENS**	67.2	89.2	69.9	20.7	**71.6**	0	65.2	68.6	**22.8**	**89.4**	66.7	66.0	67.3
	SPEC	89.6	60.2	80.9	99.7	93.3	99.7	98.0	94.2	93.7	83.4	98.1	77.5	85.7
WAV-AE	**SENS**	0	76.8	71.2	0	0	0	42.4	51.4	0	1.2	0	0	43.5
	SPEC	100	**83.9**	42.7	100	100	100	93.6	95.3	100	99.5	100	100	**96.7**
WAV-AE (ML)	**SENS**	**91.0**	**93.0**	**97.6**	0	8.3	0	**77.3**	**76.6**	5.4	88.6	0	22.3	**69.4**
	SPEC	70.9	53.4	11.6	100	99.3	100	77.3	81.7	98.5	79.1	99.9	**92.7**	83.9

Table 3.6. Sensitivity (SENS) and specificity (SPEC) accuracy achieved for each class by the reference method, the CNN-RBFNN and RGB-BOW with and without the multilabeling layer (ML) classifiers on dataset 1 (Civezzano).

		Asphalt	Grass	Tree	Vineyard	L.Vegetation	Car	Roof 1	Roof 2	Roof 3	S.Panel	B.Facade	Soil	Gravel	Rocks
HCR-B [11]	**SENS**	80.5	79.7	55.4	2.6	22.4	57.4	50	46.7	41.4	61.5	56.2	44.9	11.8	**77.2**
	SPEC	74.3	75.2	81.5	99.3	92.6	86.7	94.8	95.9	98.2	97.9	90.7	91.4	98.4	90.7
GoogLeNet -RBFNN	**SENS**	83.8	83.0	42.9	0	13.6	65.6	48.2	40.1	45.0	51.9	53.7	22.4	13.7	53.3
	SPEC	78.1	**76.0**	91.6	100	97.1	96.8	97.8	**97.2**	99.6	**99.6**	95.3	**98.9**	100	**98.8**
GoogLeNet -RBFNN (ML)	**SENS**	84.5	87.1	54.8	7.0	**45.6**	72.5	58.2	59.2	**67.5**	69.2	72.2	32.7	**25.5**	72.8
	SPEC	77.5	72.1	86.2	**99.5**	80.6	95.1	93.5	92.0	98.9	98.3	89.2	96.6	**100**	95.8
RGB-BOW-AE	**SENS**	76.6	88.5	33.4	0	1.6	61.5	40.6	43.4	20.4	51.9	45.1	25.6	0	64.1
	SPEC	**84.1**	71.3	**93.7**	100	**99.3**	**97.1**	**98.5**	97.1	**99.7**	99.1	**96.2**	98.8	100	98.4
RGB-BOW-AE (ML)	**SENS**	**92.5**	**97.8**	**56.2**	0	21.3	**90.2**	67.6	**65.8**	66.5	**78.8**	**88.9**	**60**	2.0	76.1
	SPEC	67.9	22.8	85.4	100	93.4	80.5	85.7	93.7	97.0	97.5	79.6	80.9	98.2	96.8

Concerning the processing time required per a single tile size of 224×224, one can notice that the AE based strategies are faster than both the CNN based strategies (i.e., AlexNet and GoogLeNet) and the state-of-art method [11]. In particular RGB-BOW-AE architechture reports to be 14 milliseconds faster than the state of the art method (25 against 39 milliseconds, respectively) and 15 milliseconds faster than AlexNet architecture which has scored 45 milliseconds. On the other hand, WAV-AE(ML) approach reports to be 5 times faster than AlexNet architecture (9 against 45 milliseconds, respectively) and almost 4 times faster than the reference method. This is explained by the fact that combining fast handcrafted features with shallow neural networks i.e., the one layer autoecnoder performs less computations than the Deep CNNs that are characterized by their large number of layers and blocks composing their architecture which results in further processing requirements. This suggests that this proposed architecture is an appropriate paradigm for real-time applications meeting together a satisfactory classification accuracy and reasonable processing time. The classification maps (qualitative results) of the proposed multilabel methods on some of the test images along with its related ground truth and original image are illustrated in Figures 3.9.1-3.9.3. The multilabel map allows depicting the spatial distribution of the object classes at a tile level. It can be noticed that most of the objects are sufficiently well detected.

Fig. 3.9-1 Qualitative map results of the 13 object classes obtained by the coarse WAV-AE (ML) classification technique on one of the test images of dataset 1(Povo) along with its related ground truth and original image.

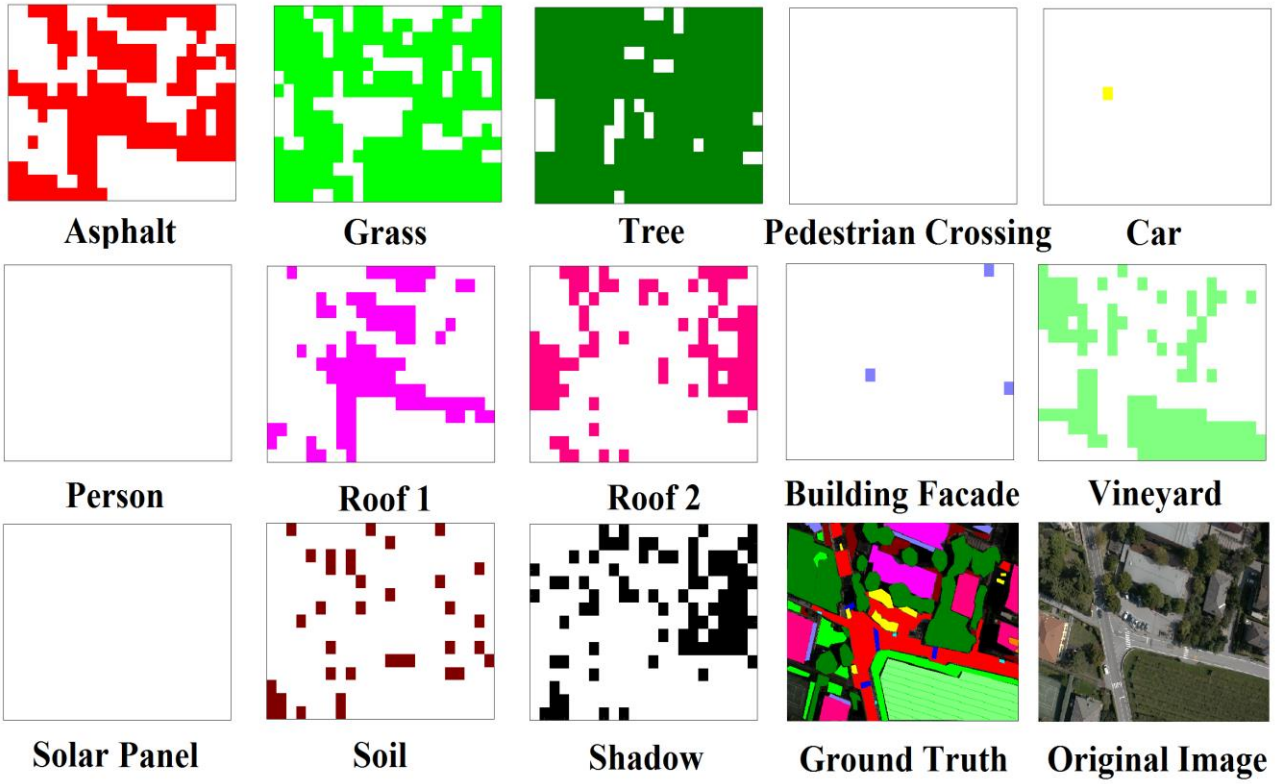


Fig. 3.9-2 Qualitative map results of the 13 object classes obtained by the coarse GoogLeNet-RBFNN (ML) classification technique on one of the test images of dataset 1(Povo) along with its related ground truth and original image.

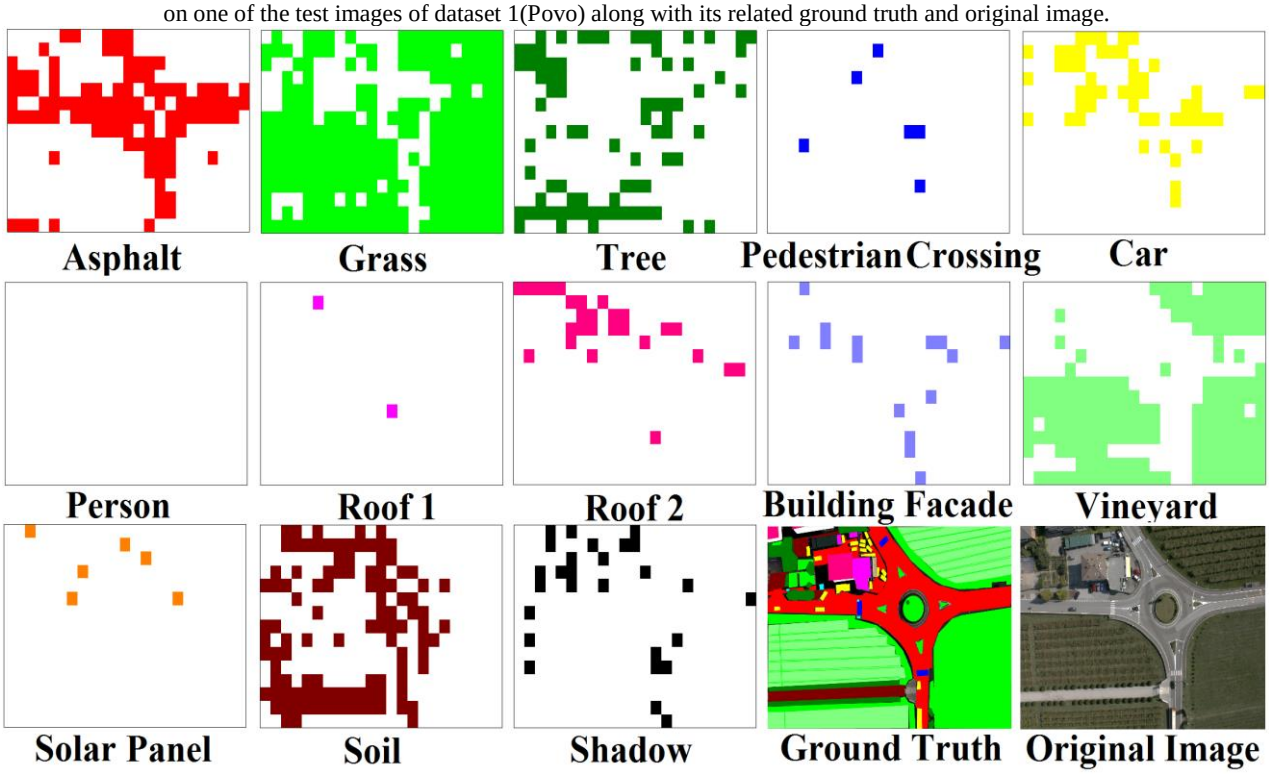
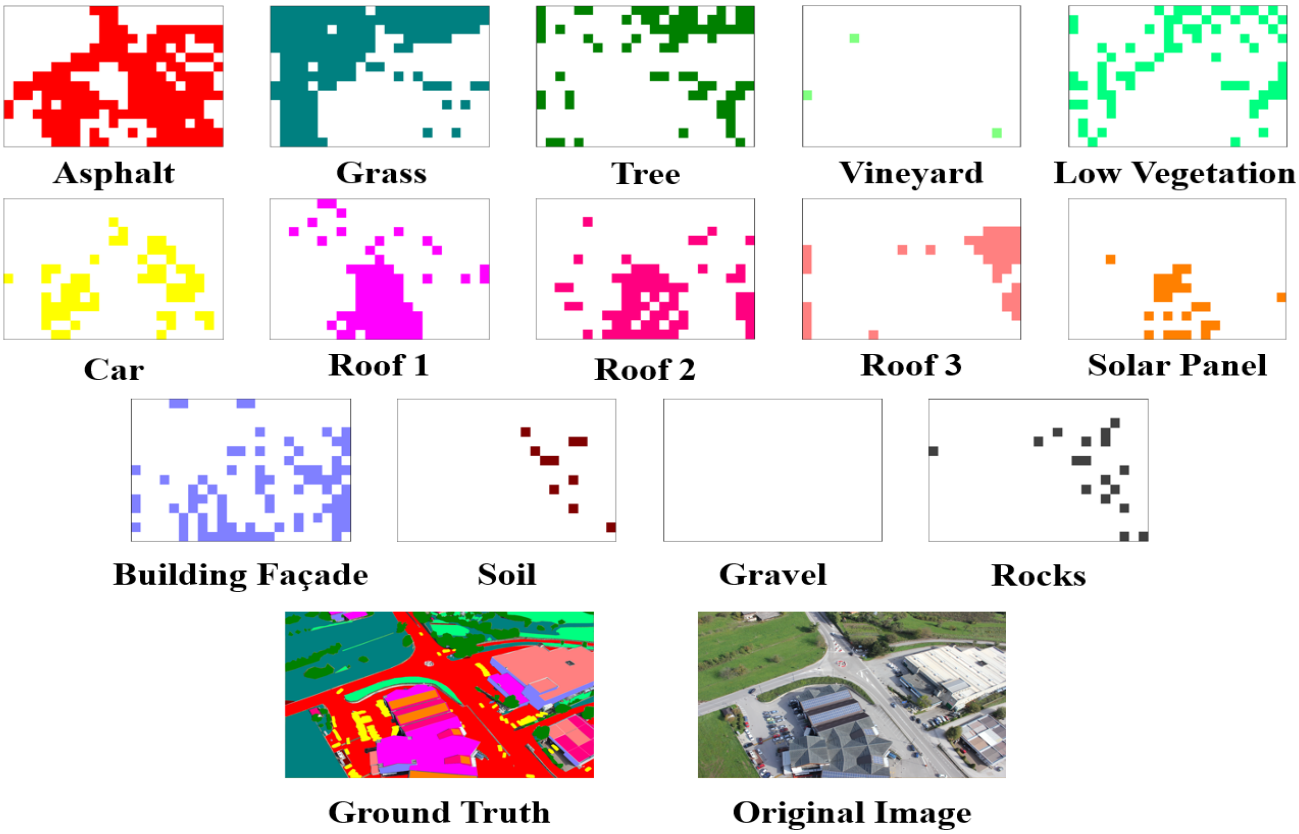


Fig. 3.9-3 Qualitative map results of the 14 object classes obtained by the coarse Alex-RBFNN (ML) classification technique on one of the test images of dataset 2 (Civezzano) along with its related ground truth and original image.



3.5. Conclusion

In this chapter, we have proposed two efficient multilabel classification methods for UAV images over urban areas. The proposed models start by subdividing the image into a set of non-overlapping equal tiles, each described by a list of objects present in it. In the first method, CNN features are extracted from each tile using the GoogLeNet pre-trained network. Then a RBFNN model is trained for the classification task. Despite the efficiency of deep CNNs as feature extractors, they are known to some extent to be computationally time-consuming for real-time scenarios, due to their deep architecture. As to cope with this, we propose in the second method to apply a two-stage signature representation for each tile. In particular, we apply a suitable handcrafted feature extraction method, followed by a feature learning step that enhances the features discrimination capability using an Autoencoder model. Then a multilayer perceptron network is trained for the multilabel classification task. Furthermore, for the multilabeling issue, a multilabel layer has been integrated on top of the whole proposed architectures to improve the obtained results. From the results, one can infer that the proposed methods are rather promising for EHR UAV multilabeling applications and can achieve substantial classification accuracy gains over the state-of-the art. The autoencoder-based methods allow obtaining good results especially in terms of processing time, making it a good candidate for real time urban monitoring applications. Moreover, we believe that this multilabel classification framework opens the door to exploit various other alternative solutions for the tile representation and the classification/matching

steps. As mentioned earlier, it is noteworthy that those methods could be applied to any typology of image descriptors and is not restricted to the three feature types considered in this chapter. A future development could consist in exploiting the spatial correlation between tiles.

3.6. REFERENCES

[1] "Nonproliferation: Agencies could improve information sharing and end-use monitoring on unmanned aerial vehicle exports," Jul. 2012.

[2] F. S. Leira, T. A. Johansen and T. I. Fossen, "Automatic detection, classification and tracking of objects in the ocean surface from UAVs using a thermal camera," in Proc. *IEEE Aerospace Conference*, Big Sky, MT, pp. 1-10, 2015.

[3] C. Luo, J. Nightingale, E. Asemota and C. Grecos, "A UAV-Cloud System for Disaster Sensing Applications," in Proc. *IEEE 81st Vehicular Technology Conference (VTC Spring)*, Glasgow, pp. 1-5, 2015.

[4] S. Li, H. Tang, S. He, Y. Shu, T. Mao, J. Li, Z. Xu, "Unsupervised Detection of Earthquake-Triggered Roof-Holes From UAV Images Using Joint Color and Shape Features," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 9, pp. 1823-1827, Sept. 2015.

[5] T. Moranduzzo and F. Melgani, "Detecting Cars in UAV Images With a Catalog-Based Approach," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 10, pp. 6356-6367, Oct. 2014.

[6] K. Liu and G. Mattyus, "Fast Multiclass Vehicle Detection on Aerial Images," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 9, pp. 1938-1942, Sept. 2015.

[7] S. Kamate, N. Yilmazer, "Application of Object Detection and Tracking Techniques for Unmanned Aerial Vehicles", in *Procedia Computer Science*, vol.61, pp. 436-441, 2015.

[8] Popescu, Dan, and Loretta Ichim. "Image Recognition in UAV Application Based on Texture Analysis." in Proc. *International Conference on Advanced Concepts for Intelligent Vision Systems*, pp. 693-704. Springer International Publishing, 2015.

[9] S. Malek, Y. Bazi, N. Alajlan, H. AlHichri and F. Melgani, "Efficient Framework for Palm Tree Detection in UAV Images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 12, pp. 4692-4703, Dec. 2014.

[10] J. Senthilnath, A. Dokania, M. Kandukuri, K.N. Ramesh, G. Anand, S.N. Omkar, "Detection of tomatoes using spectral-spatial methods in remotely sensed RGB images captured by UAV" *Biosystems Engineering*, vol. 146, pp. 16–32, 2016.

[11] T. Moranduzzo, F. Melgani, M. L. Mekhalfi, Y. Bazi and N. Alajlan, "Multiclass Coarse Analysis for UAV Imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 12, pp. 6394-6406, Dec. 2015.

[12] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proc. *IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov 1998.

[13] C. Szegedy, L. Wei, J. Yangqing , P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, "Going deeper with convolutions," in Proc. *IEEE Computer Vision and Pattern Recognition* (CVPR), Boston, MA, pp. 1-9, 2015.

[14] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, P. H. S. Torr, "Conditional Random Fields as Recurrent Neural Networks," in Proc. *IEEE Computer Vision (ICCV)*, Santiago, pp. 1529-1537, 2015.

[15] M. Chen, K. Q. Weinberger, F. Sha, and Y. Bengio, "Marginalized denoising auto-encoders for nonlinear representations," in *Proc. 31st Int. Conf. Mach. Learn.*, Beijing, China, 2014, pp. 1476–1484.

[16] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man. Cybern.*, vol. SMC-9, no. 1, pp. 62-66, 1979.

[17] J. Shan and E. Hussain, "Very High Resolution Remote Sensing Data: Processing Capabilities and Limitations in Urban Area," Purdue Univ. Sch. Civ. Eng. Ed, 2010.

[18] R. F. Nogueira, R. de Alencar Lotufo and R. Campos Machado, "Fingerprint Liveness Detection Using Convolutional Neural Networks," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 6, pp. 1206-1213, June 2016.

[19] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, J. Liang, "Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299-1312, May 2016.

[20] G. Strang and T. Nguyen, Wavelets and Filter Banks. Wellesley, MA: Wellesley-Cambridge, 1996.

[21] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Commun. Pure Appl. Math.*, vol. 41, no. 7, pp. 909–996, Oct. 1988.

[22] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *IEEE computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 886-893, 2005.

[23] L. Deng, and D. Yu, "Deep Learning: Methods and Applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197-387, 2014.

[24] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, J. Liang, "Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299-1312, May 2016.

Chapter IV

Multilabel Conditional Random Field Classification

Abstract– In this chapter, we formulate the multilabeling classification problem of unmanned aerial vehicle (UAV) imagery within a conditional random field (CRF) framework with the aim of exploiting simultaneously spatial contextual information and cross-correlation between labels. The pipeline of the framework consists of two main phases. First, the considered input UAV image is subdivided into a grid of tiles, which are processed thanks to an opportune representation and a multilayer perceptron classifier providing thus tile-wise multilabel prediction probabilities. In the second phase, a multilabel CRF model is applied to integrate spatial correlation between adjacent tiles and the correlation between labels within the same tile, with the objective to improve iteratively the multilabel classification map associated with the considered input UAV image. Experimental results achieved on two different UAV image datasets are reported and discussed.

4.1. Introduction

Unmanned aerial vehicles (UAVs) in the recent years have earned an exceptional standing within the remote sensing community especially in urban land use planning and analysis applications, and there are very good reasons for that. Due to their small size, rapid deployment, and high customizability, they provide a cheaper, safer and job faster platform in urban scenarios compared to other traditional data collection alternatives such as airborne and satellites. Furthermore, UAVs' capability of supporting high-resolution imagery and sensors in addition of their wide coverage capacity in a timely manner, grant the adequacy to collect data more efficiently with a higher level of details. In general, most methods introduced in the remote sensing literature have paid very scarce attention to multilabel recognition problem. This latter might be of a great importance in remote sensing imagery, where often many classes are likely to appear simultaneously. However in a classification setup, multilabel images pose a major challenge, since the classes may occur in different shape, appearance and texture complexity especially when dealing with extremely high resolution (EHR) images that contain a large amount of information details. This would augment the difficulty of finding a model that maps several target labels to a single instance using an opportune discriminative paradigm for all the considered object categories. Departing from this limitation, in [1] authors presented a novel classification framework for multilabel coarse scene description for EHR UAV imagery. It takes advantage of dividing the input image into a set of tiles, where each tile is described coarsely by the subset of classes present in it. Experimental results put forth a very promising performance compared to other pixel-based approaches. The latter multilabel paradigm has been adopted in other two works. The first one in [2]. It takes advantage of deep convolutional neural networks. It introduces a multilabeling layer (ML) integrated on the top of the whole architecture to improve the obtained outcomes. The second work which will be exploited in this chapter was put forth in [3]. It derives benefits from handcraft features followed by a feature learning step using autoencoder networks.

Structured Random Fields have been used in classification problems in remote sensing images by exploiting and integrating the spatially neighboring information within the decision process. Especially in EHR images, where a single object class tends to be represented by thousands of pixels where each pixel has some correlation degree with its close spatial match. To this point, the goal is to exploit the pixels that are associated to the same entity within the area of interest (i.e., Spatial contiguity). Markov random fields (MRFs) [4], and conditional random fields (CRFs) [5], are mathematical frameworks that model a given scene by expressing contextual

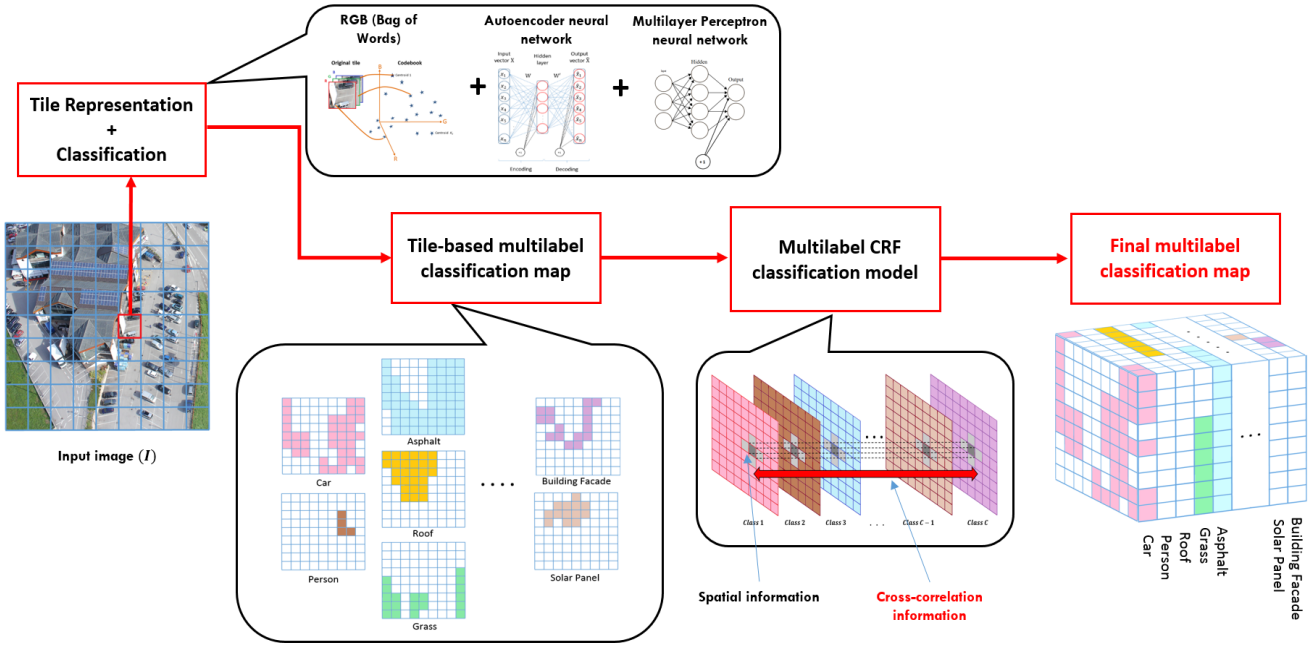


Fig. 4.1. Flow chart of the multilabel classification method.

information by means of adequate energy functions. They integrate spatial correlation of neighbors in a label image into a decision rule. This allows the reduction of the model complexity by passing from a global model to a model of local properties, defined in terms of both the potential function of single pixels and the interactions among pixels in appropriate neighborhoods. For instance, in [6], the authors put forth an overview comparison of classification methods which imposes a smoothness prior on the labels in areal images of high spatial resolution (HSR), and they have shown that exploring the spatial correlation enhances greatly the classification accuracy. Zhang et al. [7] introduce a multilevel Conditional Random Fields for multiclass pixel labeling in very high-resolution (VHR) optical remote sensing images. Volpi et al. [8] address the problem of semantic segmentation in urban remote sensing into land cover maps, they proposed to embed geographic context potentials (i.e., land cover classes' local co-occurrence and relative locations) into a pairwise CRF coupled with unary potentials from a random forest (RF) classifier. Proposed in [9], a multi-feature probabilistic ensemble conditional random field (MFPECRF) model to perform change detection task for HSR imagery. Paisitkriangkrai et al. [10] introduce a semantic pixel labeling framework of aerial and satellite imagery. They exploited different types of features (i.e., CNN features and hand-crafted features) in order to generate per-pixel class probabilities followed with a CRF as a post processing step. Another work in [11] presents a sub-pixel mapping algorithm based on CRFs for hyperspectral remote sensing imagery.

Accordingly, based over the state of the art reported so far, we propose in this chapter a novel multilabel conditional Random Field (CRF) framework for EHR UAV images. We formulate the concept of coarse description within a CRF perspective as a post processing step by applying it on a tile level, where we assign to each tile a vector of labels instead of one class label. The main novelty in this work is that the proposed CRF integrates two learning strategies, i.e., the spatial information within the same class jointly with the correlation information between different class labels. To the best of our knowledge, there is no work in literature that merges simultaneously both

spatial and correlation information for multilabel classification. We believe that such approach is subject to bridge the gap up to a reasonable extent, between the likely high semantic content of the UAV-grabbed images and their spectral information. Furthermore, it promotes a better perception of the tile labeling task taking into consideration its multilabel classification context extending the interactions between random variables in the output space. This learning strategy considers the multilabel vector descriptor that contains the list of objects of a query tile as new learning space viewed as a third dimension in the generated classification label maps (i.e., depth). See Fig. 4.1. Finally, we adopt the Iterated Conditional Modes (ICM) algorithm [12] to maximize the local conditional probabilities iteratively in the proposed multilabel CRF framework in order to regularize the final outcomes. The rest of this chapter is organized as follows, Section II recalls briefly the standard CRF and the tile-based coarse description and then outlines the proposed multilabel framework. Parameters estimation and experimental part is conducted in Section III. Finally, Section VI draws the conclusion and elaborates future developments.

4.2. Proposed Method

4.2.1. Monolabel Classification with CRF

Let X be the observed data from an input image defined as, $X = \{\boldsymbol{x}_i \in X | i = 1, \dots, n\}$ where $\boldsymbol{x}_i$ the observation data (i.e., the feature vector extracted) from the i^{th}site. Let Y be their corresponding labels, defined as $Y = \{\boldsymbol{y}_i \in Y | i = 1, \dots, n\}$. Unlike MRFs that are generative frameworks which regularize the classification output of an image by enforcing priors assumption between neighboring sites, discriminative CRFs are globally conditioned on the observations X, where they directly model the posterior distribution as a Gibbs model. This means that the potential functions in CRFs are more flexible in capturing complex spatial dependencies between labels [5]. The posterior distribution of Y conditioned on X is defined as:

$$P(Y|X) = \frac{1}{Z} \exp\big(-E(Y;X)\big) \qquad (1)$$

where Z is a normalizing constant, also called the partition function. According to (1), maximizing the a posteriori probability (MAP) of the whole image is equivalent to minimizing its corresponding energy function $E(Y;X)$. This latter is expressed as sum of unary and pairwise terms:

$$E(Y;X) = E_{data}(Y;X) + E_{spatial}(Y;X) \qquad (2)$$

where E_{data} represents the local decision term. It computes the cost of associating a given tile to a certain class without including its neighbors, which is equal to the likelihood function of the observed data at that site, and it is given by:

$$E_{data}(Y;X) = \sum_{i \in \mathcal{S}} V_1(y_i, X) \qquad (3)$$

with $\mathcal{S}$ being the set of total n sites. The second term $E_{spatial}$ is a smoothness term. It imposes spatial smoothness by penalizing dissimilarities between pair labels, and it is given by:

$$E_{spatial}(Y;X) = \sum_{i \in \mathcal{S}} \sum_{j \in N_i} V_2(y_i, y_j, X) \qquad (4)$$

where N_i is the neighborhood of site i. The order of the neighborhood system plays an important role in the spatial term and consequently affects the energy function [13]. Generally, the most common neighboring systems used in this framework are the first and the second order ones, namely, the 4-neighbor and 8-neighbor systems, respectively. The following subsection details the proposed CRF framework based on the tile-based multilabel paradigm.

4.2.2. CRF for Multilabel Classification

Let us consider a three-channel (RGB) image (I) acquired by means of a UAV. We start by subdividing I into a grid of tiles of equal size. This last is defined according to the spatial resolution of I and the expected sizes of objects to recognize. Our multilabel tile-based approach starts with the extraction of features. Since we are dealing with EHR imagery, each tile is characterized by a high level of detail and thus a rich information content. In order to extract a compact signature that describes efficiently each tile, we resort to a well-known strategy, namely, the bag of visual words (BOW) representation. To further boost its representation capability, we resort to an autoencoder neural network (AE). We feed the extracted features to an AE network, which constructs new learned features. The next step consists in adding a multilayer perceptron (MLP) network with a single hidden layer and sigmoid activation functions as a classifier at the end of the encoding part in order to classify the resulting features. This classifier fits our multilabeling requirements, and thereafter can be used on the test tiles to infer their object lists. Indeed, the MLP can handle simultaneously multiple outputs, which may characterize each tile of the query image [3]. The number of the MLP outputs C corresponds to the number of predefined object classes. The resulting MLP outputs generate what is called the classification maps X_k for $k \in \{1, \dots, C\}$, such that $X_k = \{x_{ki} \in X_k | i = 1, \dots, n\}$. These maps are inferred from the posterior probability distribution provided by the MLP outputs (Fig. 4.1). In each classification map X_k, the presence/absence of object k is indicated for each tile x_{ki}. In other words, each tile x_i is associated with a multilabel descriptor of size C, defined as $x_i = (x_{1i}, x_{2i}, \dots x_{Ci})$.

In dealing with multilabel imagery, one may observe that: 1) some labels are frequently correlated with others in many image tiles; and 2) some labels rarely appear with one another. For instance, the label ‘car’ is strongly correlated with the label ‘asphalt’, on the contrary, it is seldom that the label ‘car’ appears together with label ‘railway’ in the same image tile. To this end, the cross-correlation between labels can serve as an additional important source of information to build a robust classification framework with a strong capability of penalizing the co-occurrence of uncorrelated labels. To this purpose, we propose to encode the cross-correlation information between labels in a CRF model, namely, by changing the interaction potential term from a pairwise term (i.e., $E_{spatial}$) into a ternary one. According to the Hammersley-Clifford theorem [4], for each map k, we define a CRF over the outputs Y_k given the inputs X_k, through the following posterior distribution:

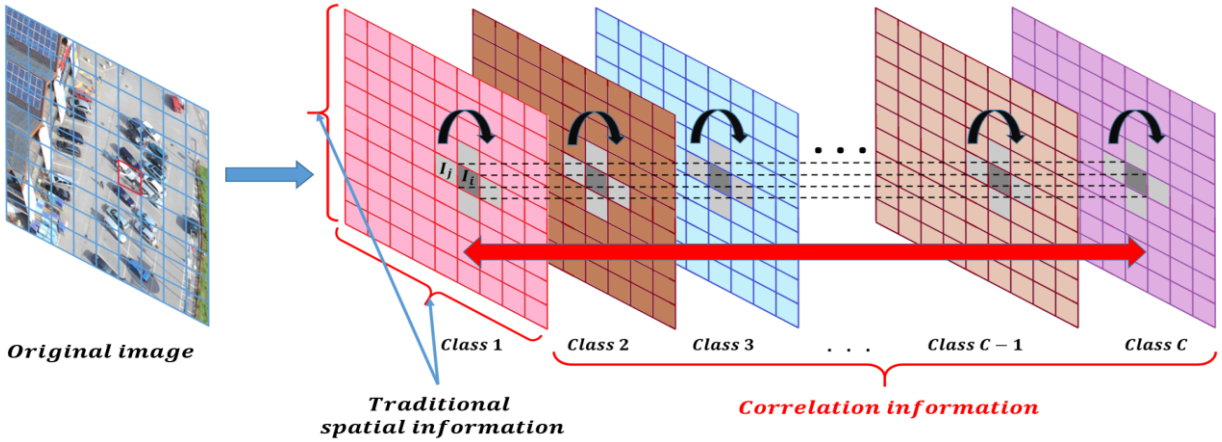


Fig. 4.2. The correlation and traditional spatial neighboring information in the proposed ML-CRF.

$$P(Y_k|X_k) = \frac{1}{Z}\exp\left\{\sum_{i=1}^{n} V_1(y_{ki}, X_k) + \sum_{i=1}^{n}\sum_{j\in\mathcal{N}_i}\sum_{l\in c_{ki}} V_2(y_{ki}, y_{kj}, y_{li}, X_k)\right\} \quad (5)$$

where V_1 is the unary potential, which is the probabilistic output of our discriminative classifier when considering each tile in isolation, and V_2 is the interaction term expressed as a trinary-wise potential that depends on 2 types of information. The first one is the traditional spatial information of neighboring tiles within the same class map k using the N_i neighborhood of tile i.
The second one is the information obtained from the neighborhood c_{ki} of tile i, namely, the multilabel components of the binary vector of tile i).

In this work, for simplicity, we consider the first order neighborhood system for the traditional spatial information and the labels belonging to the same tile (i.e., multilabel vector descriptor) for the cross-correlation information (Fig. 4.2). Moreover, we tackle each of these information sources separately. We thus quantify V_2 as a sum of two interaction terms, the first one over each map (spatial information) and the second one across the maps to encode the cross-correlation between the multiple labels lying within the same tile. Under these simplifying assumptions, the interaction potential V_2 in the posterior $P(Y_k|X_k)$ can be written as:

$$P(Y_k|X_k) = \frac{1}{Z}\exp\left\{\sum_{i=1}^{n} V_1(y_{ki}, X_k) + \sum_{i=1}^{n}\sum_{j\in\mathcal{N}_i} I_1(y_{ki}, y_{kj}, X_k) + \sum_{i=1}^{n}\sum_{l\in c_{ki}} I_2(y_{ki}, y_{li})\right\} \quad (6)$$

where I_1 is the interaction function at the level of each map (spatial term), and I_2 is the cross-correlation function, $c_k = \{(l,i), l \in \mathcal{C} \setminus k\}$. According to the CRF representation in [14], and after adding the cross-correlation term, (6) becomes:

$$P(Y_k|X_k) = \frac{1}{Z}\prod_i \psi_i(y_{ki}, X_k)\prod_{(i,j)} \psi_{i,j}(y_{ki}, y_{kj}, X_k)\prod_{(i,c_k)} \psi_{i,l}(y_{ki}, y_{c_k i}) \quad (7)$$

The terms ψ_i and ψ_{ij} are respectively the node and the edge potential functions over the map k, while ψ_{il} is the cross-correlation function (i.e., edge potential through different labels).
In the following, we define each of the potential functions, given that y_{ki} takes a binary state, $y_{ki} \in \{0,1\}$. The node potential takes the following form:

$$\psi_i(y_{ki}, X_k) = \left(e^{v_{i,1}^t f_{ki}}, e^{v_{i,2}^t f_{ki}}\right) \quad (8)$$

where $f_{ki} = [1, x_{ki}]$ and $v_{ki} = \{v_{ki,1}, v_{ki,2}\}$ are respectively, the node features and their associated weights.

The traditional edge potentials are defined as:

$$\psi_{i,j}(y_{ki}, y_{kj}, X_k) = \begin{pmatrix} e^{w_{kij,11}^t f_{kij}} & e^{w_{kij,12}^t f_{kij}} \\ e^{w_{kij,21}^t f_{kij}} & e^{w_{kij,22}^t f_{kij}} \end{pmatrix} \quad (9)$$

where f_{kij} are the edge features, defined as:
$f_{kij} = [1,\ h(x_{ki}, x_{kj})]$, with $h(x_{ki}, x_{kj}) = \frac{1}{1+|x_{ki}-x_{kj}|}$ being an arbitrary decreasing function that takes values in [0,1], and $w_{kij} = \{w_{kij,11}, w_{kij,12}, w_{kij,21}, w_{kij,22}\}$ being the weights associated with edges.

Regarding the correlation potentials $\psi_{i,l}$, we define first an auxiliary function $g_{kl}: \mathcal{L} \times \mathcal{L} \to [0,1]$ that measures the co-occurrence probability of two labels y_{li} and y_{ki} as:

$$g_{kl}(y_{ki}, y_{li}) = \frac{1}{|\{y_{li} = y_{ki}\}|}\sum_{i=1}^{n} 1_{\{y_{ki}\}}(y_{li})$$

$$\text{s.t}\ \ l \in \{1, \dots, C | l \neq k\} \quad (10)$$

Note that g_{kl} is defined when y_{li} and y_{ki} holds the same label value (i.e., 0 or 1), for the case where y_{ki} and y_{li} have different labels, we set $\overline{g_{kl}}$ as:

$$\overline{g_{kl}} = \overline{g_{kl}}(y_{ki}, y_{li}) = 1 - g_{kl}(y_{ki}, y_{li}) \quad (11)$$

The values of g_{kl} and $\overline{g_{kl}}$ are estimated from the training tiles to assess the correlation degree between label classes. Once g_{kl} and $\overline{g_{kl}}$ are computed, the cross-correlation weights for a given test tile at location i, are defined as:

$$\mu_{ki}(y_{ki}, y_{.i}) = \sum_{l \in c_k} 1_{\{y_{ki}\}}(y_{li}) g_{kl} + (1 - 1_{\{y_{ki}\}}(y_{li}))\overline{g_{kl}} \quad \text{s.t}\ y_{ki} = 1 \quad (12)$$

$$\tau_{ki}(y_{ki}, y_{.i}) = \sum_{l \in c_k} 1_{\{y_{ki}\}}(y_{li}) g_{kl} + (1 - 1_{\{y_{ki}\}}(y_{li}))\overline{g_{kl}} \quad \text{s.t}\ y_{ki} = 0 \quad (13)$$

The functions μ_{ki} and τ_{ki} define the cross-correlation weights for the two possible states of label y_{ki}, namely, presence and absence respectively by using the sum of the label pairs scores to evaluate the degree of the relationship of a given test tile. Subsequently, the correlation potential is given as:

$$\psi_{i,l}(y_{ki}, y_{c_k i}) = (e^{\lambda_1 \mu_{ki(y_{ki}, y_{.i})}}, e^{\lambda_2 \tau_{ki}(y_{ki}, y_{.i})}) \qquad (14)$$

where $\lambda_1$and λ_2 are coefficients used to ponder the importance between the two states that a given label may take i.e., presence and absence, respectively.

In order to obtain an optimum multilabeling solution over a given test image, the use of an inference algorithm during the training phase is necessary in order to estimate the parameters (i.e.,v_{ki}, w_{kij}) of the graphical model. For such purpose, we make use of the loopy belief propagation (LBP) method [15], which is a widely used inference procedure in approximation algorithms for graph structures [16]. As mentioned earlier, in the test phase, we adopt the iterated conditional modes (ICM) algorithm. This algorithm maximizes the local conditional probabilities iteratively, given an initial labeling. The label that maximizes the local conditional probability is chosen as an optimal local solution [12].

In this chapter, the local probabilities of each tile are conditioned by its multilabel vector descriptor in addition to the corresponding neighboring tiles over the same label map level. Starting from an initial multilabel combination generated from the output of the MLP classifier, at each iteration, the ICM maximizes the conditional MAP estimation:

$$\widehat{y_{kl}} \leftarrow arg \max_{y_{ki}} P(y_{ki}/y_{k\mathcal{N}_i}, y_{c_k i}, X_k) \qquad (15)$$

passing through all label maps of the test image repeatedly up to a convergence is reached, producing thus a final multilabel classification map of the considered image. Algorithm I. below illustrates the related multilabel framework pseudo code.

4.3. Experimental Validation

4.3.1. Dataset Description and Experimental Setup

In order to evaluate the performance of the proposed classification method, we exploited two real datasets of UAV and airborne images acquired over two different locations. The first set of images is a UAV dataset that was aquired near the city of Civezzano (Italy) at different off-nadir angles, on the 17th October 2012. The acquisition was performed with a picture camera Canon EOS 550D characterized by a CMOS APS-C sensor with 18 megapixels. These UAV images are characterized by three channels (RGB) with a spatial resolution of around 2 cm. The image size is 5184×3456 pixels and the radiometric resolution is 8 bits datasets. This dataset is composed of 10 images, subdivided into two groups (3 training and 7 test images). They contain 14 classes of objects, namely, ‘Asphalt’, ‘Grass’, ‘Tree’, ‘Vineyard’, ‘Low Vegetation’, ‘Car’, ‘Roof 1’, ‘Roof 2’, ‘Roof 3, ‘Solar Panel’, ‘Building Facade’, ’ Soil’, ‘Gravel’, and ‘Rocks’.

Algorithm: Pseudo-code of the proposed CRF multilabeling algorithm

Training Phase

1. *Build the graph and estimate the weights by means of LBP algorithm*
2. *Estimate cross-correlation weights with Eq. 12 and 13*

Test Phase

3. *Set iteration index S=0*
4. *Initialize labels Y*

   $Y \leftarrow ML\_unary$ (i.e., $max \prod_i \psi_i(y_i, X)$

5. ***While*** $convergence\ not\ reached$

   For each class-map y_k

   For each site y_{ki}

   Compute $Potential\ (y_{ki}) = \psi_{k,i} * \psi_{k,i,j} * \psi_{i,l}$

   Update y_{ki} with best state (present/absent) such as:

   $y_{ki} \leftarrow max(Potential(y_{ki}))$

   EndFor

   EndFor

   $S = S + 1$

   EndWhile

The second dataset was originally acquired for the sake of vehicle detection over Munich, Germany [17], but here we use it for land use classification. The images were captured from an airplane by a Canon Eos 1Ds Mark III camera with a resolution of 5616×3744 pixels, 50 mm focal length. The images were taken at a height of 1000 meters above ground, with a spatial resolution of approximately 13 cm. This dataset is composed of 20 images (10 training and 10 test images) and contain all predefined 16 classes of objects, which are 'Slate Roof', 'Clay Roof', 'Asphalt', 'Standing Steam Roof', 'Sports Field', 'Green Trees', 'Dried Trees', 'Grass', 'Building Site/Operating Machinery', 'Paved Road', 'Soil', 'Dark Soil', ' Green Roof', 'Flat Concrete Roof',' Railway', and 'Cars'. We subdivided each image in both the first and the second datasets into a non-overlapping grid of equal tiles of 50×50 pixels as explained in the previous section.

4.3.2. Evaluation metrics and experimental results

The effectiveness of the proposed framework is quantified in terms of sensitivity (SENS), specificity (SPEC) and their average. The parameters of the proposed multilabel CRF model i.e., v_{ki}, w_{kij} are learned from the training data using the Loopy Belief Propagation (LBP) inference algorithm. We set $v_{ki2} = 0\ and\ w_{kij12} = w_{kij21} = 0$ in order to avoid over-parametrization of the model [14]. For cross-correlation weights μ_{ki} and τ_{ki} they are estimated during the training phase as described in equations (12) and (13) using the training labels.

The free correlation parameters λ_1, λ_2 are fixed to $\lambda_1 = \lambda_2 = 1$. This means, they do not have any effect on the weights μ_{ki} and τ_{ki} of the correlation function $\psi_{i,l}$ and thus on the final classification results. The main purpose behind adding these two parameters into the proposed method is to provide a robust model that is able to cope with different types of datasets in which the presence and the absence states of the investigated labels are imbalanced (i.e., presence states

Table 4.1. Sensitivity (SENS), specificity (SPEC) and average (AVG) accuracies in percent obtained by the different classification methods on datasets 1 and 2.

	Dataset 1(Civezzano)			Dataset 2(Munich)		
	Accuracy (%)			Accuracy (%)		
METHOD	SPEC	SENS	AVG	SPEC	SENS	AVG
ML-Unary	**98.2**	62.6	80.4	**97.8**	55.6	76.7
ML-CRF	92.5	70.6	81.5	94.2	61.4	77.8
Full-ML-CRF	90.8	**75.9**	**83.4**	90.7	**68.5**	**79.6**

are relatively much bigger than the absence states and vice versa). The effect of varying the correlation parameters on the obtained results is analyzed later in this section.

As side notes, it is worth to mention that: 1) it is possible to recover an MRF model (i.e., the joint probability $P(Y_k, X_k)$) by setting the edge features $f_{kij} = 1$, so that w_{kij} will represent the unconditional potential edge between nodes i and j; and 2) once we set $\lambda_1 = \lambda_2 = 0$, we recover the traditional monolabel CRF that works at the level of each class map separately.

For the sake of comparison, we confront the proposed multilabel scheme with unary scores strategy obtained by means of RGB bag-of-words features coupled with autoencoder network and MLP classifier [3] as described earlier (termed as ML-Unary), along with the traditional monolabel CRF reference method which was run on each binary map (class) independently from the others, not taking into account the multilabel context (i.e., $\lambda_1 = \lambda_2 = 0$). In the following, this reference method is called ML-CRF. The Full-ML-CRF stands for the proposed method which exploits also the interclass correlation as described earlier. The results are summarized in Table 4.1.

Our proposed strategy (Full-ML-CRF) outperforms both the ML-Unary and ML-CRF methods in terms of average accuracy scoring, 83.4% and 79.6% for datasets 1 and 2, respectively. It records an increment of around 3 % and 2 % in dataset 1 and dataset 2 over ML-Unary and ML-CRF methods respectively. Moreover, our strategy offers the advantage of yielding higher sensitivity in both datasets while maintaining an appropriate rate of specificity. An interesting fact to point out is that the spatial information has allowed to recover some lost objects (true positives) but at the expense of a higher number of errors on absent objects (true negatives). The exploitation of the cross-correlation information by Full-ML-CRF has led to a further substantial boost in the correct detection of true positives (and thus a higher SENS).Indeed, the exploitation of spatial information incurs in a loss of true negatives (confirmed for both datasets) which are typically dominant in multilabel maps. This is however accompanied by an increase of the true positives, which can be very substantial (case of Full-ML-CRF on both datasets). The detailed experimental results in terms of sensitivity (SENS) and specificity of each class (SPEC), along with their average (AVG) of datasets 1 and 2 are reported respectively, in Table 4.2, 4.3.

Another element to discuss is the influence of varying the correlation parameter values i.e., λ_1, λ_2 on the classification outcomes. Fig. 4.2 describes the behavior of the average accuracy against λ_1 and λ_2. By analyzing Fig. 4.3 (a), one can notice that the best results are obtained when $\lambda_1$and λ_2 take similar values. Indeed, balancing between λ_1 and λ_2 which represent respectively the presence and the absence state of a given label leads to a better balance between sensitivity

and specificity. By contrast, magnifying one parameter at the expense of the other, leads to a very high sensitivity (i.e., 100%) and very low specificity (i.e., 0%) or vice-versa, which results in 50% of average accuracy. Same observations apply for Dataset 2 as can be seen in Fig. 4.3 (b). Fig. 4.4 illustrates examples of multilabel classification maps obtained with the proposed Full-ML-CRF along with the two reported methods on two test images from dataset 1.

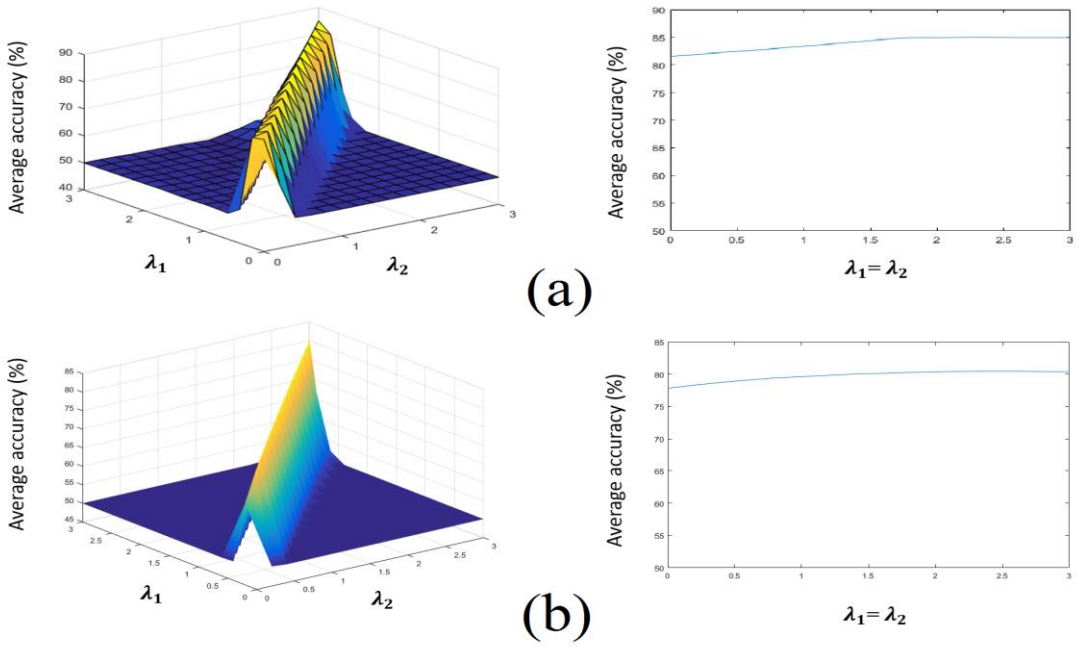


Fig. 4.3. Average accuracy versus spatial parameters λ_1, λ_2 achieved by the Full-ML-CRF method on (a) dataset 1 (Civezzano) and (b) dataset 2 (Munich).

Table 4.2. class-by-class accuracy performances achieved by the three models on dataset 1 (Civezzano).

	class	1	2	3	4	5	6	7	8	9	10	11	12	13	14
ML-Unary	**Sens**	66.3	50.1	69.3	52.2	48.0	70.2	60.3	55.1	43.8	74.4	85.8	64.7	84.4	73.3
	Spec	96.7	96.6	99.0	97.9	94.9	99.2	98.0	99.1	97.6	99.2	99.2	98.5	99.4	98.9
	Ave	**81.5**	73.3	**84.1**	75.1	71.4	84.7	79.1	77.1	70.7	86.8	92.5	81.6	91.9	86.1
ML-CRF	**Sens**	55.1	60.2	71.3	69.6	60.0	78.7	71.2	57.0	62.2	82.0	88.1	78.4	88.2	83.2
	Spec	87.5	88.5	88.1	87.7	87.9	89.2	89.1	97.5	94.2	97.3	97.6	95.8	97.0	96.8
	Ave	71.3	74.3	79.7	78.7	73.9	84.0	80.2	77.2	78.2	89.6	92.8	87.1	92.6	90.0
Full-ML-CRF	**Sens**	66.1	66.9	74.2	76.1	67.7	83.8	77.6	61.2	68.9	85.6	90.4	86.1	90.4	86.5
	Spec	84.5	85.2	86.3	86.7	86.0	89.8	86.8	96.8	91.2	96.1	96.6	94.0	96.0	95.4
	Ave	75.3	**76.0**	80.3	**81.4**	**76.8**	**86.8**	**82.2**	**79.0**	**80.1**	**90.9**	**93.5**	**90.0**	**93.2**	**90.9**

Table 4.3. class-by-class accuracy performances achieved by the three models on dataset 2 (Munich).

	class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ML-Unary	Sens	18.8	73.5	74.3	45.2	13.6	70.8	28.3	64.4	0	0.4	33.7	25.5	43.7	5.3	33.4	16.2
	Spec	99.4	98.9	88.6	98.7	99.9	88.7	97.4	88.4	100	99.9	98.7	99.3	99.7	99.6	99.6	99.0
	Ave	59.1	**86.2**	**81.5**	72.0	56.8	**79.7**	62.8	76.4	50	50.2	66.2	62.4	71.7	52.4	66.5	57.6
ML-CRF	Sens	52.5	50.4	51.0	54.3	58.0	59.8	57.8	60.0	60.4	66.5	65.6	69.5	68.6	68.7	64.3	73.6
	Spec	95.4	95.7	94.9	94.8	94.3	93.1	92.7	93.3	92.8	93.1	93.5	94.9	93.1	93.8	94.5	96.9
	Ave	73.9	73.0	72.9	74.5	76.2	76.5	75.3	76.6	76.6	79.8	79.6	82.2	80.8	81.3	79.4	85.3
Full-ML-CRF	Sens	61.8	60.2	57.6	60.8	62.2	64.4	67.1	67.4	68.0	73.3	74.5	77.0	75.6	75.4	70.9	79.0
	Spec	91.7	91.8	91.6	91.3	91.5	89.7	87.7	89.1	87.9	88.5	90.8	92.6	89.3	90.8	91.5	94.7
	Ave	**76.8**	76.0	74.6	**76.0**	**76.8**	77.0	**77.4**	**78.2**	**77.9**	**80.9**	**82.6**	**84.8**	**82.4**	**83.1**	**81.2**	**86.8**

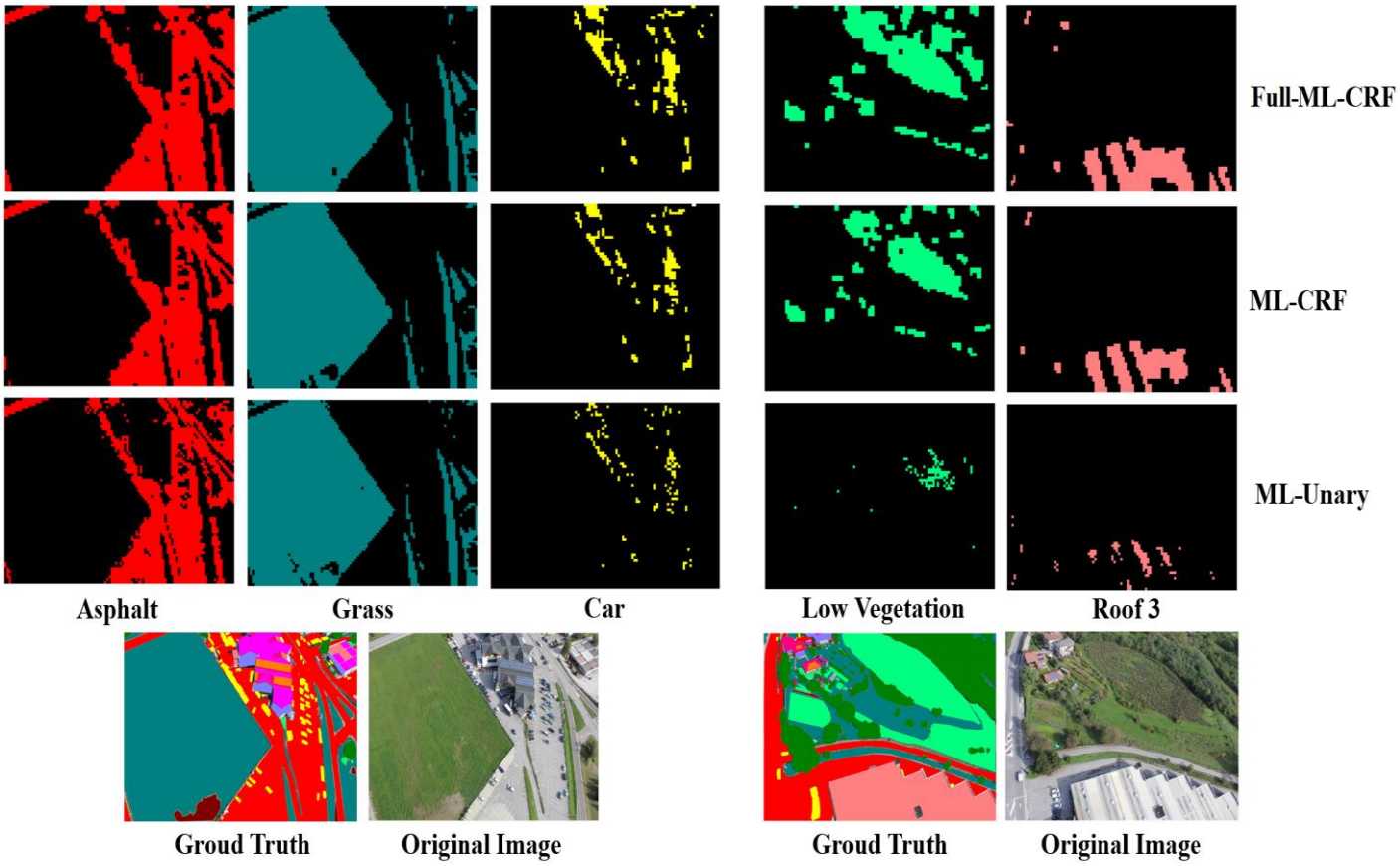


Fig. 4.4. Examples of multilabel classification maps obtained by the three classification methods (ML-Unary, ML-CRF, Full-ML-CRF) on two test images of dataset 1(Civezzano), along with their related ground truth and original image.

4.4. Conclusion

This chapter has introduced a novel multilabel conditional random field model for UAV image classification. The underlying idea is to exploit simultaneously label cross-correlation and spatial contextual information within a structured prediction framework. The proposed model starts with the subdivision of the image into a set of equal tiles, then an opportune classifier is used to generate initial predictions for each tile. Afterwards, a CRF model is applied on the resulting multilabel map to iteratively improve it. Experimental results show that the exploitation of both spatial alongside (across map) label-label information can boost significantly the quality of the multilabel map. As future development, expanding the action field of the cross-correlation term by considering wider neighborhood systems could be worth investigating, though this would lead to an increase of the method complexity. Moreover, an automatic way for estimating the optimal value of λ_1=λ_2 (here fixed to 1) could be interesting to improve further the results as Fig. 4.3 suggests.

Acknowledgment

The Authors would like to thank Mark Schmidt for supplying the UGM toolbox used in the context of this chapter. Link: https://www.cs.ubc.ca/~schmidtm/Software/UGM.html.

4.5. REFERENCES

[1] T. Moranduzzo, F. Melgani, M. L. Mekhalfi and Y. Bazi, "Multiclass coarse analysis for UAV imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 12, pp. 6394-6406, 2015.

[2] A. Zeggada, F. Melgani and Y. Bazi, "A deep learning approach to UAV image multilabeling," *IEEE Geoscience and Remote Sensing Letters*, no. 99, pp. 1-5, 2017.

[3] A. Zeggada, and F. Melgani, "Multilabeling UAV Images with Autoencoder Networks ", *IEEE Urban Remote Sensing Event(JURSE)* in Dubai, 2017.

[4] J. Besag. "Spatial interaction and the statistical analysis of lattice systems", *Journal of the Royal Statistical Society*. Series B, 36(2):192–236, 1974.

[5] J. Lafferty, A. McCallum and C.N.Pereira ," Conditional Random Fields: *Probabilistic Models for Segmenting and Labeling Sequence Data", Proceedings of the 18th International Conference on Machine Learning (ICML),* pages 282-289, Williamstown (USA), 2001.

[6] K. Schindler, "An Overview and Comparison of Smooth Labeling Methods for Land-Cover Classification," *in IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 11, pp. 4534-4545, Nov. 2012.

[7] T. Zhang, W. Yan, J. Li and J. Chen, "Multiclass Labeling of Very High-Resolution Remote Sensing Imagery by Enforcing Nonlocal Shared Constraints in Multilevel Conditional Random Fields Model," *in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 7, pp. 2854-2867, July 2016.

[8] M. Volpi and V. Ferrari, "Structured prediction for urban scene semantic segmentation with geographic context," *Joint Urban Remote Sensing Event (JURSE)*, pp. 1-4. Lausanne, 2015.

[9] P. Lv, Y. Zhong, J. Zhao, H. Jiao and L. Zhang, "Change Detection Based on a Multifeature Probabilistic Ensemble Conditional Random Field Model for High Spatial Resolution Remote

Sensing Imagery," *in IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 12, pp. 1965-1969, Dec. 2016.

[10] S. Paisitkriangkrai, J. Sherrah, P. Janney and A. van den Hengel, "Semantic Labeling of Aerial and Satellite Imagery," *in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 7, pp. 2868-2881, July 2016.

[11] J. Zhao, Y. Zhong, Y. Wu, L. Zhang and H. Shu, "Sub-Pixel Mapping Based on Conditional Random Fields for Hyperspectral Remote Sensing Imagery," *in IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 6, pp. 1049-1060, Sept. 2015.

[12] J. Besag, "On the staticical Analysis of Dirty Pictures", *Journal of Royal Statistical Society, Series B (Methodological),* VOL. 48, no. 3, pp. 259-302, 1986.

[13] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distribution, and The Bayesian Restoration of Images", *IEEE Transaction on Pattern Analysis and Machine Intelligence,* VOL, PAMI6. Issue 6, 1984.

[14] M. Schmidt, K. Murphy, G. Fung and R. Rosales, "Structure learning in random fields for heart motion abnormality detection," *IEEE Conference on Computer Vision and Pattern Recognition*, Anchorage, pp. 1-8 AK, 2008.

[15] J. Pearl, "Reverend Bayes on inference engines: A distributed hierarchical approach" . *Proceedings of the Second National Conference on Artificial Intelligence*, California: AAAI Press. pp. 133–136. 1982.

[16] V. Lempitsky, C. Rother, S. Roth and A. Blake, "Fusion Moves for Markov Random Field Optimization," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 8, pp. 1392-1405, Aug. 2010.

[17] K. Liu and G. Mattyus, "Fast multiclass vehicle detection on aerial images," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 9, pp. 1938-1942, 2015.

Chapter V

Spatial and Structured SVM for Multilabel Image Classification

Abstract *– We describe a novel multilabel classification approach based on support vector machine (SVM) for extremely high-resolution (EHR) remote sensing images. Its underlying ideas consist: 1) to exploit inter-label relationships by means of a structured SVM; and 2) to incorporate spatial contextual information by adding to the cost function a term which encourages spatial smoothness into the structural SVM optimization process. The resulting formulation appears as an extension of the traditional SVM learning, in which our proposed model integrates output structure and spatial information simultaneously during the training. Numerical experiments conducted on two different UAV and airborne acquired sets of images show the interesting properties of the proposed model in particular in terms of classification accuracy.*

5.1. Introduction

Remotely sensed data have constantly given us valuable information for various applications such as global-local environmental changes, urban growth, etc [1] [2] [3]. Suitable related analysis tools have helped us to understand their insights more deeply and to create timely land use maps with less labor force. Combined with the fast technological developments, remotely sensed data have been continuously getting more informative and at the same time challenging. Especially, unmanned aerial vehicles (UAVs), commonly known as drones, have attracted huge attention recently in the remote sensing community due to their interesting potentials for data collectability, customizability, portability, and cost efficiency. They have made it possible to gather extremely high-resolution (EHR) image data and further analyze tinier scale objects, which even enable us to develop improved or new practical systems such as in traffic monitoring, precision farming, and disaster victim detection. For instance, the authors in [4] addressed the problem of processing real-time traffic data extracted from videos mounted on UAVs for traffic monitoring applications, where the camera platforms move with UAVs. The proposed system quickly estimates traffic parameters such as traffic directions, number of vehicles, and their average stream speed. Another work in pedestrian detection in UAV surveillance camera applications was put forth in [5]. This algorithm relies basically on the combination of Haar-LBP (local binary pattern) descriptors in Adaboost cascade classifiers along with mean shift algorithm to improve their pedestrian detection capability. The authors in [6] proposed an effective strategy to detect and track multiple UAVs with a single moving camera mounted on a UAV to systemize collision avoidance systems. The detector refined by Kalman filter showed the effectiveness of their approach. As for precision agriculture and forest monitoring applications, relatively, much contribution could be found in the literature. In [7], for instance, an automatic estimation system to map land use in a vegetable farm was introduced on the basis of UAV images. In [8], the authors presented a successful system to delineate eroded areas in a tropical rain forest using digital cameras in UAVs. In disaster and victim detection applications, Bejiga *et al.* [9] interestingly suggested a method of assisting avalanche search and victim detection with UAVs fitted with vision cameras. In [10], a method for detecting and evaluating flooded areas in natural disaster scenarios was developed, where they use LBP descriptors extended to color information of UAV images.

One of the major applications of remote sensing data is classification, whose task is to assign class labels to certain regions defined by pixels or grids using trained classifiers. Object classification is a well-studied area in the remote sensing community, and many refined approaches have been proposed in the past few years. For instance, the authors in [11] proposed a

sparse land cover classification framework with heterogeneous feature extraction for airborne LiDAR point cloud data. A multiple kernel learning is embedded into the sparse representation classification, which enhances classification performances on LiDAR data. Mei *et al.* [12] introduced a convolutional neural network (CNN) that intends to learn spatial-spectral features for efficient hyperspectral classification. The network was tested on four benchmark data sets from two sensors, and the result has shown an improvement over the state-of-the-art CNN-based classification methods. Zhang *et al.* in [13] proposed a deep learning approach for object-based land cover classification in very high-resolution UAVs images. It makes use of stacked denoising autoencoder (SDAE) networks. The proposed method has shown a good performance especially when sufficient training samples are lacking. Another deep neural network based on recurrent neural networks was proposed in [14] for hyperspectral image classification, in which they utilized a new activation function called parametric rectified tanh. Experimental results on three airborne hyperspectral images put forth a very promising performance compared to other deep neural networks. However, the efforts mainly focus on satellite and airborne images. Compared with such very high-resolution images, EHR images are characterized by microscale: a few centimeters' resolution. Due to this fact, traditional pixel-based and segment-based classification strategies may not be able to provide reasonable prediction especially when dealing with tens of classes at the same time. Additionally, the plenty of information supplied by EHR images makes classification itself more challenging. In this context, a coarse description approach was first proposed in [15] to tackle these problems raised by EHR images. In particular, the authors proposed subdividing images into a grid of tiles, which creates multilabeled outputs whose binary descriptors represent the presence and absence of target objects. This strategy casts the land cover classification problem to a more complicated multilabel classification formulation.

Multilabel classification is one of the most challenging tasks in pattern recognition because, unlike multiclass cases where samples belong to only one class, multilabel outputs can exhibit more than one label simultaneously. Accordingly, the output space of multilabeling problems exponentially grows as the number of classes. Besides, such data normally contain complicated structure in samples and label sets as mentioned later. An intuitive way to deal with multilabel classification consists in decomposing the multilabel problem into a set of binary classification tasks. Afterwards, a group of binary classifiers are trained independently (i.e., one classifier per label) and the combination of their predictions is exploited to yield the final output. Boutell *et al.* introduced an interesting multilabel learning scheme in [16] based on such approach. Its underlying idea is to deal with each label combination in the training set as an identifier of a new distinct class. The resulting set of label values is then processed by a monolabel classifier. Its main issue is the limitation of label combination samples in training sets. An alternative approach to deal with multilabel sets is to adopt the monolabel models to perform directly the multilabel classification. However, some models are less easily adapted to be extended for multilabel tasks than others. In this context, the authors of [15] presented a multilabel pattern matching based on Chi-squared distance. In [17], they used an RBFNN with multiple outputs to fit the multilabeling requirements. Other methods in the literature were developed so as to better exploit correlation information between labels. For instance, in [18], a graph-based multilabel classifier is proposed by exploiting low rank representation. In [19], a correlated logistic model is introduced for joint prediction in multilabel classification problems. It consists in extending independent logistic

regressions by explicitly modeling the pairwise correlation among labels. Its model is formulated using an elastic regularized maximum pseudo-likelihood estimation, exploiting sparsity in both feature selection and label correlation. Such a formulation results in a linear computational complexity with respect to the number of labels. Another interesting multilabel framework based on deep neural networks can be found in [20]. Its cost function takes advantage of combining: 1) a max-margin term, which maximizes the score of present labels over absent ones through a predefined margin; 2) a max-correlation criterion, which maximizes the correlations between the extracted features and their corresponding labels within a learned semantic space; and 3) a correntropy (correlation-entropy) term as an alternative to the traditional softmax loss function. In [21], the authors introduced a multilabeling Bayes-optimal classifier, based on hierarchical extensions of the Hamming and ranking loss functions.

Among such machine learning methods, support vector machines (SVMs) are known to be one of the most promising classification tools [22] [23] [24], especially in the remote sensing field [25] [26]. Moreover, the SVM frameworks have been extended substantially according to the increasing complexity of remote sensing data. One of their great contributions is structure learning [27]. Structured SVM (SSVM) is an efficient classifier for data with output structure. In multilabel classification problems, for instance, "Solar Panel" labels are more likely to coincide with "Roof" labels than "Tree" labels because solar panels obviously require flat places to absorb much sunshine. This kind of output relationship should be built in classifiers to yield more reasonable classification outcomes. The SSVM classifiers achieved this request with the straightforward extension of the normal SVM learning. SSVM has already been applied to multi-class classification in various fields including remote sensing, in which they have reported the promising capability of the structure learning [28] [29] [30].

The other aspect to be considered, especially in pixel- and tile-wise land cover classification, is spatial information. It is pretty natural assumption that each tile gives more similar prediction with its neighborhood than any other randomly picked tiles. The basic classification algorithms cannot embed this principle because they treat instances as independently sampled ones. This neighborhood information is actually useful in remotely sensed images. There exist some approaches which formulate spatial dependency, and they showed significant improvement in classification accuracy with SVM-based space embedding [31]. Spatial contiguity based on the SVM learning is straightforward because the formulation still keeps the basics of the normal SVM concepts [32] [33]. Even though the extension is simple, such algorithms can make improvements on pixel-wise classification.

The aim of this chapter is to propose a novel SVM-based multilabel classification approach which simultaneously embeds the above two properties, namely output structure and spatial contiguity, to achieve accurate land cover classification for EHR remote sensing images. First, we incorporate the output structure by following a SSVM multilabel formulation. Second, we additionally embed spatial contextual information into the SSVM formulation. From a mathematical viewpoint, we add a term to lessen neighborhood discrepancy into the SSVM primal objective function. As is the same with the previous spatial embedding approaches for binary SVM cases as shown in [32] [33], the basic SVM principles are still kept, which means we can optimize

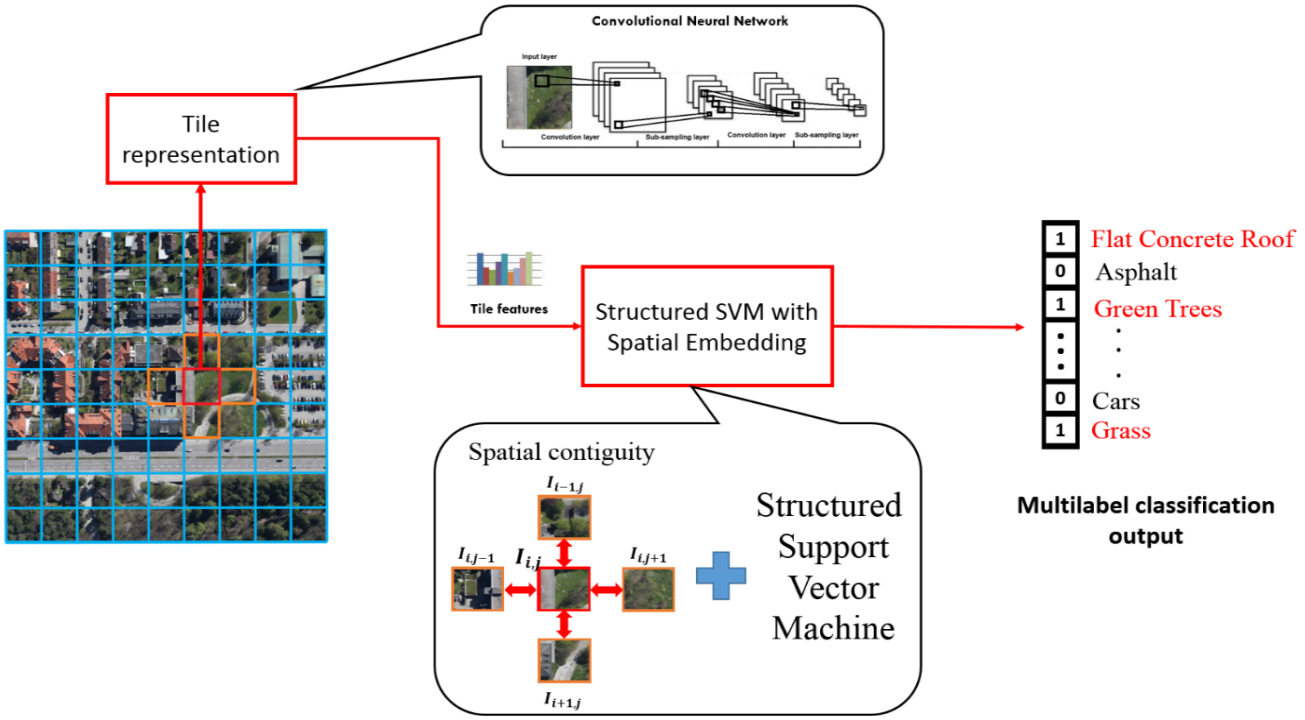


Fig.5.1. General block diagram of the proposed framework for multilabel tile-wise description.

the classifier by solving the standard quadratic programming problem of the dual. As a result, the method makes it possible to incorporate output structure and spatial information simultaneously with ease. We conducted experiments on two types of remotely sensed data, UAV and airborne imagery. The results show the proposed SVM model outperforms reference SVM-based approaches, namely multilabel SVM and its structured version, in terms of classification accuracy.The remainder of this chapter is organized as follows. Section 2 introduces the basics of multilabel classification for EHR images and SSVM. In section 3, we propose our new model. Section 4 reports the experimental results. We draw final conclusion in the section 5.

5.2. Problem Formulation and Tools

5.2.1. Multilabel Classification for EHR Imagery

Let us consider EHR images in which all the pixels are characterized by visible RGB channels. As mentioned before, EHR images contain quite abundant information that typical land classification strategies cannot deal with. As an alternative, a new strategy called coarse description of images was recently proposed [15]. Following this strategy, we first subdivide images into a grid of tiles of equal sizes as shown in Fig. 5.1. Each tile contains a set of RGB information for the pixels and the corresponding binary descriptor represents the presence and absence of target labels. This formulation results in a multilabel supervised classification problem, that is, our goal is to make a prediction as a list of existing objects for every tile in unlabeled images. In contrast to classical multi-class classification, the tiles can hold multiple classes simultaneously. In general, the size of the tiles should be defined according to the spatial resolution

of the image and the expected sizes of objects that one aims at recognizing.The tile-wise multilabel classification is composed of two main steps. The overall process is shown in Fig. 5.1.

The first step concerns suitable tile representation in order to extract discriminative features for classification. Our approach makes use of a deep learning technique, i.e., convolutional neural networks (CNNs), as a feature extractor. They have definitely drawn a major outgrowth as compared to traditional handcrafted feature extraction means and have shown great effectiveness in various computer vision applications [34]. After extracting features, we pass them to the second step, classification process, which represents the focus of this chapter. The essential aspects here are to develop an efficient classifier with the following two properties: output structure and spatial contiguity. As shown in Fig. 5.1, for instance, 'Car' labels are likely to hold the 'Asphalt' label simultaneously. Also, we can readily imagine that tiles with the 'Asphalt' label range in consecutive areas. Any model embedding these two properties has a huge potential of improving the recognition accuracy, which is our primal motivation in this chapter. To the best of our knowledge, there is no work in the literature which merges them simultaneously for multilabel classification. In the following subsections, we describe briefly the CNN pre-trained model adopted for the tile representation step, then we elaborate details outlining how to embed the two properties into the SVM-based classification.

5.2.2. Tile Representation

Departing from the need to extract discriminative features as the first step in our multilabel classification approach, as mentioned in the previous section. This tile representation model is inspired by deep neural networks, precisely CNNs, which are learning paradigms that map features from input data in a hierarchical way, resulting in multiple levels of pattern representations formed by a composition of low-level features. The CNN-based feature learning has successfully provided an effective alternative to traditional handcrafted features [35] [36] [37]. It is composed of a set of convolution and subsampling operations, followed by a classifier. The convolution layer consists of a set of filters convolved repeatedly through spatial sliding over images to generate what is called the feature maps, followed by the pooling layer that reduces the size of the obtained features. Further operations can be performed to the resulting outputs such as dropout, batch normalization, and elementwise activation function layer (i.e., Rectified Linear Units, ReLU) that can be applied to increase the sparseness of the network and better deal with the overfitting risk. The propagated features come to the fully connected classification layer from the previous layers to make final outputs.

In this chapter, we resort to the publicly available pre-trained CNN named GoogLeNet. The network was originally trained over the ImageNet ILSVRC2014 dataset which contains over 1.2 million images with 1000 categories and showed a promising capability for image recognition. Its process contains a 27-layer deep network including pooling layers, with a softmax loss layer as a classifier. The size of the receptive field is 224×224 of three RGB channels with zero mean. The ReLU operation is used in all its convolution layers. It finally generates discriminative feature vectors of the size equal to 1024. The most important component characterizing the GoogLeNet network is what is called inception modules, which are modules with a wise local sparse structure

of dense components (e.g., convolution, pooling, and softmax). They cluster the correlation statistics of the previous layer output into a group of units. The major benefit of these inception layers is their efficient reduction of dimensionality as well as computational requirements. GoogLeNet is based on nine inception modules. The width of inception modules ranges from 256 filters (in early modules) to 1024 in top inception modules. After the subdivision of our image into a non-overlapping grid of equal tiles of size 50×50 pixels as explained in Section 2.A., we resize the image tiles for each RGB channel into the size of the pre-trained GoogLeNet receptive field (i.e., 224×224). Afterwards, the features are extracted by propagating each tile throughout the network up to its fully connected layer generating a feature vector of length 1024 for each tile. The following section details the proposed SVM-based multilabel classifier.

5.2.3. Reviews of Structured SVM

We first review the background of SSVM for multilabel classification [27]. Let us define a training set as $\{(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{X} \times \mathcal{Y} | i = 1, \dots, n\}$, where $\mathcal{X}$ and $\mathcal{Y}$ are the input and output spaces, respectively. In ordinary multilabel classification, we define them as $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = \{0, 1\}^c$, where c represents the number of total classes. Each element of the output vector $\boldsymbol{y}_i$ expresses the presence and absence of class $k \in \{1, \dots, c\}$; the kth element of $\boldsymbol{y}_i$ is 1 if the sample $\boldsymbol{x}_i$ holds the class k label and 0 otherwise. The cardinality of the output space is 2^c because samples can belong to multiple classes. The basic concepts of multilabel structure learning is to make predictions for samples according to the rule defined by the function $g: \mathcal{X} \to \mathcal{Y}$ such that

$$g(\boldsymbol{x}) = \underset{\boldsymbol{y} \in \mathcal{Y}}{\operatorname{argmax}} f(\boldsymbol{x}, \boldsymbol{y}). \tag{1}$$

The discriminant function $f: \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ defines scores for the input-output pairs, which takes the inner product with the weight vector

$$f(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{y}). \tag{2}$$

The joint feature map $\boldsymbol{\psi}: \mathcal{X} \times \mathcal{Y} \to \mathcal{H}$ maps the input-output pairs jointly into a feature space $\mathcal{H}$. Flexible designs of the joint feature vector make it possible to incorporate hierarchical structure in pattern analysis. In commonly used multilabel classification settings described in [28] [29] [30] the joint feature is defined as

$$\boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{y} \otimes \boldsymbol{\phi}(\boldsymbol{x}), \tag{3}$$

where $\otimes$ is the Kronecker tensor product and $\boldsymbol{\phi}: \mathcal{X} \to \mathbb{R}^p$ is the feature mapping which maps an original d dimensional input $\boldsymbol{x}$ to p dimensional feature space. That is, the joint feature $\boldsymbol{\psi}$ finally constructs the $p \times c$ dimensional joint feature vector.

The weight vector $\boldsymbol{w}$ is trained via the similar optimization with the normal binary SVM: maximizing the margin and minimizing the empirical error [27]. Here we assume a loss function $\Delta(\boldsymbol{y}, \bar{\boldsymbol{y}}) \geq 0$ which quantifies the dissimilarity between labels $\boldsymbol{y}, \bar{\boldsymbol{y}} \in \mathcal{Y}$, where $\Delta(\boldsymbol{y}, \boldsymbol{y}) = 0$. Then, we formulate the following optimization,

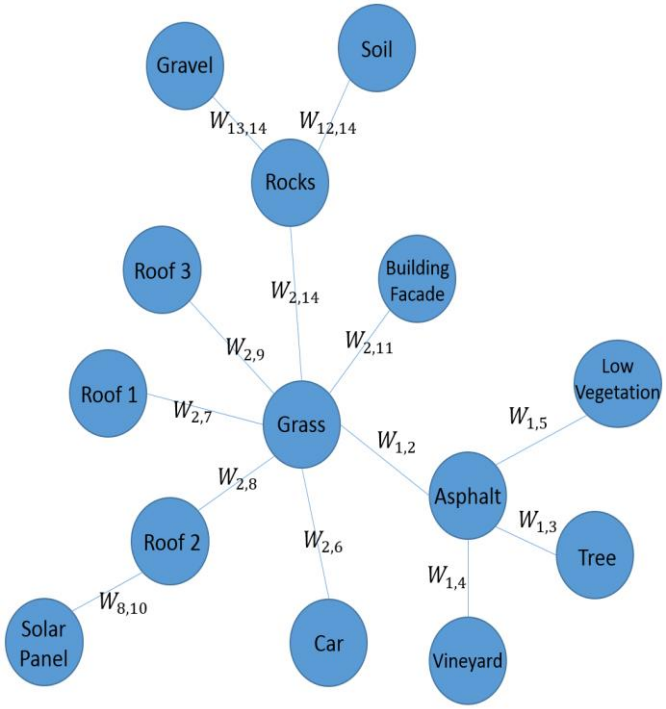


Fig. 5.2. Output graph structure obtained for dataset 1 (Civezzano).

.

$$\min_{\boldsymbol{w},\boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i, \tag{4}$$

$$\text{s.t}\ \forall \overline{\boldsymbol{y}}_i \in \mathcal{Y}, \qquad \boldsymbol{w}^{\mathrm{T}}\{\boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{\psi}(\boldsymbol{x}_i, \overline{\boldsymbol{y}}_i)\} \geq \Delta(\boldsymbol{y}_i, \overline{\boldsymbol{y}}_i) - \xi_i, \quad \text{for } i = 1, \dots, n.$$

This is called n-slack SSVM formulation with margin rescaling. Every constraint insists that the score of the true label must be greater than those of the other labels added their losses. Otherwise, ξ_i imposes a positive penalty. Hence, the weight parameter must be tuned so that the classifier surely separates unlikely events imposed large losses to minimize the objective function. In this way, we can avoid coincidence of unlikely classes.

The practical iterative optimization method based on the cutting-plane algorithm was proposed in [38], in which the n-slack formulation is first replaced with the alternative so-called 1-slack formulation as follows:

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\xi, \tag{5}$$

$$\text{s.t}\ \forall (\overline{\boldsymbol{y}}_1, \dots, \overline{\boldsymbol{y}}_n) \in \mathcal{Y}^n, \qquad \frac{1}{n}\boldsymbol{w}^{\mathrm{T}}\sum_{i=1}^{n}\{\boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{\psi}(\boldsymbol{x}_i, \overline{\boldsymbol{y}}_i)\} \geq \frac{1}{n}\sum_{i=1}^{n}\Delta(\boldsymbol{y}_i, \overline{\boldsymbol{y}}_i) - \xi.$$

This formulation shares all the slack variables by summing up $\xi = \frac{1}{n}\sum \xi_i$. This is efficient because we no longer need to store constraints for each sample individually. The theoretical equivalence between the two formulations and experimental comparison is shown in [38]. The direct optimization of the 1-slack formulation is still not practical because the total number of constraints is $|\mathcal{Y}|^n$. The cutting-plane algorithm, alternatively, includes only the most violated constraint in

each step, then the primal objective function is solved subject to the restricted constraints. The detailed algorithm is described in the next section where we explain our model.

Note that we can still utilize "kernel trick" even in SSVM by defining the kernel for the inner product of the joint feature vectors, i.e., : $(\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \to \mathbb{R}$. Especially in the multilabel classification, it can be derived readily as

$$K(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{x}_j, \boldsymbol{y}_j) = \boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{y}_i)^T \boldsymbol{\psi}(\boldsymbol{x}_j, \boldsymbol{y}_j) = \boldsymbol{y}_i^T \boldsymbol{y}_j \times \boldsymbol{\phi}(\boldsymbol{x}_i)^T \boldsymbol{\phi}(\boldsymbol{x}_j). \quad (6)$$

Once we replace the inner product of the input space with a kernel function K': $\mathcal{X} \times \mathcal{X} \to \mathbb{R}$, we do not need to describe the feature map explicitly.

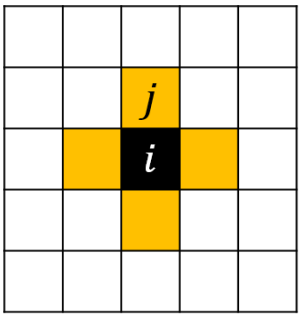


Fig. 5.3. First-order neighborhood system. The black tile is the center i and the yellow tiles are the elements of neighborhood N_i.

In order to embed the output structure into the model, we need to define the structural loss function $\Delta(\boldsymbol{y}, \bar{\boldsymbol{y}})$. In the multi-class structural learning in [28] [30], they built the hierarchical output structure a priori based on obvious facts and expert experiences. The losses among labels are computed according to the distances of meta-labeled outputs defined by the tree structure. If two classes are the branches of different trees, i.e., they are not similar classes in some sense, then the miss-classification penalty increases, and vice versa. The readers can refer [28] [30] for further details. Another approach of embedding output loss structure is data-driven rather than by experience. Chow-Liu algorithm is one of the examples, which construct output graph structure based on mutual information among labels [39]. Figure 5.2 depicts the relationships between labels estimated on dataset 1 (dataset 1 is described below in Experimental Results Section). The nodes of the graph represent the class labels, and each weight $w_{l,m}$ on the edges controls the degree of label-label relationship. For instance, whenever two nodes l, m do not have a correlation, they will not have an edge between them, namely, the edge weight $\alpha_{l,m}$ is equal to zero. We use this algorithm in our experiments in which the weights are also estimated in the training process.

5.2.3. Structured SVM with Spatial Embedding

So far, we reviewed our land cover classification strategy and the basic of the structural learning. Modifying the SSVM, in this section, we present a new SSVM-based algorithm for effective multilabel land cover classification. In the tile-wise image classification, neighbor tiles very likely tend to share same labels. To embed this property, we propose adding an extra term to

penalize the dissimilarity among neighbor tiles into the SSVM objective function. First, we focus on a center tile i and its neighbors $j \in N_i$, where N_i is the set of the tile indexes of the tile i neighborhood. Fig. 5.3 shows an example of the first-order neighborhood system: the closest four tiles on up and down, left and right. The sum of the score distances between the center tile and its neighbors,

$$\sum_{j \in N_i} \left\{ \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}_j, \boldsymbol{y}_j) \right\}^2, \tag{7}$$

should be minimized in the training process to reduce un-smoothness on images. A similar formulation has been proposed in the binary SVM classification [32] [33]. Here, we move to the SSVM framework and we embed spatial contiguity into it. Note that we have to do it on test (unknown) images, which do not have ground truth labels. Hence, we need to lessen the spatial label discrepancy among neighborhood possibly for any label patterns.

In multilabel classification, the weight $\boldsymbol{w}$ consists of the series of the weights associated with all classes, namely, $\boldsymbol{w} = (\boldsymbol{w}_1^T, \dots, \boldsymbol{w}_c^T)^T$. The individual score of class k for an instance $\boldsymbol{x}_i$, $\boldsymbol{w}_k^T \boldsymbol{\phi}(\boldsymbol{x}_i)$, can be rewritten by $\boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{e}_k)$ using the joint feature, where $\boldsymbol{e}_k$ is the unit vector and takes one in the kth element. The sum of distances of all the k scores $\boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{e}_k)$ for $k \in \{1, \dots, c\}$ derives the following formula:

$$\sum_{k=1}^{c} \sum_{j \in N_i} \left\{ \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{e}_k) - \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}_j, \boldsymbol{e}_k) \right\}^2. \tag{8}$$

minimizing the above quantity forces adjacent tiles to have similar scores not only on respective classes but also on every label pattern in the output space $\mathcal{Y}$ because they all are the combination of class scores, as long as we use the joint feature mapping defined in (3). In this way, the additional term encourages neighbor tiles even to have same predictions. We define an adjacent matrix B such that $B_{i,j} = 1$ if the tile i and j are in the neighborhood and $B_{i,j} = 0$ otherwise. The total distance of the adjacent tiles on the whole image can be written by

$$\begin{aligned} \sum_{k=1}^{c} \sum_{i,j=1}^{N} B_{i,j} \left\{ \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{e}_k) - \boldsymbol{w}^T \boldsymbol{\psi}(\boldsymbol{x}_j, \boldsymbol{e}_k) \right\}^2 \\ = 2\boldsymbol{w}^T \left\{ \sum_{k=1}^{c} \boldsymbol{\Psi}_k^T (D - B) \boldsymbol{\Psi}_k \right\} \boldsymbol{w}. \end{aligned} \tag{9}$$

Here, N $(> n)$ is the total number of samples combined training and test data, D is the diagonal matrix whose ith element $D_{i,i} = \sum_{j=1}^{N} B_{i,j}$, and $\boldsymbol{\Psi}_k = \left(\boldsymbol{\psi}(\boldsymbol{x}_1, \boldsymbol{e}_k), \dots, \boldsymbol{\psi}(\boldsymbol{x}_N, \boldsymbol{e}_k) \right)^T$ is the joint feature matrix corresponding to class k. Once we define the matrix $\boldsymbol{\Sigma}$ as

$$\boldsymbol{\Sigma} = \sum_{k=1}^{c} \boldsymbol{\Psi}_k^T (D - B) \boldsymbol{\Psi}_k, \tag{10}$$

then we merge it with the SSVM primal objective function as follows:

$$\min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T(\boldsymbol{I}+\lambda\boldsymbol{\Sigma})\boldsymbol{w}+C\xi, \tag{11}$$
$$\text{s.t}\ \forall(\bar{\boldsymbol{y}}_1,\dots,\bar{\boldsymbol{y}}_n)\in\mathcal{Y}^n,\quad \frac{1}{n}\boldsymbol{w}^{\mathrm{T}}\sum_{i=1}^{n}\{\boldsymbol{\psi}(\boldsymbol{x}_i,\boldsymbol{y}_i)-\boldsymbol{\psi}(\boldsymbol{x}_i,\bar{\boldsymbol{y}}_i)\}\geq\frac{1}{n}\sum_{i=1}^{n}\Delta(\boldsymbol{y}_i,\bar{\boldsymbol{y}}_i)-\xi.$$

The positive constant λ is the trade-off parameter between the model accuracy and smoothness. We can describe the whole algorithm to optimize the weight parameter in the same way with the original cutting-plane algorithm for the structure learning [38]. Algorithm 1 shows the related pseudo code.

Algorithm 1: Cutting-plane algorithm for 1-slack formulation with spatial information

1. Input: $\{(\boldsymbol{x}_i,\boldsymbol{y}_i)\}_{i=1,\dots,n}$, $\boldsymbol{\Sigma}, C, \lambda, \varepsilon, \mathcal{W}\leftarrow\emptyset$.
2. **Repeat**
3. $\boldsymbol{w},\xi\leftarrow\text{argmin}_{\boldsymbol{w},\xi}\ \frac{1}{2}\boldsymbol{w}^T(\boldsymbol{I}+\lambda\boldsymbol{\Sigma})\boldsymbol{w}+C\xi$ s.t $\forall(\bar{\boldsymbol{y}}_1,\dots,\bar{\boldsymbol{y}}_n)\in\mathcal{W}$, $\frac{1}{n}\boldsymbol{w}^{\mathrm{T}}\sum_{i=1}^{n}\{\boldsymbol{\psi}(\boldsymbol{x}_i,\boldsymbol{y}_i)-\boldsymbol{\psi}(\boldsymbol{x}_i,\bar{\boldsymbol{y}}_i)\}\geq\frac{1}{n}\sum_{i=1}^{n}\Delta(\boldsymbol{y}_i,\bar{\boldsymbol{y}}_i)-\xi$.
4. **for** $i=1,\dots,\mathrm{n}$ **do**
5. $\bar{\boldsymbol{y}}_i\leftarrow\text{argmax}_{\bar{\boldsymbol{y}}\in\mathcal{Y}}\{\Delta(\boldsymbol{y}_i,\bar{\boldsymbol{y}})+\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\psi}(\boldsymbol{x}_i,\bar{\boldsymbol{y}})\}$
6. **end for**
7. $\mathcal{W}\leftarrow\mathcal{W}\cup\{(\bar{\boldsymbol{y}}_1,\dots,\bar{\boldsymbol{y}}_n)\}$
8. **until** all the constraints are fulfilled with the tolerance rate ε

The algorithm starts with a null constraint $\mathcal{W}=\emptyset$, where $\mathcal{W}$ is called the working set. Each step searches for the most violated constraint under the current parameters and adds it to the working set iteratively until the algorithm terminates. The most violated constraint is found in Steps 4-6. The only difference from the original algorithm is in Step 3; we newly add the term $\lambda\boldsymbol{\Sigma}$ for the spatial contiguity. Since the primal objective function still keeps the same principle of SSVM, we can derive the quadratic programming dual problem of the primal as:

$$\max_{\boldsymbol{\alpha}}\sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^n}\alpha_{\bar{\boldsymbol{y}}}\left\{\frac{1}{n}\sum_{i=1}^{n}\Delta(\boldsymbol{y}_i,\bar{\boldsymbol{y}}_i)\right\}-\frac{1}{2}\sum_{\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'\in\mathcal{Y}^n}\alpha_{\bar{\boldsymbol{y}}}\alpha_{\bar{\boldsymbol{y}}'}H(\bar{\boldsymbol{y}},\bar{\boldsymbol{y}}'), \tag{12}$$
$$\text{s.t}\sum_{\bar{\boldsymbol{y}}\in\mathcal{Y}^n}\alpha_{\bar{\boldsymbol{y}}}=C,$$

where

$$H(\overline{\boldsymbol{y}}, \overline{\boldsymbol{y}}') = \frac{1}{n^2}\left[\sum_{i=1}^{n}\{\boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{\psi}(\boldsymbol{x}_i, \overline{\boldsymbol{y}}_i)\}\right]^T (\boldsymbol{I} + \lambda\boldsymbol{\Sigma})^{-1}\left[\sum_{i=1}^{n}\{\boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{\psi}(\boldsymbol{x}_i, \overline{\boldsymbol{y}}_i')\}\right]. \quad (13)$$

Each $\alpha_{\overline{\boldsymbol{y}}}$ is the Lagrange multiplier associated with one of the constraints in the working set. After we solve the dual problem, we pass to the weight vector,

$$\boldsymbol{w} = (\boldsymbol{I} + \lambda\boldsymbol{\Sigma})^{-1} \sum_{\overline{\boldsymbol{y}} \in \mathcal{Y}^n} \alpha_{\overline{\boldsymbol{y}}}\left[\frac{1}{n}\sum_{i=1}^{n}\{\boldsymbol{\psi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{\psi}(\boldsymbol{x}_i, \overline{\boldsymbol{y}}_i)\}\right], \quad (14)$$

as the primal solution. The weight parameter is optimized so as to find separable hyperplane for the training samples as well as simultaneously enhance smoothness in entire images. We call this algorithm *Spatial Structured SVM* (SSSVM).

5.3. Experimental Results

We conducted numerical experiments to assess the classification accuracy of the proposed method on two real datasets composed of UAV and airborne images acquired over two different locations.

5.3.1. Dataset Description

1) Dataset 1: The first set of images is a UAV dataset which had been acquired near the city of Civezzano (Italy) at different off-nadir angles on October 17, 2012 [17]. The data acquisition was performed with a picture camera Canon EOS 550D characterized by a CMOS APS-C sensor with 18 megapixels. The images are characterized by three channels (RGB) with a spatial resolution of approximately 2 cm. The obtained images are of size 5184×3456 pixels with 8-bit radiometric resolution. This dataset is composed of ten images. All the images are subdivided into a non-overlapping grid of equal tiles of 50×50 pixels as explained in section 2.1. We selected three learning images in such a way they contain all predefined 14 classes of objects, which are 'Asphalt', 'Grass', 'Tree', 'Vineyard', 'Low Vegetation', 'Car', 'Roof 1', 'Roof 2', 'Roof 3', 'Solar Panel', 'Building Facade', ' Soil', 'Gravel', and 'Rocks'. We divided the 'Roof' category into three different classes due to its large heterogeneity. Furthermore, we randomly subdivided the learning set into two parts, training and validation sets, the former for training the model and the latter for estimating related hyperparameters. The final sizes of the training, validation, and test sets are 5000, 16321, and 49749 respectively. Table 5.1 lists all the class occurrences in the data.

2) *Dataset 2*: The second dataset was originally acquired for the sake of vehicle detection over Munich, Germany [40], but here we use it for land use classification. The images were captured from an airplane by a Canon Eos 1Ds Mark III camera with a resolution of 5616×3744 pixels, 50 mm focal length. The images were taken at a height of 1000 meters above ground, with a spatial resolution of approximately 13 cm. This dataset is composed of 20 images and contain all predefined 16 classes of objects, which are 'Slate Roof', 'Clay Roof', 'Asphalt', 'Standing Steam Roof', 'Sports Field', 'Green Trees', 'Dried Trees', 'Grass', 'Building Site/Operating

Table 5.1. Number of class occurrence on dataset 1 (Civezzano).

Class		Tiles		
No	Name	Training	Validation	Test
		5000	16321	49749
1	Asphalt	1195	3957	13280
2	Grass	2082	6906	18508
3	Tree	886	2843	8790
4	Vineyard	58	182	1228
5	Low Vegetation	374	1110	4354
6	Car	215	610	1458
7	Roof1	239	772	1728
8	Roof2	218	657	1254
9	Roof3	121	341	2833
10	Solar Panel	98	295	456
11	Building Facade	217	706	1224
12	Soil	205	755	1957
13	Gravel	17	58	491
14	Rocks	156	538	731

Table 5.2. Number of class occurrence on dataset 2 (Munich).

Class		Tiles		
No	Name	Training	Validation	Test
		5000	77880	82880
1	Slate Roof	296	4394	9836
2	Clay Roof	285	4046	6069
3	Asphalt	1344	21389	31366
4	Standing Seam Roof	224	3378	2786
5	Sports Field	13	234	383
6	Green Trees	1487	24105	32983
7	Dried Trees	685	11112	7466
8	Grass	1797	27697	25737
9	Building Site/Operating Machinery	19	408	275
10	Paved Road	162	2367	968
11	Soil	336	5107	3649
12	Dark Soil	243	3571	1563
13	Green Roof	130	2002	2594
14	Flat Concrete Roof	231	3955	4773
15	Railway	128	2020	299
16	Cars	306	4830	8709

Machinery', 'Paved Road', 'Soil', 'Dark Soil', ' Green Roof', 'Flat Concrete Roof',' Railway', and 'Cars'. We used ten images for test. The rest of images were further divided into the training and validation sets, whose sizes are 5000 and 77880, respectively. The detailed numbers of every set and the class occurrences are provided in Table 5.2.

5.3.2. Experimental Setup

We compared our SSSVM model with two closely related reference approaches, namely the standard SVM and the structured SVM (SSVM). The SSVM was implemented by "pystruct" python module [41]. The standard SVM can be viewed as a special case of SSVM, simply by implementing it without giving any structure in the module. Doing in this way, one can solve the multi-class classification problem directly without dividing the output space as done in the so-called One-Against-All classification. For SSVM and SSSVM, we first built the output structure on the basis of Chow-Liu algorithm using the output labels in their training sets. We tuned the regularization parameter C according to validation errors using the following set of values $\{2^{-15}, \dots, 2^{-3}\}$. In SSSVM models, we first tuned C based on the validation errors of SSVM models, then selected the best λ in $\{2^{-9}, \dots, 2^{2}\}$ according to the minimum validation error.

In all the experiments, we adopted the simple linear kernel, which does not ask for additional parameter except for λ. The neighborhood system used is the first-order neighborhood (Fig. 3). We assessed the classification accuracy with the specificity, sensitivity, and their average metrics. We also computed the Hamming loss between the ground truth and the prediction. The loss is given as the mean losses of all the samples, namely, lies in $[0, c]$, where c is the number of total classes. Moreover, we conducted the McNemar's statistical test to assess the pairwise significance of differences between two classifiers. I consists in the calculation of [42]:

$$Z_{ij} = \frac{f_{ij} - f_{ji}}{\sqrt{f_{ij} + f_{ji}}}, \tag{15}$$

where f_{ij} indicates the number of samples correctly classified by the ith classifier and wrongly classified by the jth classifier. At the commonly used 5 % significant level, the difference in accuracy of two classifiers is said to be statistically significant if $|Z_{ij}| > 1.96$. The signs of Z_{ij} stand for the goodness of the two; $Z_{ij} > 0$ means the ith classifier is more accurate than the jth classifier and vice versa.

5.3.3. Experimental Results

5.3.3.1. Results of Dataset 1

Table 5.3 shows the quantitative experimental results achieved on dataset 1. The SSVM performs better than SVM, and both are outperformed by our proposed SSSVM model. Indeed, the average accuracy of SSVM and SSSVM are 83.82 % and 84.12 %, respectively. Although the less than 1% gain may sound tiny, the McNemar's test shows this improvement is statistically significant (see. Table 5.4). Such improvement can also be confirmed by analyzing the classification maps (see Fig. 5.4). Fig. 5.4(a) represents the ground truth image of test image 10. The tiles of Fig. 5.4(b-d) are colored according to the Hamming distances between the ground truth labels and the predicted labels obtained by the three classifiers. The map based on SVM in Fig. 5.4(b) is apparently bright, which indicates many of the tiles were wrongly predicted. On the other hand, the darkness of the SSSVM map in Fig. 5.4(d) shows that the classification by the proposed model is seemingly more accurate than what yielded by the SVM and even the SSVM classification. Fig. 5.5 illustrates the classification maps generated by SSSVM for test image 10. It is noteworthy that in multilabel classification, one gets as many classification maps as the total number of classes. In general, the achieved results are good, even though, as expected, there are some confusion in discriminating between classes with similar visual appearance (e.g., 'Grass' and 'Trees'), besides some false positives which emerge due to the intrinsic class variability especially when dealing with UAV imagery and a considerable number of classes simultaneously. However, in overall terms, the classification maps exhibit a satisfactory description of the scene. In greater detail, Table 5.7 compares the class accuracies of the three classifiers. Since the overall sensitivity of SVM is high and its specificity is low, SVM classifier tends to overestimate present labels. This gives it an advantage in the classification of the large classes such as class 1 ("Asphalt") and class 2 ("Grass") as shown in Table 5.7. Compared with SVM and SSVM, SSSVM produce a higher accuracy in terms of specificity. For the dominant classes (with large samples) such as classes 1,

Table 5.3. Classification accuracies of the three classifiers on dataset 1 (Civezzano).

	Sensitivity	Specificity	Average	Hamming
SVM	88.20	73.04	80.62	3.597
SSVM	78.18	89.47	83.82	1.607
SSSVM	76.69	91.57	**84.12**	**1.355**

Table 5.4. Statistical comparison based on MCNEMAR's test between the three models on dataset 1(Civezzano).

	SVM	SSVM
SSVM	-293.5	
SSSVM	-313.6	-81.1

2, and 3, their average rates are comparable. On the other hand, SSSVM achieves a more accurate classification than SSVM for medium size classes (intermediate sizes of samples). For small classes in the training set such as class 4 ("Vineyard") and class 13 ("Gravel"), the SSVM classifier is a bit more accurate than SSSVM. This can be explained by the fact that the spatial contextual information tend to smooth out isolated labels or small structures. However, in general, SSSVM model succeeds in improving both accuracy and Hamming loss of SSVM.

Regarding the output structure, based on the structure estimated by the Chow-Liu Algorithm, the "Asphalt" class has connection with the following four classes: "Grass", "Tree", "Vineyard", and "Low Vegetation" as shown in Figure 5.2. That is to say, the estimated classifier can increase or reduce the penalties for the coincidence with each of them. From the results, the penalty for the coincidence of "Asphalt" and "Vineyard" was the biggest of the four. This is indeed reasonable because there are no samples in the training set which share both labels. The training process was able to distinguish unlikely events automatically.

There are three aspects that deserve to be mentioned. First, many Average criterion values of SVM are greater than those of SSVM and SSSVM. This is caused by the low cardinality of the dataset, where the cardinality represents the number of labels each instance holds on average. Indeed, the cardinality of the dataset is 1.17. In this context, even though SVM overestimates false positives, Specificity is not affected as much as Sensitivity, because true negatives still largely exist. That is why the classification performances of SVM look nice at a glance. Even in such situation, however, structured models are superior to SVM and the Hamming distances support their superiority as well. Second, structured models behave so as to reduce wrong label co-occurrences in its predictions, thanks to the exploitation of the inter-label correlation.

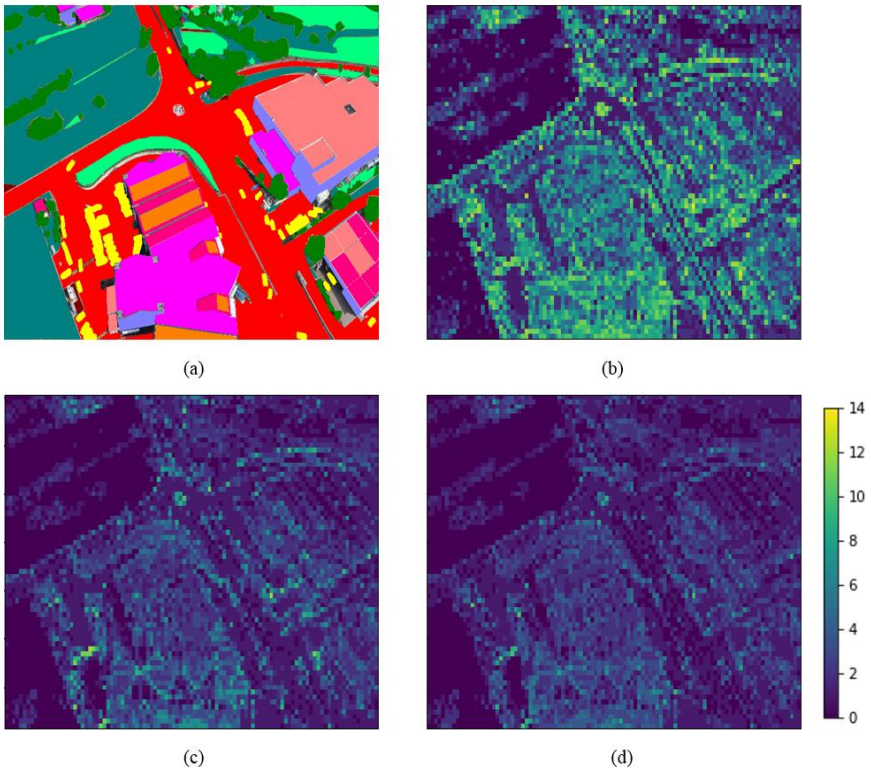


Fig. 5.4. (a) Ground truth of a test image of dataset 1 (Civezzano). (b, c, d) Classification results of the test image obtained by SVM(b),SSVM(c) and SSSVM(d). Each tile is colored according to the Hamming distance between the ground truth and prediction.

This suggests that the SSVM and SSSVM classifiers can learn that the cardinality of the dataset is low. This effect can be confirmed by the estimated weights of the graph edges. The cardinality of predictions of SSVM and SSSVM were low, i.e., 2.27 and 1.98 respectively. Third, as the downside of SSSVM compared with SSVM, spatial embedding of SSSVM sometimes works in a wrong way on boundaries of two objects, for which it is typically harder to provide discriminative features. As a result, SSSVM sometimes underestimates the presence of labels on boundaries. Globally however, as the tables show, utilizing spatial information typically allows generating more reasonable outcomes.

We also analyzed the influence that the value of spatial parameter λ has on the classification outcomes. Fig. 5.7(a) shows the behavior of the average accuracy against λ. In the range from $\lambda = 2^{-4}$ to 2^{0} except for 2^{-3}, the accuracy exceeds the baseline of the SSVM classification accuracy. The larger the λ, the larger the spatial smoothness. This may occasionally incur in worse classification, but the analysis confirms that the SSSVM accuracy is relatively stable in a wide range of λ values.

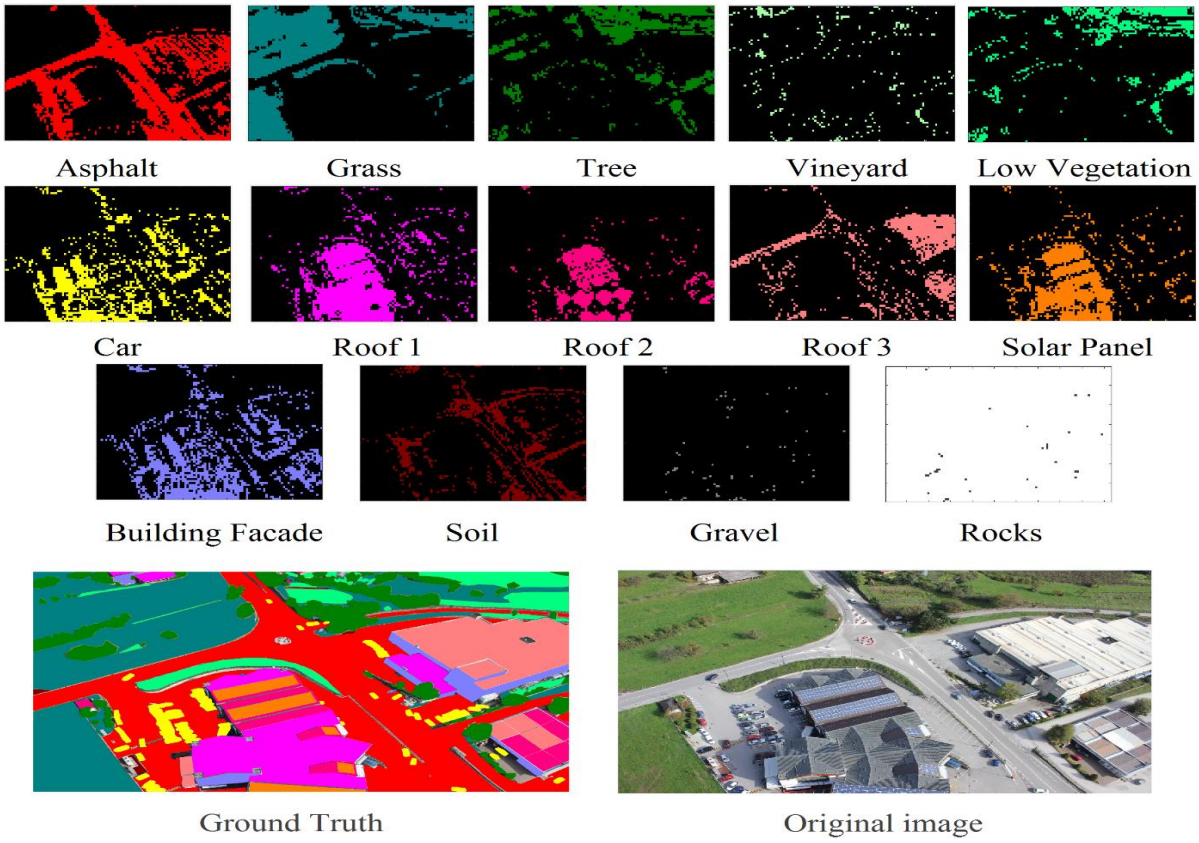


Fig. 5.5. Multilabel classification map obtained by the SSSVM classifier on one of the test images of dataset 1 (Civezzano), along with its related ground truth and original image.

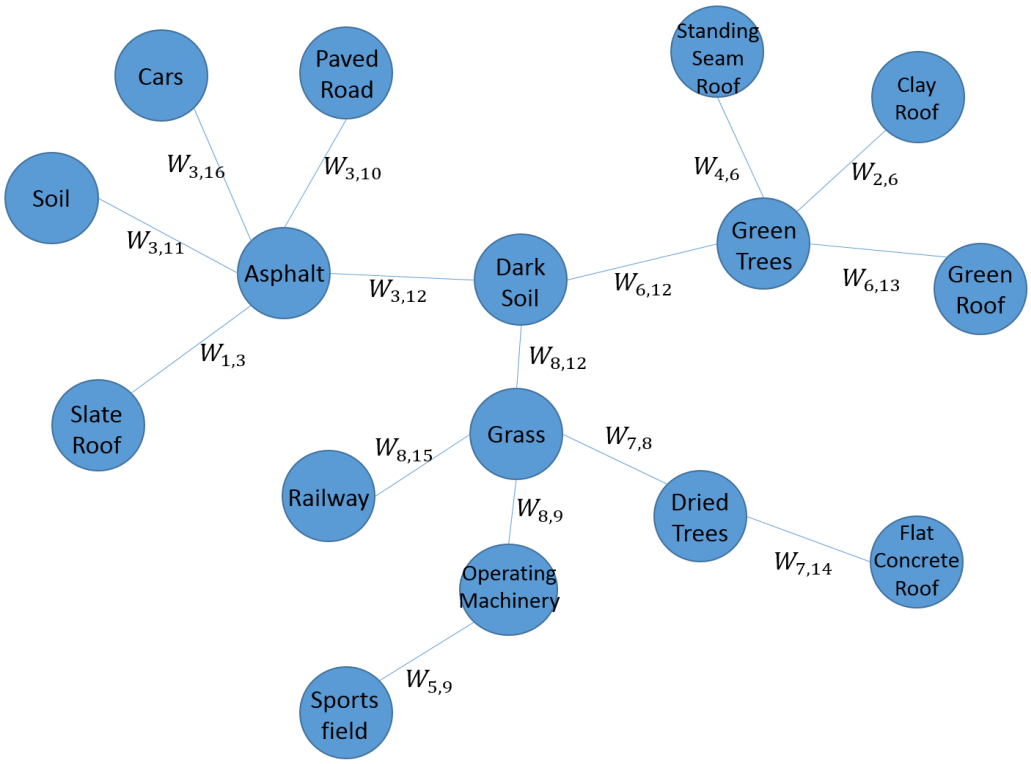


Fig. 5.6. Output graph structure obtained for dataset 2 (Munich).

5.3.3.2. Results of Dataset 2

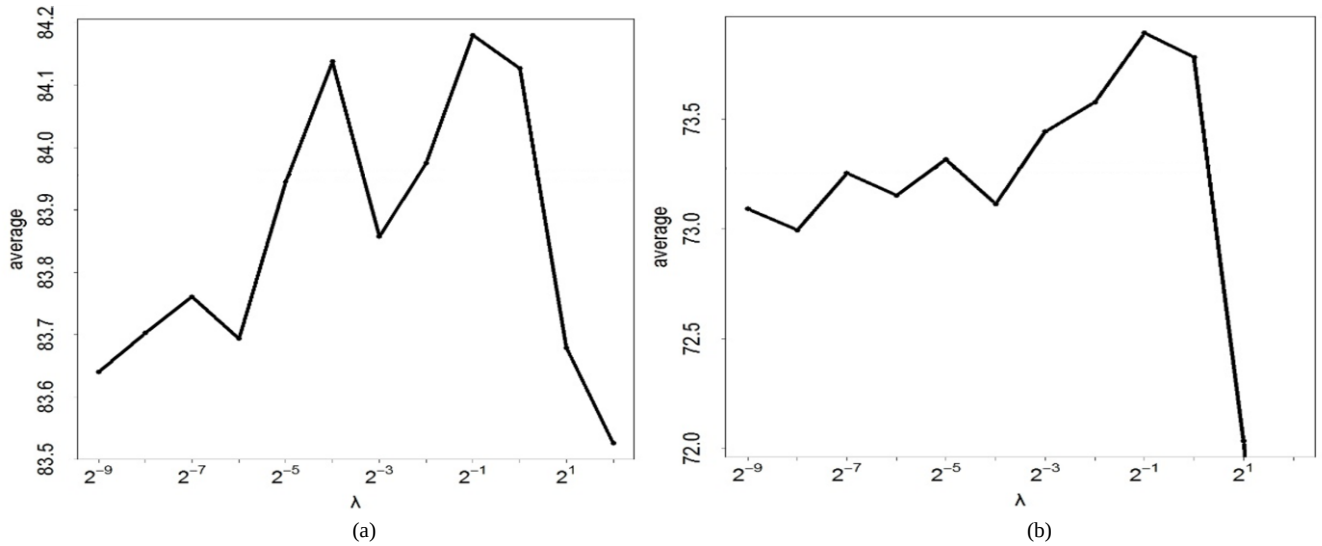
On dataset 2, the proposed SSSVM also outperforms SVM and SSVM. Table 5.5 lists the classification accuracies. The average accuracy without the structure is 71.83%. The structured models, SSVM and SSSVM, get approximately 1.5% and 2.0% gains of accuracy. The inferred structure for these models is given in Fig. 6. Thanks to the exploitation of spatial information, SSSVM performs better than the other two methods. Its average metric and Hamming loss are 73.78% and 2.67 respectively. According to the McNemar's test shown in Table 5.6, SSSVM significantly improves the accuracy over SVM and SSVM. The result suggests output structure with spatial modeling has potentials of making classification more reliable. This improvement is visually confirmed in Fig. 8(b-d). Fig. 9 illustrates the classification maps of SSSVM obtained on one of the test images. Detailed class accuracies are provided in Table 5.8. The SVM model classifies well the large classes such as classes 3 ("Asphalt"), 6 ("Green Trees") and 8 ("Grass"). The SSSVM classifier handles well medium-size classes, while SSVM tends to preserve small classes such as class 5 ("Sports Field") and class 9 ("Building Site"), which benefit from the captured inter-label relationships in the structured model. Similarly to dataset 1, we also checked the accuracy is not strongly influenced by the values of λ as shown in Fig. 7(b).

Table 5.5. Classification accuracies of the three classifiers on dataset 2 (Munich).

	Sensitivity	Specificity	Average	Hamming
SVM	79.17	64.50	71.83	5.43
SSVM	67.34	79.21	73.28	3.53
SSSVM	61.74	85.82	**73.78**	**2.67**

Table 5.6. Statistical comparison based on MCNEMAR's test between the three models on dataset 2 (Munich).

	SVM	SSVM
SSVM	-360.6	
SSSVM	-420.7	-195.6



(a) (b)

Fig. 5.7. Average accuracy versus spatial parameter λ (of SSSVM model) for dataset 1 (Civezzano) (a) and dataset 2 (Munich) (b).

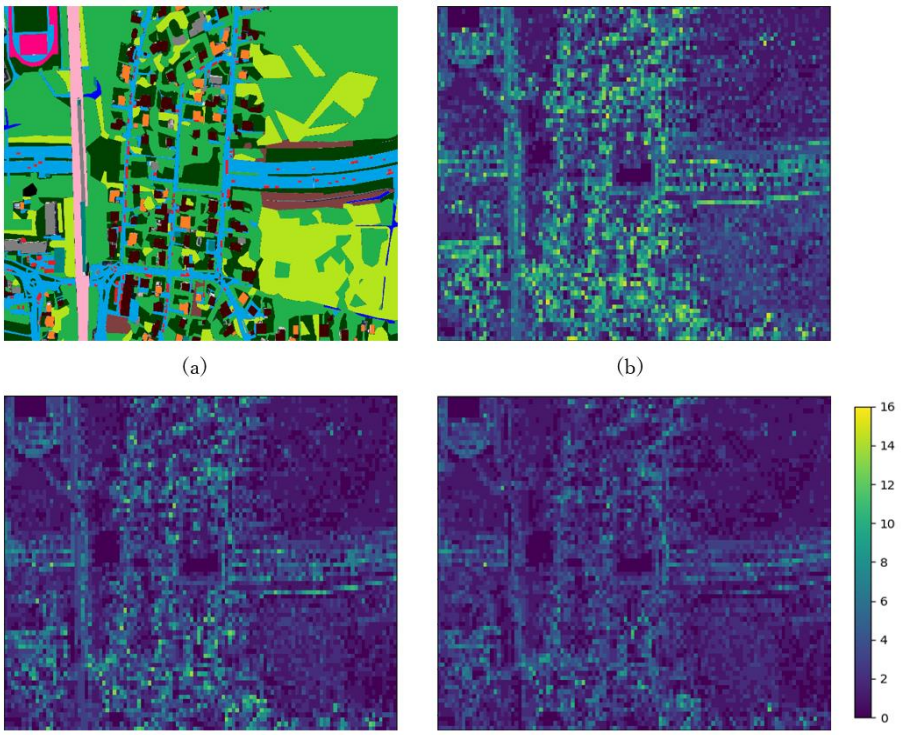


Fig. 5.8. (a) Ground truth of a test image of dataset 2 (Munich). (b, c, d) Classification results of the test image obtained by SVM(b), SSVM(c) and SSSVM(d). Each tile is colored according to the Hamming distance between the ground truth and prediction.

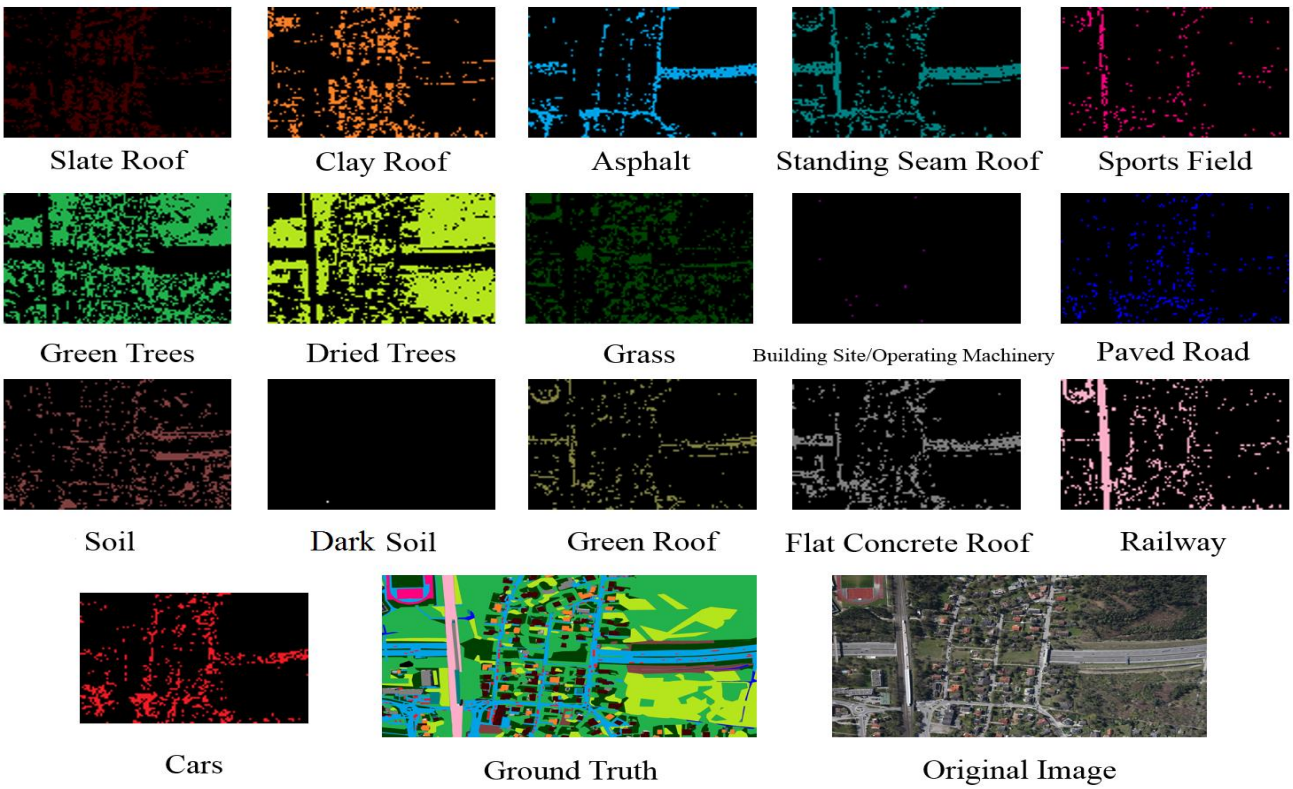


Fig. 5.9. Multilabel classification map obtained by the SSSVM classifier on one of the test images of dataset 2 (Munich), along with its related ground truth and original image.

Table 5.7. class-by-class accuracy performances achieved by the three models on dataset 1 (Civezzano).

	class	1	2	3	4	5	6	7	8	9	10	11	12	13	14
SVM	Sens	86.6	87.4	89.2	82.2	86.5	94.0	83.9	88.2	96.3	99.1	87.5	96.0	73.7	94.0
	Spec	85.4	89.2	78.5	63.7	65.0	79.6	80.0	74.9	71.9	74.6	71.2	61.5	65.3	70.9
	Ave	**86.0**	**88.3**	83.9	**72.9**	**75.7**	86.8	81.9	**81.5**	84.1	86.9	**79.4**	78.8	**69.5**	**82.4**
SSVM	Sens	76.2	83.3	85.4	44.7	67.4	87.7	77.4	66.1	79.6	97.1	73.3	82.5	18.1	40.2
	Spec	89.0	92.0	82.6	91.2	82.5	89.2	90.2	95.9	87.5	89.3	84.0	83.8	95.5	98.7
	Ave	82.6	87.7	84.0	67.9	74.9	88.4	83.8	81.0	83.6	93.2	78.7	**83.2**	56.8	69.5
SSSVM	Sens	73.3	82.6	88.5	25.4	65.9	90.5	76.9	60.4	80.7	96.1	71.2	78.1	4.3	22.3
	Spec	89.8	92.3	81.8	96.2	83.3	90.0	92.0	97.0	91.7	93.0	87.4	86.1	99.1	99.6
	Ave	81.5	87.5	**85.2**	60.8	74.6	**90.3**	**84.5**	78.7	**86.2**	**94.5**	79.3	82.1	51.7	60.9

Table 5.8. class-by-class accuracy performances achieved by the three models on dataset 2 (Munich).

	class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
SVM	Sens	87.5	94.0	77.8	88.3	58.0	76.0	85.3	74.0	81.8	60.2	87.5	97.8	76.6	77.3	91.3	80.7
	Spec	63.9	70.1	77.9	62.9	60.1	79.7	62.0	76.9	59.9	57.0	56.8	54.8	62.6	62.6	63.2	76.1
	Ave	75.7	82.1	**77.9**	75.6	**59.5**	**77.9**	73.7	**75.5**	**70.9**	58.6	72.2	**76.3**	69.6	69.9	77.3	78.4
SSVM	Sens	81.2	91.4	62.8	83.0	42.3	62.6	71.7	65.3	42.5	45.2	76.7	4.80	71.0	66.4	91.0	76.8
	Spec	72.5	76.7	88.7	72.1	76.2	88.8	77.8	84.7	90.4	74.3	73.5	99.1	72.8	71.9	75.8	79.4
	Ave	76.8	84.0	75.7	77.5	59.2	75.7	74.8	75.0	66.5	**59.8**	75.1	52.0	71.9	69.2	83.4	78.1
SSSVM	Sens	76.5	89.9	52.9	82.4	13.6	55.8	70.2	60.0	6.91	20.5	74.9	0.0	66.3	63.9	90.0	81.3
	Spec	78.0	81.3	92.7	73.8	93.6	91.8	81.1	88.2	99.4	89.5	81.2	100.0	85.1	76.3	85.2	78.4
	Ave	**77.3**	**85.6**	72.8	**78.1**	53.6	73.8	**75.7**	74.1	53.2	55.0	**78.1**	50.0	**75.7**	**70.1**	**87.6**	**79.9**

For the sake of further comparison, we confront these obtained results with two Multilabel Conditional Random Field based Classification frameworks (detailed in chapter 4). The first one is the traditional CRF that takes into consideration the tile-based spatial information of each class separately (termed as ML-CRF). The second method similarly to the SSSVM method exploits both the spatial information between tiles along with the label-label correlation information (termed as Full-ML-CRF).

The main difference between these two methods is that the SSSVM framework incorporates; 1) the spatial contextual information and 2) the labels cross-correlation information along with 3) the classification process within a single cost function (i.e., one-step process). On the other hand, the Full-ML-CRF operates as a two-steps framework instead of a single one, where at first, the classification predictions are generated using an opportune classifier (i.e., SVM), and afterwards the Full-ML-CRF is applied as a post-processing step exploiting both the labels correlation and the spatial information to further improve the yielded classification accuracies. Table 5.9 reports the obtained accuracies.

Table 5.9. Sensitivity (SENS), specificity (SPEC) and average (AVG) accuracies in percent obtained by the different classification methods on datasets 1and 2. Average Hamming distances are also reported.

	Dataset 1 (Civezzano)				Dataset 2 (Munich)			
	Accuracy (%)				Accuracy (%)			
METHOD	**SPEC**	**SENS**	**AVG**	**Hamming**	**SPEC**	**SENS**	**AVG**	**Hamming**
SVM	88.20	73.04	80.62	3.59	79.17	64.50	71.83	5.43
SSVM	78.18	89.47	83.82	1.60	67.34	79.21	73.28	3.53
SSSVM	76.69	**91.57**	84.12	1.35	61.74	**85.82**	73.78	2.67
ML-CRF	**93.59**	71.71	82.65	**1.15**	**91.64**	55.76	73.70	**1.94**
Full-ML-CRF	90.71	78.64	**84.69**	1.43	86.76	62.95	**74.85**	2.51

By analyzing the reported results, one can observe that in terms of average accuracy, the Full-ML-CRF performs better than the reported methods achieving 84.69% and 74.85% with a gain of around 0.5% and 1% over the SSSVM method on dataset 1 and dataset 2 respectively. However, in terms of sensitivity the SSSVM yields higher scores than Full-ML-CRF achieving 91% against 78% in dataset 1, and 85% against 62%, in dataset 2. Indeed, the higher sensitivity (true positives) achieved by the SSSVM comes at the cost of some loss in the specificity (true negatives) compared to the basic SVM and the Full-ML-CRF methods in which this latter displays more balanced scores in terms of both sensitivity and specificity.

Concerning the Hamming loss distances computed between the obtained predictions and the ground truth labels, the ML-CRF achieves the smallest loss scoring 1.15 in the first dataset, and 1.94 in the second dataset. This is explained by the fact that the ML-CRF provides the best specificity accuracies in both datasets taking into consideration the fact that the cardinality (how many labels each sample holds on average) in both datasets are low favoring the object absence (which is the case of most multilabel datasets). For instance, if we set all the prediction outputs labels to zeros, we get 1.17 and 1.68 of Hamming loss distance in dataset 1 and 2 respectively. In overall terms, both methods (i.e., SSSVM and Full-ML-CRF) confirm the usefulness of exploiting both the contextual spatial information and the inter-label correlation information in improving the results of the basic SVM classifier. Another element to be highlighted in the comparison between these two multilabel frameworks is the fact that in the Full-ML-CRF we did not tune the free parameters λ_1 and λ_2 (see chapter 4) and we have set them to one ($\lambda_1 = \lambda_2$=1). By contrast, in the SSSVM method, we have tuned the free parameter λ during the training phase due to its importance in the cost function. This suggests that the Full-ML-CRF model provides very good results without resorting to parameter empirical tuning like the case of the SSSVM model.

5.4. Conclusion

In this chapter, we have coped with the challenging problem of multilabel classification. In particular, the input image, assumed to be of extremely high-resolution, is first subdivided into a grid of tiles. Each tile can hold multiple class labels simultaneously. Because of the image resolution, one can reasonably expect that the tiles convey intrinsic label relationships potentially useful to improve the classification accuracy if suitably exploited. Additionally, it is also to expect that in tile-wise classification adjacent tiles are likely to have similar labels. Starting from these considerations, we propose a new multilabel classification method which can embed the output

structure and spatial contiguity simultaneously. The output structure is embedded by using the conventional structural SVM learning. Then, we integrate a new term to encourage spatial smoothness in the optimization processes, handled as a natural extension of the standard SVM learning. In summary, the method can model the spatial contiguity as well as the output structure. The experimental results on two remotely sensed data sets demonstrate that our method significantly outperforms the conventional approaches. The major drawback of the proposed method is the computational cost since SSVM needs to solve linear programming problems for all samples in every step. To keep such cost contained, we reduced the number of training samples in the experiments. It is noteworthy that kernelization of the SSSVM model is possible using Woodbury matrix identity. Since it causes additional computations, it is however better to create discriminative features and adopt the linear kernel as we did in our experiments.

5.5. REFERENCES

[1] C. J. Tucker, J. R. Townshend and T. E. Goff, "African land-cover classification using satellite data," *Science,* vol. 227, no. 4685, pp. 369-375, 1985.

[2] A. M. Dewan and Y. Yamaguchi, "Land use and land cover change in Greater Dhaka, Bangladesh: Using remote sensing to promote sustainable urbanization," *Applied Geography,* vol. 29, no. 3, pp. 390-401, 2009.

[3] S. Tanaka and R. Nishii, "Nonlinear regression models to identify functional forms of deforestation in East Asia," *IEEE Transactions on Geoscience and Remote Sensing,* vol. 47, no. 8, pp. 2617-2626, 2009.

[4] R. Ke, Z. Li, S. Kim, J. Ash, Z. Cui and Y. Wang, "Real-time bidirectional traffic flow parameter estimation from aerial videos," *IEEE Transactions on Intelligent Transportation Systems,* vol. 18, no. 4, pp. 890-901, 2017.

[5] W. G. Aguilar, M. A. Luna, J. F. Moya, V. Abad, H. Parra and H. Ruiz, "Pedestrian detection for UAVs using cascade classifiers with Meanshift," in *IEEE 11th International Conference on Semantic Computing (ICSC)*, San Diego, CA, 2017.

[6] J. Li, D. H. Ye, T. Chung, M. Kolsch, J. Wachs and C. Bouman, "Multi-target detection and tracking from a single camera in Unmanned Aerial Vehicles (UAVs)," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, 2016.

[7] L. David and A. H. Ballado, "Vegetation indices and textures in object-based weed detection from UAV imagery," in *6th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, Batu Ferringhi, 2016.

[8] H. O. Cruz, M. Eckert, J. M. Meneses and J. F. Martinez, "Precise real-time detection of nonforested areas with UAVs," *IEEE Transactions on Geoscience and Remote Sensing,* vol. 55, no. 2, pp. 632-644, 2017.

[9] M. B. Bejiga, A. Zeggada, A. Nouffidj and . F. Melgani, "A convolutional neural network approach for assisting avalanche search and rescue operations with UAV imagery," *Remote Sensing,* vol. 9, no. 2, 2017.

[10] A. L. Sumalan, D. Popescu and L. Ichim, "Flood evaluation in critical areas by UAV surveillance," in *8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, Ploiesti, 2016.

[11] Y. Gu, Q. Wang and B. Xie, "Multiple kernel sparse representation for airborne LiDAR data classification," *IEEE Transactions on Geoscience and Remote Sensing,* vol. 55, no. 2, pp. 1085-1105, 2017.

[12] S. Mei, J. Ji, J. Hou, X. Li and Q. Du, "Learning sensor-specific spatial-spectral features of hyperspectral images via convolutional neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. PP, no. 99, pp. 1-14, 2017.

[13] X. Zhang, G. Chen, W. Wang, Q. Wang and F. Dai, "Object-based land-cover supervised classification for very-high-resolution UAV images using stacked denoising autoencoders," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* vol. PP, no. 99, pp. 1-13, 2017.

[14] L. Mou, P. Ghamisi and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing,* vol. PP, no. 99, pp. 1-17, 2016.

[15] T. Moranduzzo, F. Melgani, M. L. Mekhalfi and Y. Bazi, "Multiclass coarse analysis for UAV imagery," *IEEE Transactions on Geoscience and Remote Sensing,* vol. 53, no. 12, pp. 6394-6406, 2015.

[16] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[17] A. Zeggada, F. Melgani and Y. Bazi, "A deep learning approach to UAV image multilabeling," *IEEE Geoscience and Remote Sensing Letters*, no. 99, pp. 1-5, 2017.

[18] Q. Tan, Y. Liu, X. Chen and G. Yu, "Multi-label classification based on low rank representation for image annotation," *Remote Sensing*, vol. 9, no. 2, 2017.

[19] Q. Li, B. Xie, J. You, W. Bian and D. Tao, "Correlated Logistic Model With Elastic Net Regularization for Multilabel Image Classification," in *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3801-3813, Aug. 2016.

[20] W. Shi, Y. Gong, X. Tao and N. Zheng, "Training DCNN by Combining Max-Margin, Max-Correlation Objectives, and Correntropy Loss for Multilabel Image Classification," *in IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, no. 99, pp. 1-13, 2017.

[21] W. Bi and J. T. Kwok, "Bayes-Optimal Hierarchical Multilabel Classification," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 11, pp. 2907-2918, Nov. 1 2015.

[22] K. Konstantinos, T. Grigorios, Z. Michalis and T. Panagiotis, "Deep learning for multi-label land cover classification," *in Proc. SPIE 9643, Image and Signal Processing for Remote Sensing* XXI, 2015.

[23] C. Corinna and V. Vladimir, "Support-vector networks," *Machine Learning,* vol. 20, no. 3, pp. 273-297, 1995.

[24] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *Journal of Machine Learning Research,* vol. 2, pp. 265-292, 2001.

[25] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Transactions on Geoscience and Remote Sensing,* vol. 42, no. 8, pp. 1778-1790, 2004.

[26] G. Mountrakis, J. Im and C. Ogole, "Support vector machines in remote sensing: A review," *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 66, no. 3, pp. 247-259, 2011.

[27] T. Ioannis, J. Thorsten, H. Thomas and A. Yasemin, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research,* vol. 6, pp. 1453-1484, 2005.

[28] T. Devis, M. M. jordi, K. Mikhail and V. C. Gustavo, "Structured output SVM for remote sensing image classification," *Journal of Signal Processing Systems,* vol. 65, no. 3, pp. 301-310, 2011.

[29] B. Alexander, M. Klaus-Robert and K. Motoaki, "On taxonomies for multi-class image categorization," *International Journal of Computer Vision,* vol. 99, no. 3, pp. 281-301, 2012.

[30] V. N. Navid, S. L. Roberto and W. Stefan, "Structured output prediction with hierarchical loss functions for seafloor imagery taxonomic categorization," in *Proc. 7th Iberian Conference on Pattern Recognition and Image Analysis*, 2015.

[31] M. Fauvel, J. Chanussot and A. Benediktsson, "A spatial-spectral kernel-based approach for the classification of remote-sensing images," *Pattern Recognition,* vol. 45, no. 1, pp. 381-392, 2012.

[32] M. Dundar, J. Theiler and S. Perkins, "Incorporating spatial contiguity into the design of a support vector machine classifier," in *IEEE International Symposium on Geoscience and Remote Sensing*, 2006.

[33] R. Flamary and A. Rakotomamonjy, "Support Vector Machine with spatial regularization for pixel classification," in *Proc. International Workshop on Advances in Regularization, Optimization, Kernel Methods and Support Vector Machones: Theory and Applications*, 2013.

[34] C. Szegedy, L. Wei, J. Yangqing, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, 2015.

[35] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25 (NIPS)*, Lake Tahoe, NV, 2012.

[36] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," preprint arXiv: 1409.1556, 2014.

[37] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016.

[38] J. Thorsten, F. Thomas and Y. J. Chun-Nam, "Cutting-plane training of structural SVMs," *Machine Learning,* vol. 77, no. 1, pp. 27-59, 2009.

[39] C. K. Chow and C. N. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE Transactions on Information Theory,* vol. 14, no. 3, pp. 462-467, 1968.

[40] K. Liu and G. Mattyus, "Fast multiclass vehicle detection on aerial images," *IEEE Geoscience and Remote Sensing Letters,* vol. 12, no. 9, pp. 1938-1942, 2015.

[41] A. C. Müller and S. Behnke, "pystruct - Learning Structured prediction in Python," *Journal of Machine Learning Research,* vol. 15, pp. 2055-2060, 2014.

[42] G. M. Foody, "Themantic map comparison: Evaluating the statistical significance of differences in classification accuracy," *Photogrammetric Engineering and Remote Sensing,* vol. 70, no. 5, pp. 627-633, 2004.

Chapter VI

Conclusions

In this thesis, different analysis and detection methods have been investigated for extremely/very high spatial resolution (HER/VHR) imagery acquired by means of unmanned aerial vehicle acquisition technologies. This dissertation covered several major aspects and challenges characterizing the requirements of the processing and analysis techniques that are meant to deal with the powerful potential displayed by such EHR images. Indeed, EHR imagery contains a big amount of information details to be processed, which represent a double-edged means. In other words, in spite of the fact that increasing the quantity of information that represent the characteristics of a single/ list of objects that might appear in an image enriches the amount of discriminative features promoting a better detection, still it is rather hard to design a suitable model that is capable to discriminate between different classes of objects when taking into consideration the resulting large intrinsic-class variability.

The scope of the thesis is mainly focused on scene description and analysis of extremely high-resolution images acquired over urban areas. Such complex scenarios are usually characterized by the simultaneous presence of several classes of objects in the scene. Consequently, we have addressed this issue by investigating several multilabel classification approaches. A key-contribution to be emphasized in this dissertation, is merging simultaneously both correlation between labels and spatial information (i.e., the likely high semantic content of the UAV-grabbed images and their spectral information) within i) a single decision process, ii) a conditional random fields framework. Furthermore, a UAV-imagery-based prototype for assisting avalanche search and rescue operations has been presented. In the following, we summarize the underlining highlights of the proposed strategies. For more details we forward the reader to the respective chapters.

Chapter 2, (A Deep Learning Approach for Assisting Avalanche Search and Rescue Operations with UAV Imagery), addresses another interesting application, namely, supporting avalanche search and rescue operations to track and detect victims by means of UAV technology. Equipped with a digital camera, UAVs represent an efficient real-time system that provides an interesting solution to integrate to the traditional work performed by the rescue teams. Such solution enhances the chances of victims' survival in emergency situations. In particular, UAVs are exploited to acquire EHR images of the avalanche debris fields. Afterwards, the acquired sequence of images are fed to a pre-trained CNN for feature extraction. The resulting features are classified using a linear SVM in order to detect objects of interest. Furthermore, we introduced (i) a pre-processing step to filter areas of interest from the video frames to decrease the processing time, and (ii) a post-processing step that exploits the correlation between the sequential frames output by means of hidden markov models to improve the classifier prediction performance.

In Chapter 3, (Multilabel Deep Learning Strategies for UAV Imagery Description), we investigated the problem of describing UAV images by exploiting two deep learning methods. Basically, our aim was to detect multiple object classes at once using a tile-based coarse description strategy. The two proposed strategies start with subdividing the query grabbed UAV image into a set of equal tiles. Afterwards, we infer the list of present objects within each tile. As a matter of fact, traditional pixel-based and object-based approaches would hardly provide a satisfactory scene classification, since neither pixel-based strategies can describe the class

behavior while taking into consideration each pixel separately, nor the object-based strategies that they ignore the global semantic context of the described scene, especially when dealing with a multilabeling task, where several classes might appear simultaneously. Departing from this, we resort to two deep learning representation techniques for the tile representation strategy. In particular, the first strategy exploits the features of a deep pre-trained convolution neural networks (i.e., GoogLeNet architecture) for the feature extraction. Afterwards, we substitute the softmax classifier on top of the GoogLeNet with a Radial Basis Fuction Neural Network to suit the multilabel classification requirements. The second strategy, takes advantage of three opportune handcrafted feature descriptors (i.e., HoG, wavelets transform, and RGB channels BoW). Subsequently, for the scope of providing a compact, yet affective representation of the handcrafted extracted features, a nonlinear feature reduction step take place using an autoencoder network. The resulting features are fed to an MLP classifier. Furthermore, for both proposed strategies, we apply a refinement step to further improve the yielded results by including on top of the whole architectures a multilabeling layer, which consists in a set of customized thresholding operations for each class output. The second strategy has demonstrated a better behavior than the first one in terms of processing time while maintaining a similar accuracy rate.

In Chapter 4, (Multilabel Conditional Random Field Classification for UAV Images), we reformulated the concept exposed in chapter 3, within a two stage tile-based structured prediction framework. In particular, we presented a novel multilabel conditional random field model that derives benefits from exploiting simultaneously label cross-correlation and spatial contextual information as a post-processing step to improve the initial classification predictions of the EHR UAV images. The proposed framework at the first stage generates initial predictions for each tile by means of an opportune classifier, after the subdivision of the query image into a set of equal tiles. At the second stage, the proposed multilabel CRF model is applied on the resulting map predictions in an iterative way to enhance the obtained outcomes.

Chapter 5, (Spatial and Structured SVM for Multilabel Image Classification), puts forth a novel multilabel SVM-based classification framework for extremely high resolution imagery. Its underlying idea relies on exploiting two types of information i) the label-label correlations captured by the conventional structural SVM model, and (ii) the spatial contextual information derived from the considered EHR scene. In particular, after the subdivision of the original image into small tiles, where each tile is subject to hold multiple class labels, we incorporate the output structure of the Structured SVM together with the likely expected spatial contiguity between the adjacent tiles into a single cost function. This basically aims to increase (i) the spatial smoothness and (ii) the semantic homogeneity of the resulting structured output of the proposed tile-based classification framework.

In overall terms, the proposed contributions provided in this thesis, have been concentrated on developing some novel detection and analysis methods to keep pace with the extremely high-resolution imagery acquired by the modern and very fast growing UAV technology. The obtained results are generally very promising and open the door for potential ameliorations and future developments. In this regard, we put forward some aspects and open issues of the research that would be worth investigating further.

One of the main concerns in the proposed tile-based multilabel classification frameworks is the heterogeneous spatial size and spectral characteristics of the list of object classes that one aims at recognizing. Indeed, adopting the same paradigm to proceed with the recognition of different classes of objects that are alike in terms of their respectful feature characteristics and homogeneity increases the complexity of getting a satisfactory detection rate for all of them. In other words, the multilabel nature of the problem entails handling a multitude of objects that manifest various colors, shapes, illumination and scale changes. Indeed, despite the good overall results, the proposed methods have the drawback of poor detection rate for some classes. In particular, while they enhance the accuracies of some classes, they degrade others. This usually happens due to some main factors: i) dominant classes get a better accuracy at the expense of minor classes (i.e., the unbalanced numbers of present objects classes in the images which is the case of most datasets); ii) accuracy metrics exploited to measure the performance of the proposed methods (we focused on the average of sensitivity and specificity in our case) which is relative to the semantic importance of the targeted classes varying from an application to another iii) low cardinality of present labels in each image tile. Therefore, these raised points are very interesting issues to be further investigated in future works.

As foresaid, it is a matter of fact that the detection of some classes of objects is harder than others, especially when using the same set of parameters, which are fixed in a way to compromise between all these diverse objects properties within the adopted classification paradigm. On this point, some potential solutions could be investigated. For instance, in all the presented classification frameworks we have limited the recognition process to RGB aerial imagery acquired by means of a digital camera (i.e., red, green, and blue bands). However, the usage of different multispectral sensors would enhance the capability to capture in a better way the characteristics of the different objet classes and thus to extract better discriminative features. For instance, the presence of shadows represents an obstacle when using only color information. To overcome this problem, the integration of light detection and ranging (LiDAR) data would significantly lessen the impact of shadows on the interpretability of the acquired images. Moreover, RGB data alone fails to distinguish between object classes that share similar texture and colors, such as Asphalted Roofs and Asphalted Roads or different types of vegetation i.e., Grass, Low-Vegetation and Trees. However, thanks to the capability of LiDAR and multispectral data to estimate the elevation of the targeted objects, the distinction between such classes becomes easier. The same way goes for thermal sensors, which have proven to be very efficient in persons' detection in both urban scenarios and avalanche search and rescue operations.

Furthermore, on what concerns the propsed multilabel tile-based methods, instead of using fixed tile sizes, one could consider using several tile sizes adapted to the shape of each class of the predefined list of objects. For instance, one could exploit different sizes each time in the subdivision of the image into a set of equal tiles. Afterwards, an ensemble of the multilabel classification methods are trained on each set of tile size. The final step to take is to exploit an appropriate method to fuse the resulting posterior classification probabilities by penalizing the classification errors, for instance, the Ordered Weighted Averaging (IOWA) method [1]. The same concept could be used to couple relevant state-of-the-art methods i.e., tile representation, spatial correlation, and multilabel classifiers in order to consolidate the framework capability of handling

different aspects of the objects properties. We believe that such improvements are reasonably expected to further enhance the classification performance.

Another subject of future developments, yet very pivotal concerning the deep learning techniques (i.e., CNNs) in multilabeling classification. As a matter of fact, the existing CNN architectures are not intrinsically developed to handle either multilabeling tasks nor to perform contextual decisions. However, they need to be adapted for instance by changing the structure of the top layer [2] [3] [4]. Therefore, the most adequate way to make a full advantage of them would be to intervene on their cost functions to satisfy the aforesaid concern.

Ultimately, in remote sensing applications, the acquired data are subject to data shift problems due to the various changes of the surveyed geographical areas, atmospheric conditions, and the quality of the exploited sensors. Therefore, investigating some domain adaptation techniques such as transfer learning is worth addressing in order to adapt the proposed image description and analysis methods to operate on different data topologies. In particular, a very promising direction to adopt is to make use of the generative adversarial networks [5], which are an instance of generative probabilistic models. They are exploited to generate samples from a source data (i.e., training) without explicitly defining its density distribution. Generative adversarial networks have been proven effective in diverse unsupervised learning tasks such as image inpainting [6], text /image-to-image translation [7] [8], image super resolution [9], clustering [10], and domain adaptation [11][12][13]. The underlying idea of GANs is to set up an adversarial game between two models (i.e., generative and discriminative) iteratively. In particular, at one hand, the generative model is trained to fool the discriminator model by generating new samples that are intended to be derived from the distribution of the training data. At the other hand, the discriminative model tries to divide these generated samples into two classes (i.e., fake and real) in order to authenticate weather they belong to the same distribution of the training data or not. The final goal of the generative model is to be able to create samples that are indistinguishable from the real ones. Moreover, GANs can also be exploited for classification tasks [14][15], yet such technique favors the scenarios in which the target labels are binary. Indeed, the basic GANs has a single discriminator and generator models, however, we believe that adapting GANs framework to mutli-label/class tasks is very promising. For instance, instead of restricting the GANs framework to a single discriminator with two classes output (i.e., real and fake) and a single generator, one could think of adding for each object class a discriminator with a real and fake version together with a generator to distinguish between each class distribution.

6.1. REFERENCES

[1] R. R. Yager, and D. P. Filev. "Induced ordered weighted averaging operators," in *IEEE Transactions on Systems, Man, and Cybernetics*, Part B, 29(2):141-150, 1999.

[2] W. Shi, Y. Gong, X. Tao and N. Zheng, "Training DCNN by Combining Max-Margin, Max-Correlation Objectives, and Correntropy Loss for Multilabel Image Classification," in IEEE Transactions on Neural Networks and Learning Systems, vol. PP, no. 99, pp. 1-13.

[3] J. Zhu, S. Liao, D. Yi, Z. Lei and S. Z. Li, "Multi-label CNN based pedestrian attribute learning for soft biometrics," *2015 International Conference on Biometrics (ICB)*, Phuket, 2015, pp. 535-540.

[4] G. Chen, D. Ye, Z. Xing, J. Chen and E. Cambria, "Ensemble application of convolutional and recurrent neural networks for multi-label text categorization," *2017 International Joint Conference on Neural Networks (IJCNN)*, Anchorage, AK, 2017, pp. 2377-2383.

[5] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. 2014. Generative Adversarial Nets. *Advances in Neural Information Processing Systems 27*. Curran Associates, Inc. 2672–2680.

[6] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell and A. A. Efros, "Context Encoders: Feature Learning by Inpainting," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 2536-2544.

[7] M. Cha, Y. Gwon and H. T. Kung, "Adversarial nets with perceptual losses for text-to-image synthesis," *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, Tokyo, Japan, 2017, pp. 1-6.

[8] Z. Yi, H. Zhang, P. Tan and M. Gong, "DualGAN: Unsupervised Dual Learning for Image-to-Image Translation," *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017, pp. 2868-2876.

[9] C. Ledig *et al*., "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 105-114.

[10] J. T. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. arXiv preprint arXiv:1511.06390, 2015.

[11] E. Tzeng, J. Hoffman, K. Saenko and T. Darrell, "Adversarial Discriminative Domain Adaptation," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 2962-2971.

[12] M. Wulfmeier, A. Bewley and I. Posner, "Addressing appearance change in outdoor robotics with adversarial domain adaptation," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada, 2017, pp. 1551-1558.

[13] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan and D. Krishnan, "Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 95-104.

[14] Y. Zhan, D. Hu, Y. Wang and X. Yu, "Semisupervised Hyperspectral Image Classification Based on Generative Adversarial Networks," in *IEEE Geoscience and Remote Sensing Letters*, vol. PP, no. 99, pp.1-5. doi: 10.1109/LGRS.2017.2780890.

[15] R. Tachibana, T. Matsubara and K. Uehara, "Semi-Supervised learning using adversarial networks," *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, Okayama, 2016, pp. 1-6.

List of Related Publications

Journal Papers:

[J1] A. Zeggada, F. Melgani and Y. Bazi, "A deep learning approach to UAV image multilabeling," *IEEE Geoscience and Remote Sensing Letters*, no. 99, pp. 1-5, 2017.

[J2] M. B. Bejiga, A. Zeggada, A. Nouffidj, and F. Melgani, "A convolutional neural network approach for assisting avalanche search and rescue operations with UAV imagery," *Remote Sensing*, vol. 9, no. 2, 2017.

[J3] A. Zeggada, S. Benbraika, F. Melgani and Z. Mokhtari, "Multilabel Conditional Random Field Classification for UAV Images," *IEEE Geoscience and Remote Sensing Letters*, vol. PP, no. 99, pp. 1-5, 2018.

[J4] S. Koda, A. Zeggada, F. Melgani, R. Nishii, "Spatial and Structured SVM for Multilabel Image Classification" *IEEE Transactions on Geoscience and Remote Sensing* (accepted with minor revisions).

Conference Proceedings:

[C1] A. Zeggada and F. Melgani, "Multilabel classification of UAV images with Convolutional Neural Networks," 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, 2016, pp. 5083-5086.

[C2] M. B. Bejiga, A. Zeggada and F. Melgani, "Convolutional neural networks for near real-time object detection from UAV imagery in avalanche search and rescue operations," *2016 IEEE International Geoscience and Remote Sensing Symposium* (IGARSS), Beijing, 2016, pp. 693-696.

[C3] A. Zeggada and F. Melgani, "Multilabeling UAV Images with Autoencoder Networks ", *IEEE Urban Remote Sensing Event (JURSE)* in Dubai, 2017.