SEGMENTAÇÃO SEMÂNTICA DE IMAGENS DE SENSORIAMENTO REMOTO EM CENÁRIO DE CONJUNTO ABERTO

CAIO CESAR VIANA DA SILVA

SEGMENTAÇÃO SEMÂNTICA DE IMAGENS DE SENSORIAMENTO REMOTO EM CENÁRIO DE CONJUNTO ABERTO

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

Orientador: Jefersson Alex dos Santos

Belo Horizonte
Novembro de 2019

CAIO CESAR VIANA DA SILVA

OPEN SET SEMANTIC SEGMENTATION OF REMOTE SENSING IMAGES

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

Advisor: Jefersson Alex dos Santos

Belo Horizonte
November 2019

Ficha catalográfica elaborada pela bibliotecária Irénquer Vismeg Lucas Cruz- CRB 6ª Região nº 819

Silva, Caio Cesar Viana da.

S586o Open set semantic segmentation of remote
sensing images / Caio Cesar Viana da Silva— Belo
Horizonte,
2019.
xxvii, 70 f. : il.; 29 cm.

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

Orientador: Jefersson Alex dos Santos

1. Computação – Teses. 2. Visão por computador
– Teses. 3. Aprendizado do computador – Teses.
4. Sensoriamento remoto – Teses. 5. Linguagem de
programação (Computadores) - Semantica. – Teses.
I. Orientador. II. Título.

CDU 519.6*82(043)

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

FOLHA DE APROVAÇÃO

Open Set Semantic Segmentation of Remote Sensing Images

CAIO CESAR VIANA DA SILVA

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

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

PROF. FABRÍCIO MURAI FERREIRA
Departamento de Ciência da Computação - UFMG

Rodrigo Minetto
PROF. RODRIGO MINETTO
Departamento Acadêmico de Informática - UTFPR

PROF. HEITOR SOARES RAMOS FILHO
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 1 de Novembro de 2019.

I dedicate this work to my family, teachers and especially friends who have supported me since the beginning of this journey until this moment of realization.

Acknowledgments

My thanks go to

This institution for the excellent environment offered to its students and the qualified professionals it provides to teach us.

To Professor Jefersson Alex dos Santos for all his attention, dedication and effort so that I could have confidence and security in the completion of this dissertation.

To Ana Luiza da Silva, my wife, for being by my side through all this journey, helping me trace goals and achieving them, for also listening to my ideas and helping find solutions to the problems that surfaced.

To my family, Simone Beatriz Pedrozo Viana, José Everton da Silva and Marcos Vinícius Viana da Silva, for their love, affection, patience, wisdom and complete support in all stages of my life.

To Keiller Nogueira for helping me understand the base concepts of open set techniques and for helping me transform theory in practice.

To my friends and colleagues for the trust they have placed in me, for all the moments they have provided, for the knowledge they have exchanged and for the help received.

This project was partially financed by CAPES, CNPq and FAPEMIG.

“It’s not the answers that move the world, it’s the questions.”

(Albert Einstein)

Resumo

As principais abordagens desenvolvidas em visão computacional e processamento de imagem digital são voltadas para dados obtidos por meio de smartphones e câmeras compactas. Essas câmeras normalmente são usadas para capturar cenas nos canais RGB, ou seja, apenas no espectro visível. Outra fonte de imagens que são exploradas pela visão computacional, são as imagens de satélite ou imagens aéreas. Entretanto, o desenvolvimento de abordagens de visão computacional que exploram as imagens de satélite é relativamente recente devido principalmente à pouca disponibilidade a esse tipo de imagem. Até pouco tempo atrás elas eram de exclusivo uso militar. O acesso a imagens aéreas, inclusive com informação espectral, vem aumentando principalmente devido ao baixo custo de drones, novos satélites de uso civil, e conjuntos de dados em diversas plataformas públicas. Na área de sensoriamento remoto, as aplicações que empregam técnicas de visão computacional são modeladas para classificação em cenários fechados (*closed set*). No entanto, o mundo não é puramente *closed set*, muitos cenários apresentam classes que não são previamente conhecidas pelo algoritmo, um cenário de conjunto aberto (*open set*). Desse modo, o objetivo principal desta dissertação é o estudo e desenvolvimento de técnicas de segmentação semântica considerando o cenário *open set* aplicado a imagens de sensoriamento remoto. As principais contribuições dessa dissertação são: (1) uma discussão dos trabalhos relacionados, mostrando evidências de que técnicas de segmentação semântica podem ser adaptadas para cenários *open set*; e (2) o desenvolvimento de dois métodos para segmentação semântica *open set*. Os métodos OpenPixel e OpenFCN apresentaram resultados competitivos quando comparados aos métodos *closed set* no mesmo conjunto de dados. Em média, o método OpenPixel apresentou uma acurácia geral de 57,51%, uma acurácia normalizada de 54,23% e um Índice Kappa de 0,5602. Para o OpenFCN, o método resultou em uma acurácia geral de 82,27%, uma acurácia normalizada de 64,39% e um Índice Kappa de 0,7630. É possível concluir que os métodos propostos podem segmentar classes desconhecidas enquanto ainda classificam de forma correta a maioria das classes conhecidas, realizando uma segmentação semântica *open set* em imagens de sensoriamento remoto.

Palavras-chave: Conjunto Aberto, Aprendizado Profundo, Segmentação Semântica, Sensoriamento Remoto.

Abstract

The main approaches developed in computer vision and digital image processing are focused on data obtained through smartphones and compact cameras. These cameras are typically used to capture scenes on RGB channels, only in the visible spectrum. Another source of images that are exploited by computer vision is satellite images or aerial images. However, the development of computational vision approaches that exploit satellite imagery is relatively recent, mainly due to the limited availability of this type of image. Until recently they were exclusively for military use. Access to aerial imagery, including spectral information, has been increasing mainly due to the low cost of drones, new civilian satellites, and data sets on various public platforms. In the area of remote sensing, applications that employ computational vision techniques are modeled for classification in closed set scenarios. However, the world is not purely closed set, many scenarios present classes that are not previously known by the algorithm, an open set scenario. Thus, the main objective of this dissertation is the study and development of semantic segmentation techniques considering the open set scenario applied to remote sensing images. The main contributions of this dissertation are: (1) a discussion of related works, showing evidence that semantic segmentation techniques can be adapted for open set scenarios; (2) the development of two methods for open set semantic segmentation. The OpenPixel and OpenFCN methods presented competitive results when compared to the closed set methods in the same data set. On average, the OpenPixel method had an overall accuracy of 57.51%, a normalized accuracy of 54.23% and a Kappa Index of 0.5602. For OpenFCN, the method resulted in an overall accuracy of 82.27%, a standard accuracy of 64.39% and a Kappa Index of 0.7630. It is possible to conclude that the proposed methods can segment unknown classes while still correctly classifying most of the known classes, performing open set semantic segmentation on remote sensing images.

Palavras-chave: Open Set, Deep Learning, Semantic Segmentation, Remote Sensing.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Motivation

Since the first experiments with cameras and photography in the 18^{th} century, the ability to capture a moment have brought a lot of possibilities of study and applications. In recent years, we can observe an increase in the number of studies and techniques of computer vision caused by the improvement of the quality and ease of acquisition of photographic images Sumbul et al. [2017].

The main approaches developed in computer vision and digital image processing are focused on data obtained through smartphones, compact cameras, smartwatches, glasses, other wearables, and IoT devices. Those cameras are normally used to capture scenes, people and objects, and in general only capture images in the RGB channels, the visible spectrum.

Another source of images that are exploited by computer vision is satellite images or aerial images. However, satellite imagery has not been widely used mainly due to the limited availability of this type of image. Until recently they were of exclusive military use. Access to aerial images, including spectral information, has been increasing mainly due to the low cost of drones, new civilian satellites, and data sets on various public platforms.

With the advances in studies on computer vision and image processing, a lot of remote sensing applications are being developed, for example, we have softwares that help to monitor deforestation or forest burns, as well as change detection in waste dams and erosion segmentation in railway constructions. Those applications could only be developed with the advance in remote sensing research, including new technologies as networks based on deep learning.

These aerial images can bring extra information, such as infrared, ultra-violet and other bands, which increases the amount of data about a given scene, and that can be used in several remote sensing applications Lillesand et al. [2014].

According to Andrade [2012], there are also several research challenges from the computational perspective when working with remote sensing image classification such as:

- remote sensing data is inherently big, even at 250 meters coarse spatial resolution, a low resolution for remote sensing application, some aerial images contains more than 20 million pixels, jointly with a time series and a high number of bands. Most machine learning models described as state-of-art (e.g., Deep Neural Networks, nonlinear Support Vector Machines), cannot handle with the magnitude of this data easily, making it obligatory the use of preprocessing techniques;
- segmentation scale, accompanied by a large amount of information at the level of the object in very high spatial resolution images, the segmentation algorithms have difficulty in defining the optimum scale to be used;
- pixel mixture and dimensionality reduction, images with high spectral resolution must be preprocessed due to problems such as high dimensionality, treatment of noise and corrupted bands, a mixture of pixels due to the low spatial resolution;
- even collecting information from various sensors, efficiency and capability to process that amount of data is desired or even crucial depending on the application. In some applications, the data must be analyzed in near real-time, which can be translated to a very computationaly consuming task.

Within the area of remote sensing, most applications work with closed set scenario-based computer vision techniques. However, the world is not purely closed set, many scenarios present objects that are not previously known by the algorithm. These other scenarios would benefit from using open set algorithms Scheirer et al. [2013], due to the nature of the images. Many of the plantations, cities or objects that appear in the images are restricted to the place where the images were registered. It can increase the difficulty of training all the possible classes so that the algorithms can be utilized in more than one place. Their scenario cannot be as well controlled as others in traditional computer vision applications, there is an undefined amount of objects not known by the machine that can be registered on the images.

The open set classification can be described as a classification in which an image can be labeled as belonging to one of the classes learned by the algorithm or as an unknown class if it belongs to any class not learned. The biggest challenges of open set classification are: (1) the diversity of features of the unknown class, since the unknown class can aggregate multiple classes that were not present during training; (2) the similarity between features of known and unknown classes, in example, an algorithm that have seen examples of aerial images containing trees, may have difficulty in labeling as unknown an aerial image of grass.

Even with a variety of possible uses for open set scenario algorithms, this area is still unexplored, especially when compared to the infinite number of closed set methods. Analyzing the small world of open set methods, one can observe that most of them perform scene classification, lacking the open set concept on semantic segmentation methods.

Semantic Segmentation is a task, in computer vision, that aims to classify not an image as a whole but every pixel in an image accordingly to the classes learned by the algorithm. This task can be classified as a difficult task since it needs: (1) complex datasets, in which each image needs to have all the pixels annotated, consuming more time to be developed than a dataset for image classification; (2) complex algorithms, that needs to take in account the classification of every single pixel in its decision-making process to achieve a better accuracy.

In this way, an open set semantic segmentation would be a technique that receives an image as input and outputs a prediction in which all the pixels are labeled with a known class, seen during training or as belonging to an unknown class.

Observing these motivations, this dissertation is innovative, since it is the first work to introduce the concept of open set semantic segmentation for images, to propose two methods based on this new concept and to apply them to the field of remote sensing that usually happens in an open set scenario context.

1.2 Objectives

The main objective of this master's dissertation is to study and to develop techniques for semantic segmentation for open set scenarios, applied to the remote sensing field. Specific objectives include:

- Literature review of existing open set techniques
- Adaptation and development of techniques for semantic segmenting satellite images containing known and unknown classes during the test
- Comparison of the results obtained with the methods developed in relation to closed set techniques on an open set scenario.

1.3 Contributions

In practice, the main contributions of this work are:

- evidence that semantic segmentation techniques can be applied to open set scenarios,
- a new method for open set semantic segmentation based on the Pixelwise network Nogueira et al. [2016], and
- a new method for open set semantic segmentation based on the Openmax method Bendale and Boult [2016].

1.4 Outline

The rest of the document is structured in seven chapters. The first chapter (2) consists of the main concepts needed to understand the work of this dissertation. Chapter 3 presents the related works, explaining some others techniques existent on the open set scenario and what are their differences in relation to the methods proposed in this dissertation. Chapter 4 brings the methodology adopted in this dissertation for achieving the objectives defined previously. All the configuration and some assumptions needed to reproduce this work are described in Chapter 5. Chapter 6 brings the results found using the proposed methods and their analysis. Finally, Chapter 7 brings a conclusion on the work.

Chapter 2

Key Concepts

Before explaining the related works, methodology and experiments done in this dissertation, it is important to understand some key concepts behind this work.

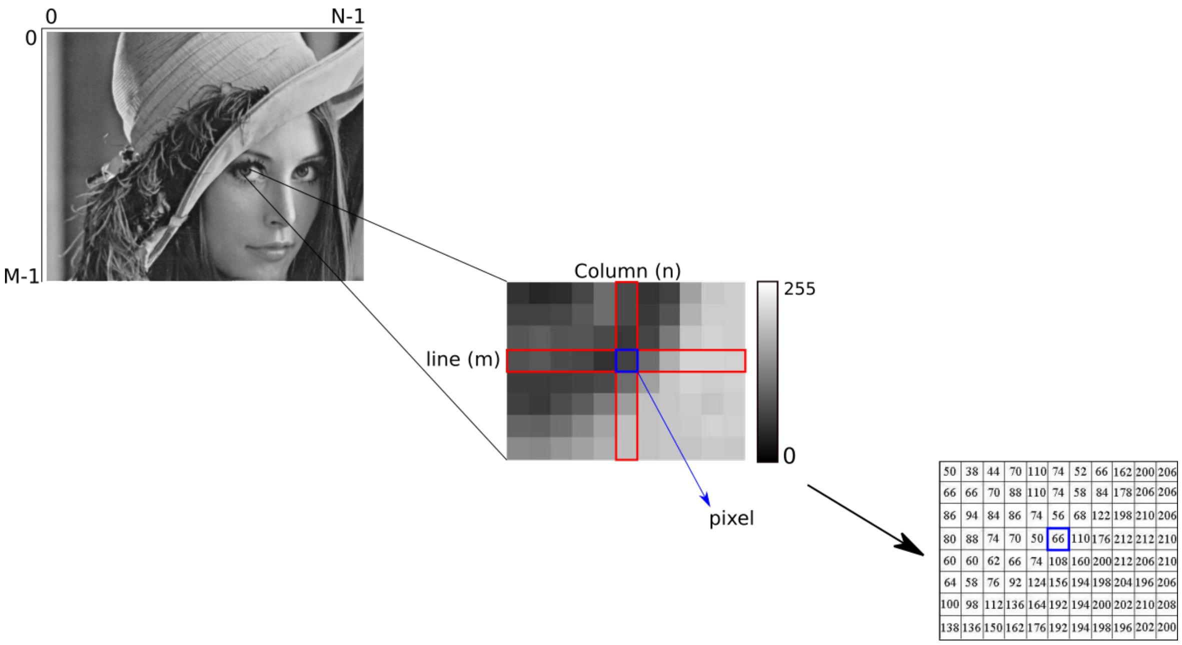
2.1 Digital Images

A monochromatic image is a continuous two-dimensional function $f(x,y)$, in which x and y are spatial coordinates and the value of f at any point (x,y) is proportional to the intensity (brightness or luminance level) at the point considered. As computers are not able to process continuous images, but only arrays of digital numbers, it is necessary to represent images as two-dimensional point arrangements.

Each point in the two-dimensional grid that represents the digital image is called image element or pixel. Figure 2.1 shows the usual matrix notation for the location of a pixel in the pixel arrangement of a two-dimensional image. The first index denotes the position of the line, m, at which the pixel is located, while the second, n, denotes the position of the column. If the digital image contains M lines and N columns, the index m will vary from 0 to $M-1$, while n will vary from 0 to $N-1$. It is valid to observe that the reading direction and the convention usually adopted in the spatial representation of a digital image differs from the traditional Cartesian Plane, in which the x axis is read from the left to right and the y varies from bottom to the top.

The luminous intensity at the point (x,y) can be decomposed into: (i) lighting component, i(x,y), associated with the amount of light incident on the point (x,y); and the reflectance component, r(x,y), associated with the amount of light reflected by the point (x,y). The product of i(x,y) and r(x,y) results in:

$$f(x,y) = i(x,y) \cdot r(x,y),$$



50	38	44	70	110	74	52	66	162	200	206
66	66	70	88	110	74	58	84	178	206	206
86	94	84	86	74	56	68	122	198	210	206
80	88	74	70	50	66	110	176	212	212	210
60	60	62	66	74	108	160	200	212	206	210
64	58	76	92	124	156	194	198	204	196	206
100	98	112	136	164	192	194	200	202	210	208
138	136	150	162	176	192	194	198	196	202	200

Figure 2.1: Digital representation of a gray scale image.

in which $0 < i(x, y) < \infty$ and $0 < r(x, y) < 1$, where i(x,y) depends on the characteristics of the light source, while r(x,y) depends on the characteristics of the object surfaces.

In a digital color image in the RGB system, a pixel can be seen as a vector whose components represent the red, green and blue intensities of its color. The color image can be seen as the composition of three monochromatic images, a vector whose components represent the red, green and blue intensities of its color:

$$f(x, y) = f_R(x, y) + f_G(x, y) + f_B(x, y)$$

in which $f_R(x, y)$, $f_G(x, y)$, $f_B(x, y)$ represent, respectively, the luminous intensities of the red, green and blue components of the image at point (x, y).

Figure 2.2 shows the monochromatic planes of an image and the result of the composition of the three planes. The same concepts formulated for a monochrome digital image apply to each plane of a color image.

2.2 Remote Sensing

Remote sensing is defined as the measurement of object properties on the Earth's surface using data acquired from aircraft and satellites. It is, therefore, an attempt to measure something at a distance, rather than *in situ*. Schowengerdt [2006]. Or as

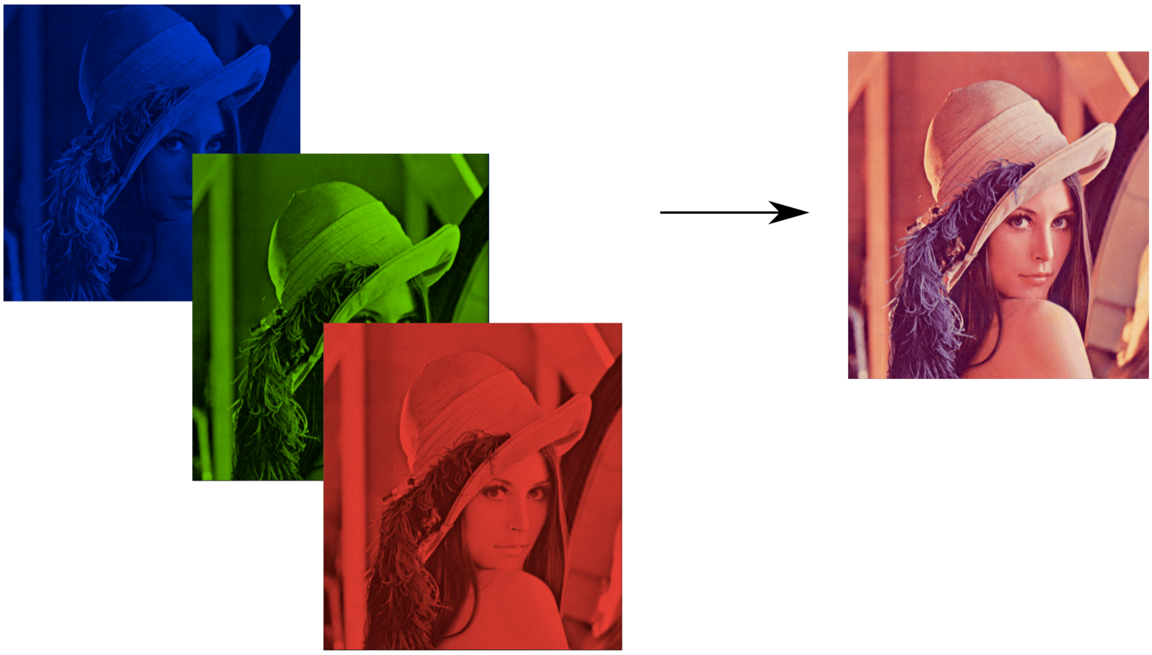


Figure 2.2: Digital representation of an image in the RGB space.

Câmara et al. [1996] defines, the assembly of processes and techniques used to measure and record the electromagnetic properties of the terrestrial surface through detection of the radiant energy flow reflected or emitted by natural targets, objects, through the utilization of sensors with no direct contact between them.

Understanding the spectral characteristics of a body, forest, a field, pasture or an agricultural crop, is essential to comprehend the process of its detection by sensors in satellite cameras Moreira [2005]. After the process of detection, the data acquired by the sensors are converted into digital format. The scanned data is a set of pixels, which, distributed in rows and columns, form an image.

A formal definition of multi-band images used in remote sensing, according to da Silva Torres and Falcao [2006], is denoted by $\hat{I} = \left(D_I, \vec{I}\right)$, where:

- D_I is a finite set of points in $\mathbb{Z}^n$ (image domain) and n refers to the size of the image, and

- $\vec{I} : D_I \to D'$ is a function that assigns each p-pixel in D_I a set of scalar values $\{I_1(p), I_2(p), \ldots, I_k(p)\}$ associated with some physical property. The value of k refers to the number of bands. For example, $D' = \mathbb{R}^3$ when a color, in the RGB system, is assigned to a pixel.

An example of remote sensing image in the RGB system is presented in Dos Santos [2013], and can be seen in Figure 2.3, in which a flow of radiant energy was detected

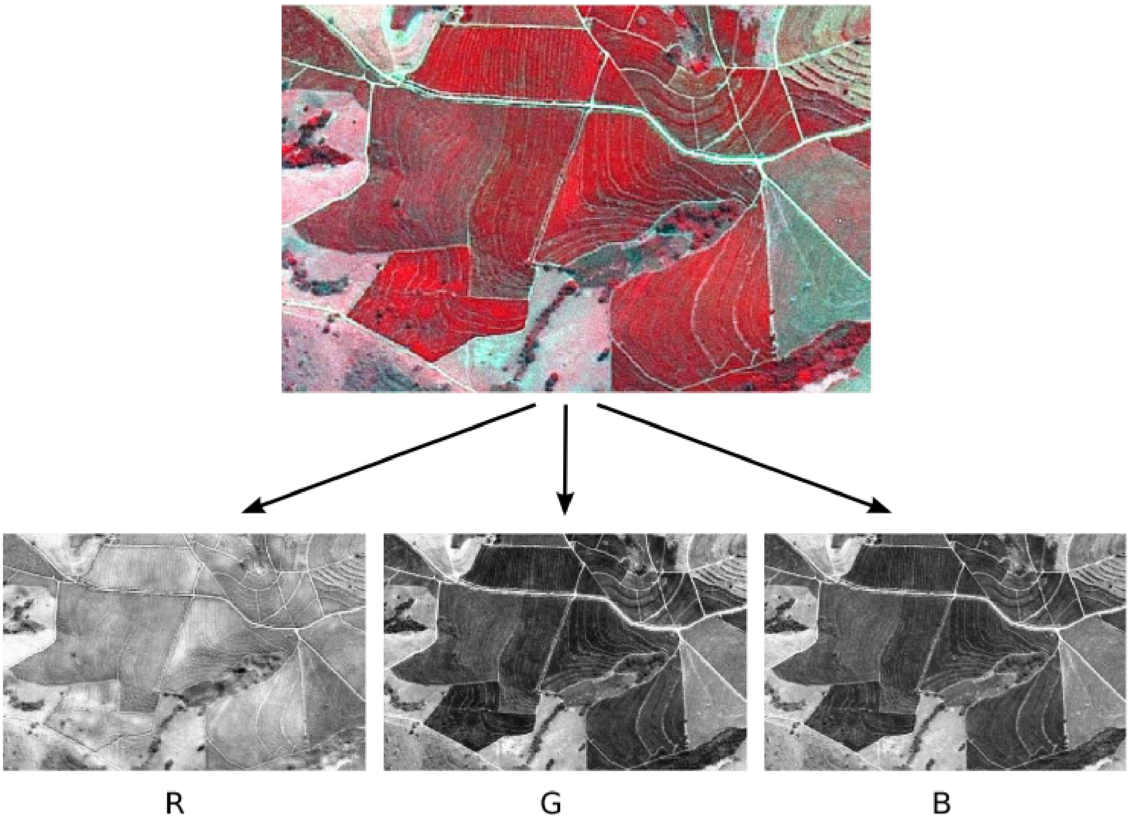


Figure 2.3: An excerpt of a remote sensing image extracted from a coffee region in the municipality of Monte Santo (MG) and its decomposition in RGB color channels Dos Santos [2013].

in a region of the city of Monte Santo (southern Minas Gerais State, Brazil), a traditional region in coffee cultivation. In Figure 2.3, the red color channel (R) corresponds to values captured in the near-infrared spectral range, the green channel (G) to the medium infrared and the blue channel (B) to the visible green spectrum.

Many of the studies in remote sense were focused on the adaptation of techniques used in traditional computer vision scenarios for scene classification, segmentation and object recognition.

2.3 Artificial Intelligence

Artificial intelligence is a term broadly used by technology companies, marketing, and even academic projects. But the definition of artificial intelligence is not as precise as one could expect, and normally differs from the technology used by nowadays applications.

The term artificial intelligence first appeared in McCarthy et al. [2006] as an invitation to a group of researchers from a variety of disciplines, including language simulation, neuron nets and complexity theory, for a summer workshop called the Dartmouth Summer Research Project on Artificial Intelligence with the aim to discuss the, still to be born, field.

The main goal of the researchers was to develop the concepts of thinking machines, a concept with divergent explanations by that time. The choice of the term artificial intelligence was made based on its neutrality. Since the field was and still is, very vast, the use of a neutral term would not give more importance to one or other applications.

The discussed subject was based on the concept that the study is to proceed based on the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it McCarthy et al. [2006].

In the computational area, artificial intelligence is usually used as a term to designate an algorithm or process that may allow the computer to perform tasks that need a level of intelligence for decision-making. This concept is usually blended with the idea of machine learning or deep learning. Figure 2.4 better represents the relation between those concepts, which will be used in this dissertation.

2.3.1 Machine learning

Machine learning can be treated as a subsection inside artificial intelligence. Although the name machine learning was coined in Samuel [1959], it was in Mitchell [1997] that the term gained a more formal and widely used definition. The concept was that a computer program is said to learn from experience E concerning some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.

This concept means that the term machine learning covers the idea of teaching a computer to take actions in the same way as a human would be given a specific task and data and that those actions taken could be compared to actions done by humans and this could help improve the quality of the decisions taken.

The concept is always dynamic, evolving and changing alongside the evolution of technology, and nowadays the concept is more based on the idea that the algorithm should learn how to get the same results as a human. It does not necessarily need to perform the same process of decision-making, but it is expected to take the same action a human would. This change in the concept was introduced more hardly with the development of deep learning algorithms.

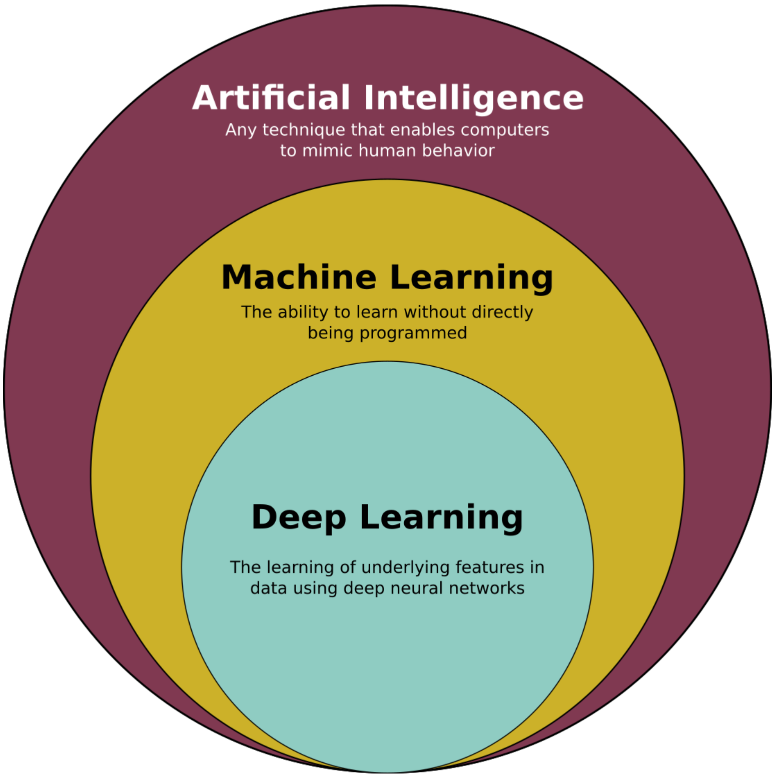

Artificial Intelligence
Any technique that enables computers to mimic human behavior
Machine Learning
The ability to learn without directly being programmed
Deep Learning
The learning of underlying features in data using deep neural networks

Figure 2.4: Representation of the hierarchy of Artificial Intelligence, Machine Learning and Deep Learning.

2.3.1.1 Deep Learning

One of the big problems conventional machine learning techniques had before the use of deep learning was their limited ability to process natural data in their raw form. According to LeCun et al. [2015], to construct a method that performed machine learning would require engineering and domain expertise to design a feature extractor that transformed the raw data (such as the pixel values of an image) into a suitable internal representation or feature vector from which the learning subsystem, often a classifier, could detect or classify patterns in the input.

The biggest shift with the introduction of a deep learning system was not a change in the process, but an integration of the feature extraction step, normally done by humans, in the algorithm. Deep learning methods are learning methods for representation, which means that they can be fed with raw data and to automatically discover the representations needed for detection or classification. These representations are done in multiple levels, obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level.

In LeCun et al. [2015] it is also explained that this process of applying multi-

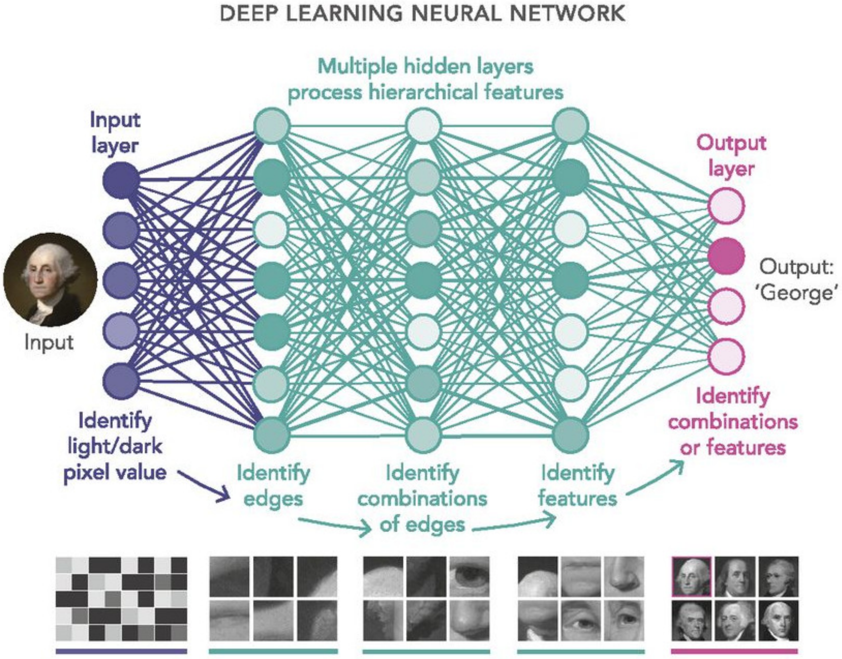

DEEP LEARNING NEURAL NETWORK
Multiple hidden layers process hierarchical features
Input layer
Output layer
Output: 'George'
Input
Identify light/dark pixel value
Identify edges
Identify combinations of edges
Identify features
Identify combinations or features

Figure 2.5: Simple representation of a deep learning networking and the features related to multiple levels Waldrop [2019] .

ple level transformations allowed the methods to learn very complex functions. For example, in classification tasks, higher layers of representation can be interpreted as amplifiers of important aspects of the input for the classification.

When applied to use on images, as in this dissertation, the first layers of the method typically represent the presence or absence of edges at particular orientations and locations in the image. The next layers typically detect corners by spotting particular arrangements of edges, regardless of small variations in the edge positions. The higher layers normally assemble those corners into larger combinations that correspond to parts of familiar objects, and subsequent layers would detect objects as combinations of these parts and so on. Figure 2.5 shows an example of the network and the types of features being used on each level.

The main aspect of deep learning, when compared to other machine learning techniques is that these layers of features are not designed by human experts, they are learned from the input data. This introduces the idea of layers that correlate data in a way humans may not be able to understand. Those layers may increase the accuracy of methods or even find results and information not predicted by humans. At the same time, this freedom is given to the algorithms to find the best combination and the use of the raw data also made it harder to interpret the decision process. It is possible to analyze the result of the decision, but it became harder to understand why and how that decision was made.

2.4 Closed and Open Set Scenario

All the techniques of machine learning are based on the concept of one of the two possible scenarios, closed and open set. Even though those scenarios are opposites, they still share some similarities, methods, and networks.

2.4.1 Closed Set Scenario

The closed set scenario is the most used in all types of networks and methods in machine learning. This scenario is applied in methods for scene classification, recognition, and semantic segmentation.

The closed set can be described as all the techniques in which all testing classes are known at training time Scheirer et al. [2013]. This way, when a new image is tested, it has to belong to one of the classes learned in the training phase. In this scenario, researchers have assumed one has examples from all classes, and have subsequently labeled the entire space.

To better understand this scenario, a good example would be a method that can classify an image in cats or dogs. During the training, the method would have multiple examples of images belonging to those two classes, cats and dogs. This way the model generated by the technique can differentiate both classes. During the test phase, the algorithm would be fed with input images different from the ones seen during the training. Although the example images are different, the class they belong to tends to be the same. This is the ideal scenario for closed set programs.

In this ideal scenario, the test image would be a cat or a dog and the system would need to analyze the image, and predict which of both classes better represents that image. But sometimes the context is not so ideal, like in the real world. In this case, the image evaluated during the test may not belong to the classes learned, for example, an image of a horse. The closed set algorithm when used on an image with a class different from the one learned, has only one option, to try to predict which trained class better describes the test image, labeling the horse picture as a dog or a cat. Figure 2.6 represents both cases, when the image being evaluated belongs to one of the learned classes and when it does not.

It is obvious that when a closed set method is tested against a context that allows images from unknown classes to be evaluated, its accuracy and correctness will not be as good as planned. This is solved using the idea of concept restriction: the closed set

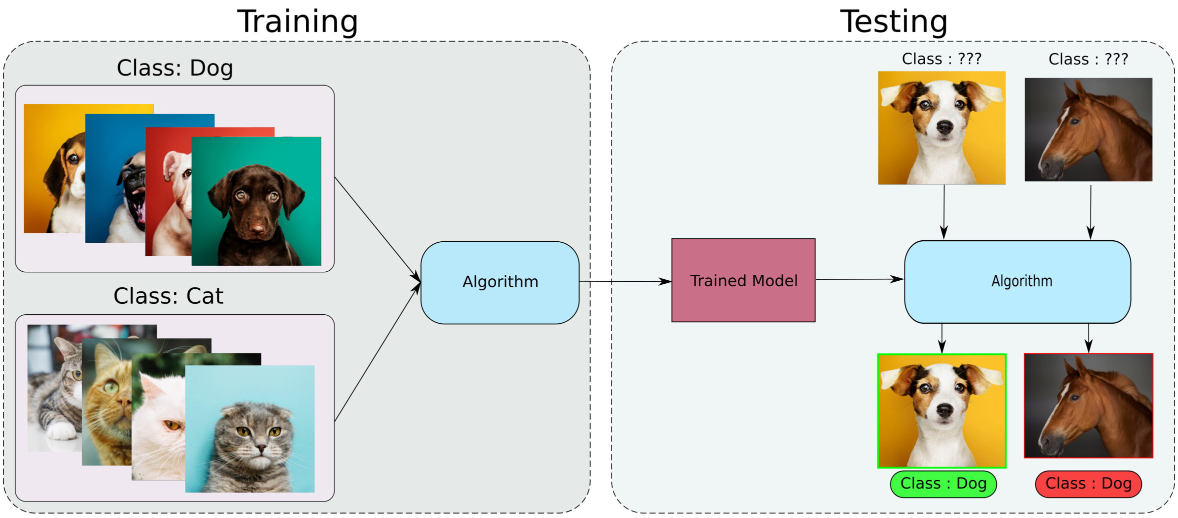


Figure 2.6: Representation of a closed set algorithm for classifying images in dogs and cats.

scenario should be used during the entire flow of the method, training, and testing, so, for the method to work, the test images must belong to one of the learned classes. This idea is used from scene classification to semantic segmentation.

2.4.2 Open Set Scenario

The open set scenario exists to be an alternative to the closed set scenario in the not ideal contexts described above. Instead of limiting the use of the algorithms to the context in which all the classes in testing already seemed during training, the open set allows a relaxation, covering more real-world problems.

According to Scheirer et al. [2013], an open set scenario has classes, not just images, in testing that were not seen in training. This way, the examples evaluated during the test phase can belong to many classes, learned or not by the method. The only distinction here is that the algorithm does not need to know all the classes, or predict all the classes during the test, it only needs to understand that some examples may not belong to any known class.

Following the cats and dogs example utilized in subsection 2.4.1, a method developed using the open set scenario as a concept also only sees images containing examples of cats and dogs during the training. The big difference is that during the test phase, the method is supposed to work on the ideal closed set scenario but also the non ideal context. This means that the test examples can belong to the class cat, and the method is expected to predict the class cat, but it also may belong to the class horse and the method is expected to not predict either cat or dog, but unknown instead. Similarly to

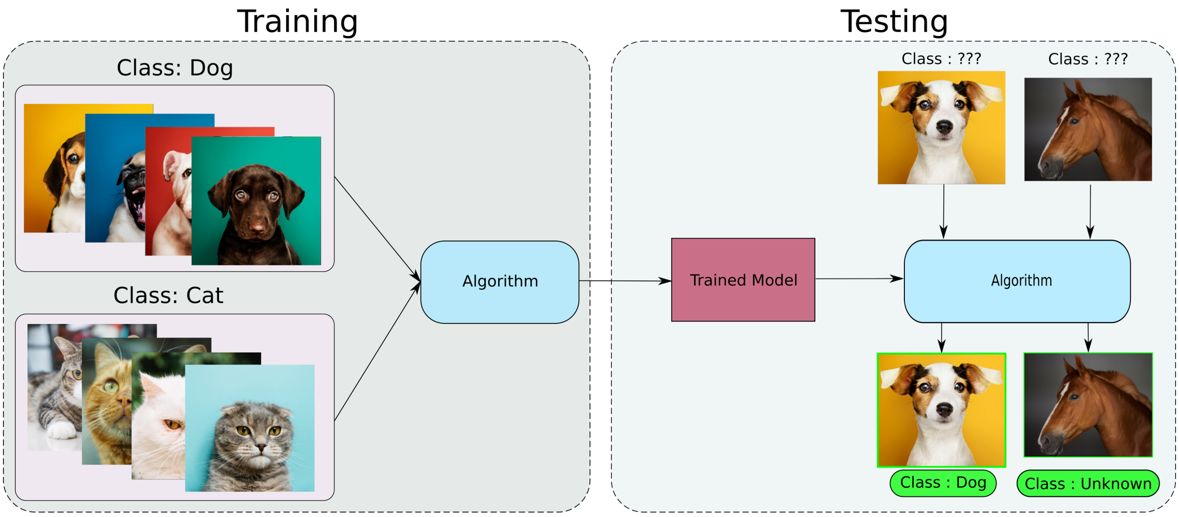


Figure 2.7: Representation of an open set algorithm for classifying images in dogs and cats.

Figure 2.6, Figure 2.7 represents the cases in which the image being evaluated is from one of the classes seen during training and when it isn't.

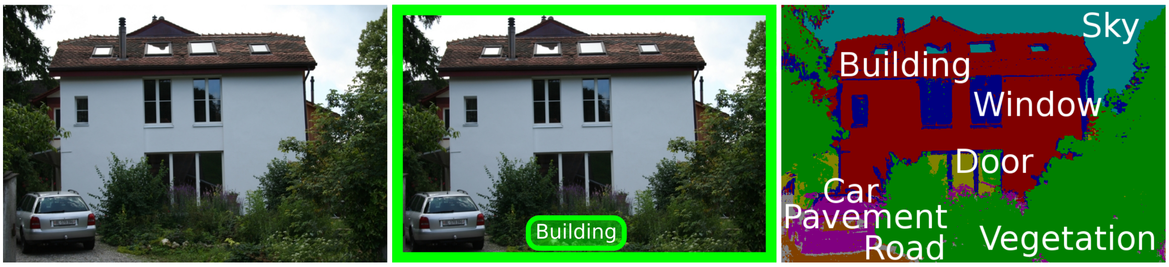
It is important to clarify that the method that uses the concept of open set scenario does not need to predict the classes not learned, it only needs to know and acknowledge that it is not a learned class. This means that, on the example, an image of a horse, a bird or a person would all receive the same label, unknown, since they do not belong to learned classes, cat and dog.

It is reasonable to think that it is possible to use the closed set concept in the open set scenario by gathering examples of all the possible classes, but the number and variety of those are not well modeled. But in reality, a method cannot know all the possible classes that may appear during the test phase in an uncontrolled ambient.

2.5 Image Classification and Semantic Segmentation

Besides the scenario utilized by the method, another important characteristic is the type of task it is trying to achieve. The task can be scene classification, recognition, segmentation, instance classification, semantic segmentation, and others.

One of the most common tasks done in deep learning and machine learning methods is scene classification. The scene classification is based on the idea that an algorithm should be able to predict what an image contains or represents. It does not need to know necessarily wherein an image is an object or if there are more objects in the same



(a) Input RGB Image (b) Scene Classification (c) Semantic Segmentation

Figure 2.8: Example of an image processed by an scene classification and semantic segmentation algorithm Hazirbas [2014].

image, it only needs to have a general idea of the content of the image.

In contrast to scene classification, semantic segmentation, is defined by Garcia-Garcia et al. [2017] as a fine-grained inference that has a goal to make dense predictions inferring labels for every pixel instead of making a prediction for the input as a whole. The resulting prediction is a map in which each pixel is labeled with the class of its enclosing object or region. The Figure 2.8 shows the difference between scene classification and semantic segmentation.

It is possible to interpret the semantic segmentation task as a scene classification in which the input is not a whole image, but each pixel. This way, the algorithms need to infer what is the content of each pixel in the image. This task can be used to find out which objects are present in an image, where they are spatially, their shapes and other detailed information.

Chapter 3

Related Work

Due to the demand of open set techniques, adapting closed set ones to this other scenario became as popular as creating new methods. This chapter presents background knowledge and a literature review on open set based methods for image classification, described in section 3.1 and also, on section 3.2, a literature review on semantic segmentation techniques.

Even though the work presented in Pham et al. [2018] is not deeply explored in this section, it is valid to mention it. The reason it was not chosen to be described as related work is the fact that the paper method uses the concept of open set scenarios on the semantic instance segmentation task. The result, in some cases, can seem close to the one obtained by the proposed methods in this dissertation, but the concept and task goal are different and the semantic instance segmentation needs to be designed and tweaked to be comparable with the open set semantic segmentation.

3.1 Open Set Classification

To better compare the existing approaches, Table 3.1 brings their key aspects summarized, which are detailed in the following subsections. Analyzing Table 3.1, we notice that each of the newer techniques tries to fulfill a gap left by the technique before using different methodologies.

3.1.1 Towards Open Set Recognition

In Scheirer et al. [2013] the authors present the idea of the Open Set scenario and describe a version of a support vector machine (SVM) developed to classify scenes in this scenario. The choice of using SVM was made because it has various alluring

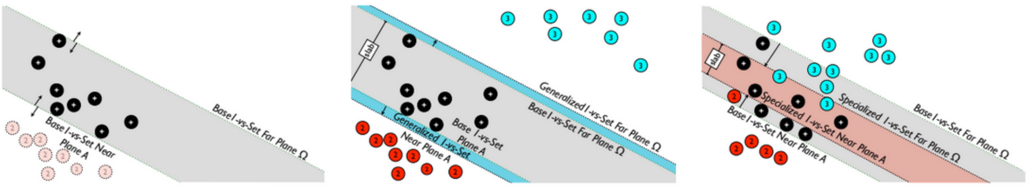
Table 3.1: Open set techniques comparison

Subsection	Technique	Approach	Threshold	SVM	Deep Learning
3.1.1	1-vs-Set Machine [Scheirer et al., 2013]	1-vs-All	×	✓	×
3.1.2	NNO [Bendale and Boult, 2015]	Multiclass	✓	×	×
3.1.3	EVM [Rudd et al., 2015]	Multiclass	✓	×	×
3.1.4	OpenMax [Bendale and Boult, 2016]	1-vs-All	×	×	✓
3.1.5	OSNNcv [Mendes et al., 2017]	Multiclass	×	×	×
3.1.5	OSNN [Mendes et al., 2017]	Multiclass	✓	×	×
3.1.6	G-OpenMax [Ge et al., 2017]	1-vs-All	×	×	✓

characteristics that can help in this scenario: its answers are global and unique; it has a basic geometric understanding, and it does not rely upon the dimensionality of the information space.

Their method consists of a 1-vs-Set machine that adjusts the unknown classes by getting a center edge around the choice limit A from the base SVM, specializing the subsequent half-space by including another plane, and after that, generalizing or specializing the two planes to upgrade experimental and open space risk. It is known where positive training samples exist and that in "open space" (space far away from known instances) there is no evidence for labeling as the class of interest.

The base linear 1-vs-Set machine, shown in Figure 3.1(a), will just touch the extremes of the positive examples. It then turns to greedy optimization to move the planes simultaneously. If all negative training classes are outside that slab, the over-specialization risk terms will counteract the open space risk term and move the planes to generalize, as in Figure 3.1(b). If the negative examples overlap the base slab, the overspecialization risk will be 1, and the over-generalization risk term and probably the empirical risk term will require the planes to move inward, as in Figure 3.1(c). When addressing open set problems, the risk of the unknown is reduced by specializing the slab to be closer to the positive examples.


Base l-vs-Set Near Plane A
Base l-vs-Set Far Plane Ω
slab
Generalized l-vs-Set Far Plane Ω
Base l-vs-Set Far Plane Ω
Base l-vs-Set Plane A
Generalized l-vs-Set Near Plane A
Specialized l-vs-Set Far Plane Ω
Specialized l-vs-Set Near Plane A
Base l-vs-Set Near Plane A
Base l-vs-Set Far Plane Ω

(a) Base Linear 1-v-Set Machine (b) Generalization (c) Specialization

Figure 3.1: Example of linear 1-vs-Set Machine showing the (a) base slab for both the 1-class and binary formulations, where the second class is only considered in the latter case (b) the generalization, and (c) the specialization operators. Blue refers to generalization, red for specialization and gray for the base linear 1-vs-Set Machine [Scheirer et al., 2013].

3.1.2 Towards Open World Recognition

Going a step further, Bendale and Boult [2015] present and develop a technique for an "open-world", a recognition system that should update new object categories and be robust to these unseen groups, besides, to have minimum downtime. To do so, its first step is to continuously detect novel classes; The second is to update the system to include these new classes when novel inputs are found, as represented in Figure 3.2.

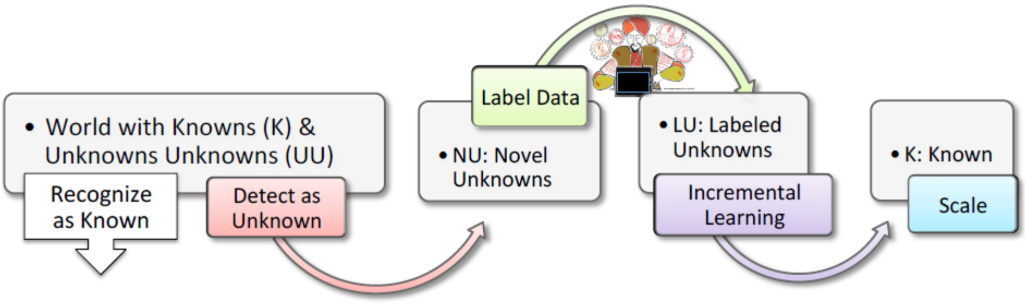


Figure 3.2: Open world recognition system which is able to recognize objects from known and unknown classes. The novel unknown classes should be collected and labeled, adapting itself and learning in an open Bendale and Boult [2015].

In particular, in open-world recognition, the Law of Total Probability and Bayes' Law cannot be directly applied and hence cannot be used to normalize scores. As time goes by and odd classes are added, the normalization factors and probabilities keep changing, therefore Nearest Class Mean type Algorithms (NCM) is not suitable for open set recognition.

Bendale and Boult [2015] show how to extend NCM to a Nearest Non-Outlier (NNO) algorithm that evolves model efficiently adding object categories incrementally while detecting outliers and managing open space risk.

3.1.3 Extreme Value Machine

While *Towards Open World Recognition* extends the previous methods including the incremental learning process Bendale and Boult [2015], the proposed Nearest Non-Outlier (NNO) algorithm updates the model efficiently adding object categories incrementally while detecting outliers and managing open space risk.

Unfortunately, NNO lacks strong theoretical grounding, using thresholded values for decision function and ignoring distribution information. To deal with that, a model called Extreme Value Machine (EVM) is presented Rudd et al. [2015]. The model is derived from *extreme value theory*, which calculates the radial probability of inclusion of a point in a class. A compact probabilist representation using extreme vectors (EV) is achieved using points and distributions that best summarize each class.

In EVM, a training set is represented by a set of extreme vectors, each of them associated with the Probability of Sample Inclusion (Ψ) derived from EVT. The Ψ term is modeled in terms of the distribution of sample half-distances relative to a reference point. For each positive reference point, we get half distances to the nearest negative samples - as in figure 3.3.

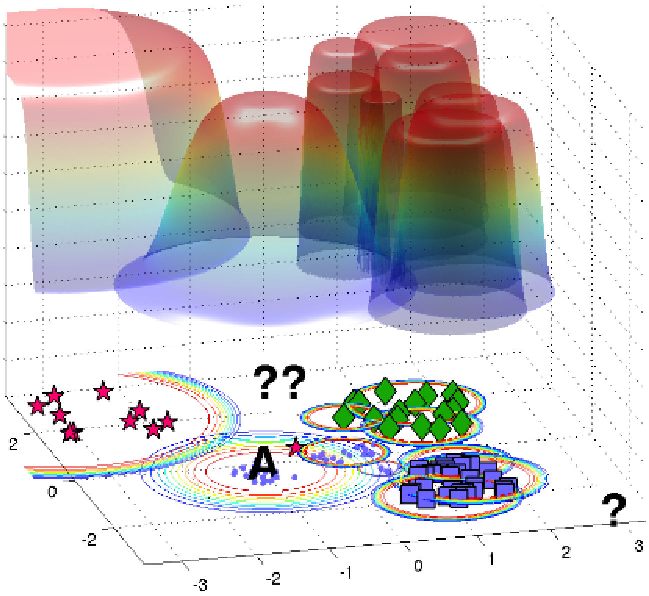


Figure 3.3: EVM algorithm trained on four classes. The colors in the rings show a probability for each extreme vector chosen by the algorithm. EVM supports open set recognition and can reject the three "?" inputs that lie beyond the support of the training set as "unknown" Rudd et al. [2015] .

As EVT theorem states, the minimal values of margin for a given point is given by Weibull distribution. This way, the probability that a sample x' is included in the boundary estimated by x_i is defined as:

$$\Psi(x_i, x', k_i, \lambda_i) = \exp^{-\left(\frac{\|x_i - x'\|}{\lambda_i}\right)^{k_i}} \tag{3.1}$$

where $\| x_i - x' \|$ is the distance of x' from sample x_i, x', k_i, λ_i are parameters of Weibull distribution.

Given a point x' in space, the probability that x' belongs to a class is defined as the max probability $\hat{P}(C_l|x')$ between the known classes compared to a threshold(δ). If $\hat{P}(C_l|x') \geq \delta$ then x' belongs to class C_l, otherwise the point is classified as "unknown".

3.1.4 Towards Open Set Deep Networks

Looking for a deep learning solution, Bendale and Boult [2016] shows a new model, called OpenMax, that represents an alternative for the SoftMax function as the final layer of the network, which estimates the probability of an input being from an unknown class. Reducing the number of errors made by a deep network when given fooling generated images.

Figure 3.4 shows an example of the use of the model described in Bendale and Boult [2016], comparing the activation maps of the features. This figure also demonstrates the characteristics of the method to be linked to scene classification, since when the image has cropped the values of the activation maps can drastically change, modifying the results predicted.

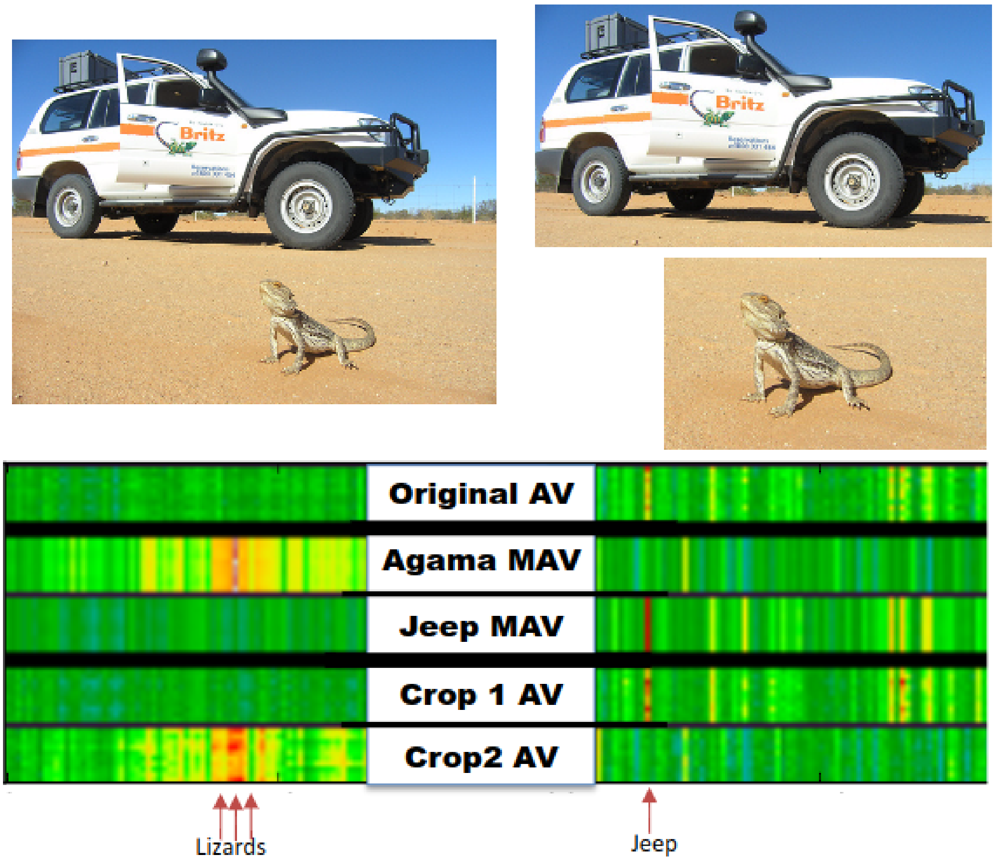


Figure 3.4: OpenMax predicting failure during training. The official class is agama but the MAV for agama is rejected and the highest scoring class is jeep with probability 0.26. However, cropping out image regions can find windows where the agama and the Jeep are well detected, with probability 0.32 and 0.21 respectively [Bendale and Boult, 2016].

By dropping the restriction for the probability for known classes to sum to 1, and rejecting inputs far from known inputs, OpenMax can formally handle unknown/unseen classes during operation. This new layer uses the scores from the penultimate layer of deep networks (the fully connected layer before SoftMax) to estimate if the input is far from known training data. The approach is based on the fact that the values from this layer (Activation Vector), are not an independent per-class score estimate, but rather they provide a distribution of what classes are related.

3.1.5 Nearest Neighbors Distance Ratio

Using a shallow approach, Mendes et al. [2017] proposes a method named Open Set NN (OSNN) and a variation called OSNNcv, both can recognize samples from unknown classes during training time and outperform other approaches in the literature. OSNNcv method verifies if the test sample can be classified as unknown, checking if the two closest samples are from different classes. The OSNN method uses the ratio of similarity scores to the two most similar classes by applying a threshold on it. One of the advantages of this approach is that it is inherently multiclass, which means that the computational time is not affected as the number of classes for training increases.

In the interest of finding the best value for the threshold in an open set scenario, a *parameter optimization* is performed. For this purpose, a simulation of an open set environment is established. For that, a training set is created with half of the known classes. In addition, a validation set receives the other half of known classes and also all instances of unknown classes. After all, the threshold is based on the accuracy of the validation set. Details of this operation can be seen in figure 3.5.

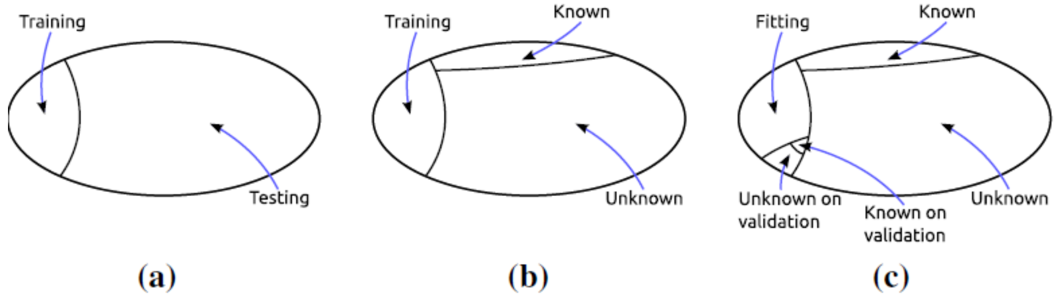


Figure 3.5: Scheme of data partitioning for the experiments and the parameter optimization of the OSNN. **a** A dataset is divided into training and testing sets. **b** Most of the samples in testing set are unknown . **c** Partitioning of the training set by simulating an open set scenario.[Mendes et al., 2017]

OSNN has the characteristic of being inherently multi-class (non-binary-based), differently from other state-of-the-art approaches. Usually these approaches lose some efficiency when the number of classes is increased, while the method proposed by Mendes et al. [2017] is not affected by the number of classes.

3.1.6 Generative OpenMax for Multi-Class Open Set Classification

Generative OpenMax (G-OpenMax) Ge et al. [2017], extends OpenMax Scheirer et al. [2013] by providing explicit probability estimation over unknown categories. This is done by using generative adversarial networks (GANs), which initially are a technique to estimate models via an adversarial process between two neural networks and are Scheirer et al. [2013] uses to generate the unknown classes. The synthetic samples are created by mixture distributions of known classes in space.

That is, while OpenMax estimates the pseudo probability of unknown class using an aggregating calibrated score from known classes, the G-OpenMax which is an intuitive solution, directly estimates the probability of unknown class. Being performed through synthetic images as an extra training label apart from known labels.

The main difference between OpenMax and G-OpenMax is show in 3.6. In G-OpenMax the network Net^G is trained with an extra class where the extra images come from the generator G to represent the unknown class.

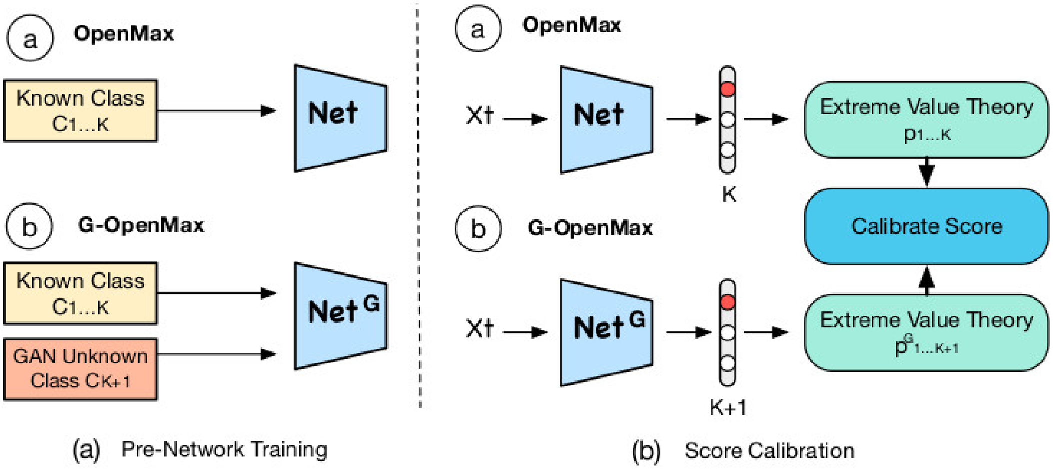


Figure 3.6: a) Illustrates the pre-training process of Net and Net^G. GAN-based synthetic images are used as an extra training label. b) Explains the difference between score calibration in normal OpenMax and G-OpenMax [Ge et al., 2017].

3.2 Semantic Segmentation

The subsections presented here bring a brief overview of the state-of-art methods of semantic segmentation. All of these methods were developed for the closed set scenario.

3.2.1 Fully Convolutional Networks for Semantic Segmentation

Besides the open set characteristics of this dissertation, the state of art algorithms for semantic segmentation also needes to be understood . With this in mind, Long et al. [2015] shows that convolutional networks trained end-to-end, pixels-to-pixels, exceed the state-of-the-art in semantic segmentation.

To do so, the proposed algorithm builds a fully convolutional network (FCN) that receives an arbitrary size input and outputs a correspondingly-sized output with inference and learning.

The system is divided into two steps: first they adapt contemporary classification networks as AlexNet, the VGG net, and GoogLeNet into fully convolutional networks and transfer learned representations by fine-tuning to the segmentation task; second step is to use a skip architecture that combines semantic information from a deep, coarse layer with appearance information from a shallow, fine layer to produce accurate and detailed segmentation.

3.2.2 Conditional Random Fields as Recurrent Neural Networks

Zheng et al. [2015] proposed that one central issue in the use of deep learning techniques for image recognition to tackle pixel-level labeling tasks, semantic segmentation, is the limited capacity of deep learning techniques to delineate visual objects.

As a solution, Zheng et al. [2015] introduces a convolutional neural network that combines Convolutional Neural Networks (CNN) and Conditional Random Fields (CRFs) probabilistic graphical modeling. To do so, the proposed system is divided in two main parts, the first is to formulate the CRF with Gaussian pairwise potentials and mean-field approximate inference as Recurrent Neural Networks, the second part is to plug in the CRF as a part of a CNN to obtain a final deep network that has desirable properties of both. The Figure 3.7 presented an example of the approach described in Zheng et al. [2015].

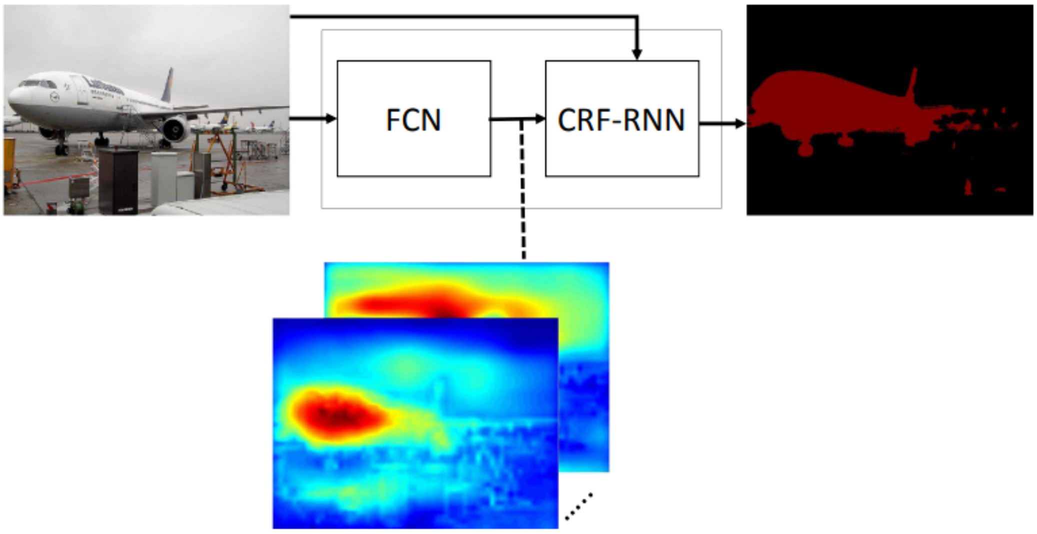


Figure 3.7: The End-to-end Trainable Network. Schematic visualization of the full network which consists of a CNN and the CNN-CRF network [Zheng et al., 2015]

One big advantage of this methodology is that it makes possible to train the whole deep network end-to-end with the usual back-propagation algorithm, avoiding offline post-processing methods for object delineation.

3.2.3 U-Net: Convolutional Networks for Biomedical Image Segmentation

Even though the U-Net method was designed for biomedical image segmentation it is one of the most used methods in the semantic segmentation area on any domain, alongside the FCN, used in this dissertation. In Ronneberger et al. [2015], the authors present a network and training strategy that relies on the strong use of data augmentation to use the available annotated samples more efficiently.

The architecture consists of a contracting path and an expansive path, as represented in Figure 3.8. The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3×3 convolutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2×2 max pooling operation with stride 2 for downsampling. At each downsampling step, we double the number of feature channels.

Every step in the expansive path consists of an upsampling of the feature map followed by a 2×2 convolution (“up-convolution”) that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3×3 convolutions, each followed by a ReLU. The cropping

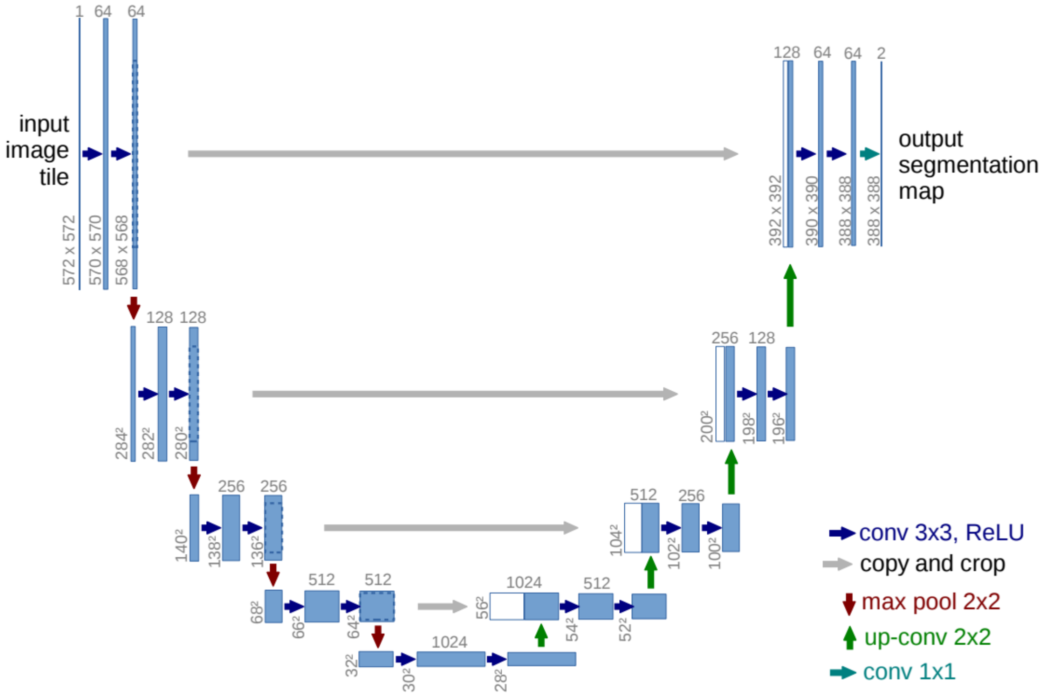


Figure 3.8: U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. [Ronneberger et al., 2015]

is necessary due to the loss of border pixels in every convolution. At the final layer, a 1×1 convolution is used to map each 64-component feature vector to the desired number of classes. In total the network has 23 convolutional layers.

3.2.4 Fully convolutional networks for dense semantic labeling of high-resolution aerial imagery

Using the idea that higher resolution remote sensing imagery facilitates a transition from land-use classification to object-level scene understanding, Sherrah [2016] presents a model that does not rely purely on spectral content, but it also uses appearance-based image features.

For this, the authors use deep convolutional neural networks applied to semantic labeling of high-resolution remote sensing data, adapting fully convolutional networks (FCNs) to overhead data. This approach is described as effective as in other domains.

More specifically, full-resolution labeling is inferred using a deep FCN with no downsampling, obviating the need for deconvolution or interpolation. To make better use of image features, a pre-trained CNN is fine-tuned on remote sensing data in a hybrid network context, resulting in superior results compared to a network trained from scratch. The architecture of this hybrid network is presented in Figure 3.9.

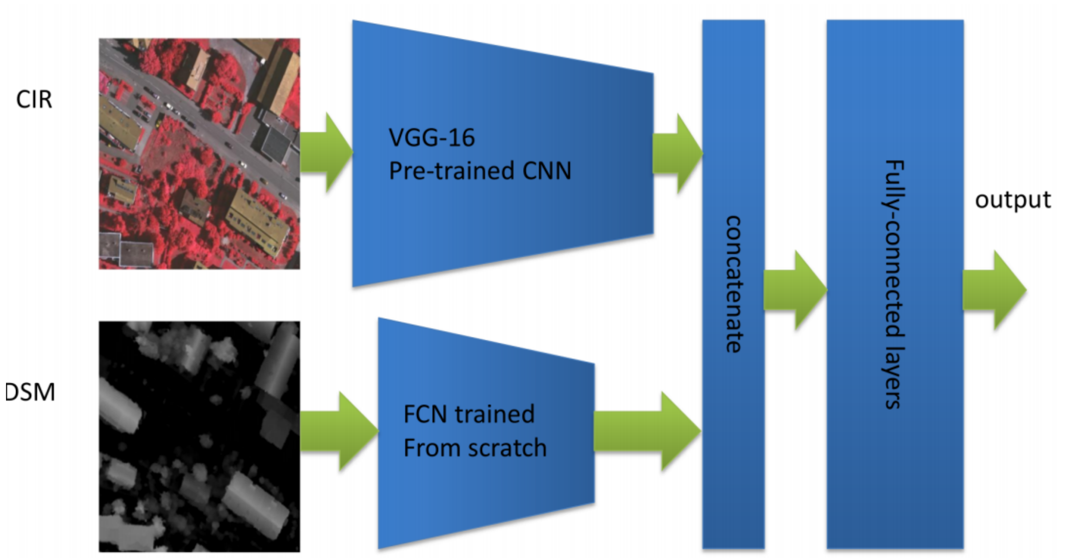


Figure 3.9: Schematic of the hybrid network architecture, combining pre-trained image features with DSM features trained from scratch. [Sherrah, 2016]

3.2.5 Semantic segmentation of earth observation data using multimodal and multi-scale deep networks

The work present in Audebert et al. [2016] investigates the use of deep fully convolutional neural networks (DFCNN) for pixel-wise scene labeling of Earth Observation images. In a more specific manner, the authors trained a variant of the SegNet architecture on remote sensing data over an urban area using diverse strategies for performing accurate semantic segmentation. Since the overall architecture used in Audebert et al. [2016] is based on the SegNet model, the Figure 3.10 is similar, in interpretation, to the one presented on in Badrinarayanan et al. [2017].

The authors of the paper were able to transfer efficiently a DFCNN from images taken by smartphones and compact cameras, registering pictures on the RGB channel, to remote sensing images. They also introduced a multi-kernel convolutional layer for fast aggregation of predictions at multiple scales and performed a data fusion from optical and laser sensors using residual correction.

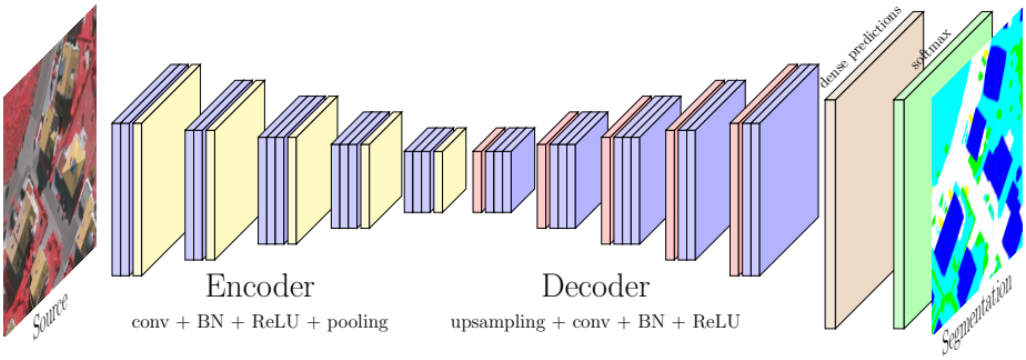


Figure 3.10: Illustration of the SegNet architecture applied to Earth Observation data. [Audebert et al., 2016]

3.2.6 SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

A novel and practical deep fully convolutional neural network architecture for semantic pixel-wise segmentation termed SegNet is presented in Badrinarayanan et al. [2017]. The core trainable segmentation engine consists of an encoder network, a corresponding decoder network followed by a pixel-wise classification layer. The architecture of the encoder network is topologically identical to the 13 convolutional layers in the VGG16 network. Figure 3.11 show the approach introduced in Badrinarayanan et al. [2017]

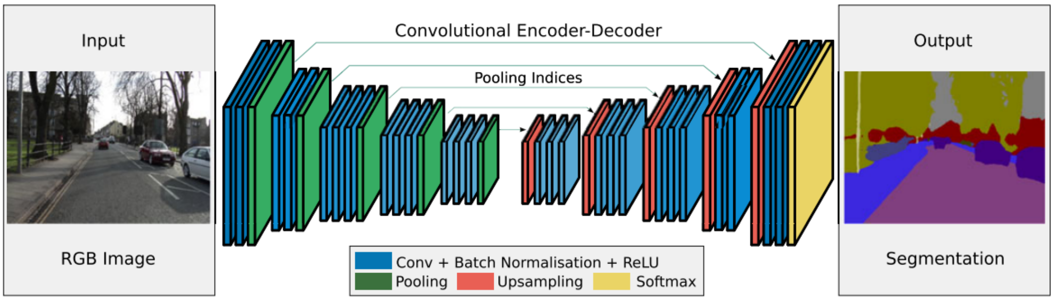


Figure 3.11: An illustration of the SegNet architecture. There are no fully connected layers and hence it is only convolutional. A decoder upsamples its input using the transferred pool indices from its encoder to produce a sparse feature map(s). It then performs convolution with a trainable filter bank to densify the feature map. The final decoder output feature maps are fed to a soft-max classifier for pixel-wise classification. [Badrinarayanan et al., 2017]

The role of the decoder network is to map the low-resolution encoder feature maps to full input resolution feature maps for pixel-wise classification. The novelty of SegNet is in the manner in which the decoder upsamples its lower resolution input feature map(s). Specifically, the decoder uses pooling indices computed in the max-pooling step of the corresponding encoder to perform non-linear upsampling. This eliminates the need for learning to upsample. The upsampled maps are sparse and are then convolved with trainable filters to produce dense feature maps.

SegNet was primarily motivated by scene understanding applications. Hence, it is designed to be efficient both in terms of memory and computational time during inference. It is also significantly smaller in the number of trainable parameters than other competing architectures and can be trained end-to-end using stochastic gradient descent.

3.2.7 DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs

Another work that uses CRFs is the one by Chen et al. [2017], but their approaches differ from Zheng et al. [2015]. The proposed technique consists of three main ideas. The first is the use of atrous convolution, convolution with upsampled filters to explicitly control the resolution at which feature responses are computed within Deep CNNs. Another use of atrous convolution is to enlarge the field of view of filters to incorporate larger context without increasing the number of parameters.

The second part of the method is an atrous spatial pyramid pooling (ASPP) to robustly segment objects at multiple scales, using an incoming convolutional feature layer with filters at multiple sampling rates and effective fields-of-views.

Finally, the third is the combination of Deep Convolutional Neural Networks (DCNNs) and probabilistic graphical models to improve the localization of object boundaries. The combination of max-pooling and downsampling in DCNNs achieves invariance but has the price of decreasing the accuracy of object location. To solve this problem, the authors, combine the results of the final layer with a fully connected CRF, improving localization performance. Figure 3.12 represents the architecture used in the paper.

3.2.8 Classification with an edge: Improving semantic image segmentation with boundary detection

In Marmanis et al. [2018], the authors present an end-to-end trainable deep convolutional neural network (DCNN) for semantic segmentation with built-in awareness of semantically meaningful boundaries. One of the reasons that semantic segmenta-

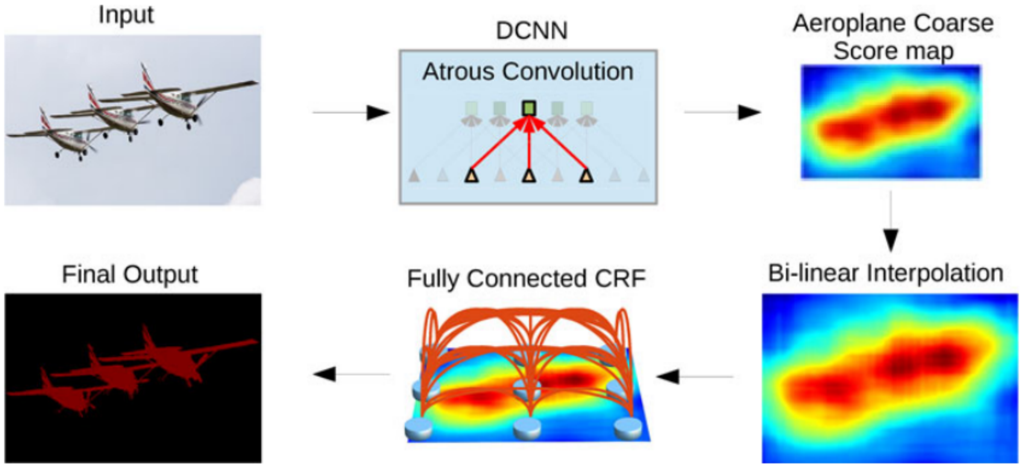


Figure 3.12: . Model illustration. A deep convolutional neural network such as VGG-16 or ResNet-101 is employed in a fully convolutional fashion, using atrous convolution to reduce the degree of signal downsampling (from 32x down 8x). A bilinear interpolation stage enlarges the feature maps to the original image resolution. A fully connected CRF is then applied to refine the segmentation result and better capture the object boundaries. [Chen et al., 2017]

tion networks work on remote sensing areas is that deep networks learn to accumulate contextual information over very large receptive fields.

However, it usually has a cost, since the associated loss of effective spatial resolution washes out high-frequency details and leads to blurry object boundaries. Observing this, the Marmanis et al. [2018] proposes a combination of semantic segmentation with semantically informed edge detection, thus making class boundaries explicit in the model.

To do so, the approach defined was to construct a comparatively simple, memory-efficient model by adding boundary detection to the SEGNET encoder-decoder architecture. It also included boundary detection in FCN-type models and set up a high-end classifier ensemble. Figure 3.13 shows a representation of the class-boundary network proposed.

3.3 Gaps Explored by the Proposed Methods

The open set methodologies proposed in this dissertation differ from all of those mentioned above. Even though the scenario considered is the same, the ones here described were developed for scene classification, while the technique proposed is focused on semantic segmentation.

In a similar way, when observing the state-of-art semantic segmentation algo-

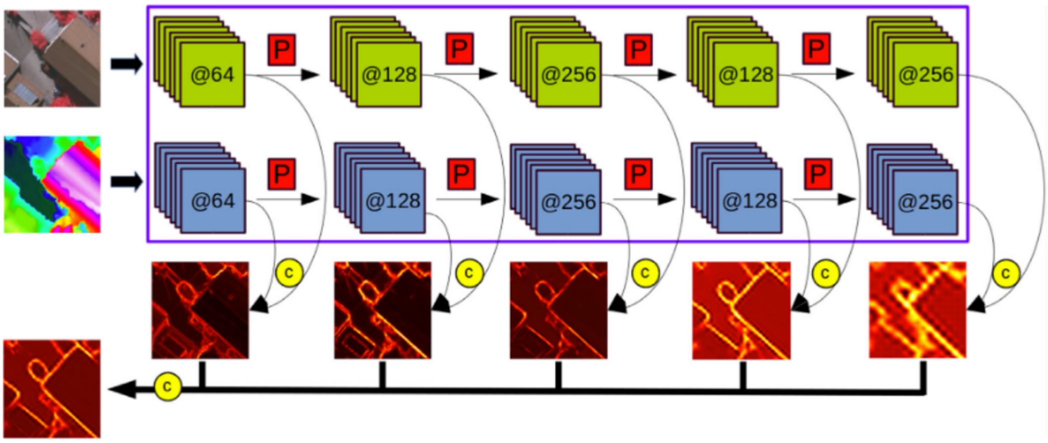


Figure 3.13: The Class-Boundary network. Colour and height are processed in separate streams. Before each pooling level (red squares) the network outputs a set of scale-dependent class-boundaries, which are fused into the final multi-scale boundary prediction. Yellow circles denote concatenation of feature maps. [Marmanis et al., 2018]

rithms the gap becomes clear, no method can classify each pixel of the input into a known or unknown class, as the methods proposed in this dissertation do.

It is also valid to note that most of the methods here described can be adapted to work on the open set scenario. The adaptation can be in the form of aggregating a classification layer to perform the open set prediction. This way the core of the network can remain the same. There are other possibilities, but they may require a change in the architecture of the original network in a manner that can hurt the accuracy of the method when applied to closed set scenarios.

Chapter 4

Methodology

This chapter describes in detail the two proposed methods, they are: (1)the OpenPixel method presented in section 4.1, and (2) the method OpenFCN presented in the section 4.2. Both methods are based on existing closed set methods for semantic segmentation, but the OpenPixel technique does a pixel-level classification and applies a threshold to differentiate between known and unknown classes while OpenFCN uses the relation between activation maps of each known class and the FCN network to perform the task.

4.1 OpenPixel: Pixel-based open set classification

One of the methods developed and presented in this dissertation is an open set adaptation from the closed set Pixelwise algorithm proposed in Nogueira et al. [2016]. The Pixelwise approach consists of the individual treatment of all the pixels present in the images. Context windows, however, consist of 55x55 crops with the central pixel representing the crop class. In this way, context windows have been created for all pixels in the image, and each of these windows is used as the input of the network.

A context window is a group of pixels surround the main pixel being evaluated that are used for the network to understand the existing context. This group, usually is set as a square matrix, in which, the size can be varied. The Figure 4.1 represents a context window, the green labels mark the pixels used for context and the red label notes the pixel being evaluated.

Thus, the proposed neural network can individually classify all the pixels of the image, taking into account the nearest neighbors, and through that, it is possible to obtain thresholds of the desired regions.

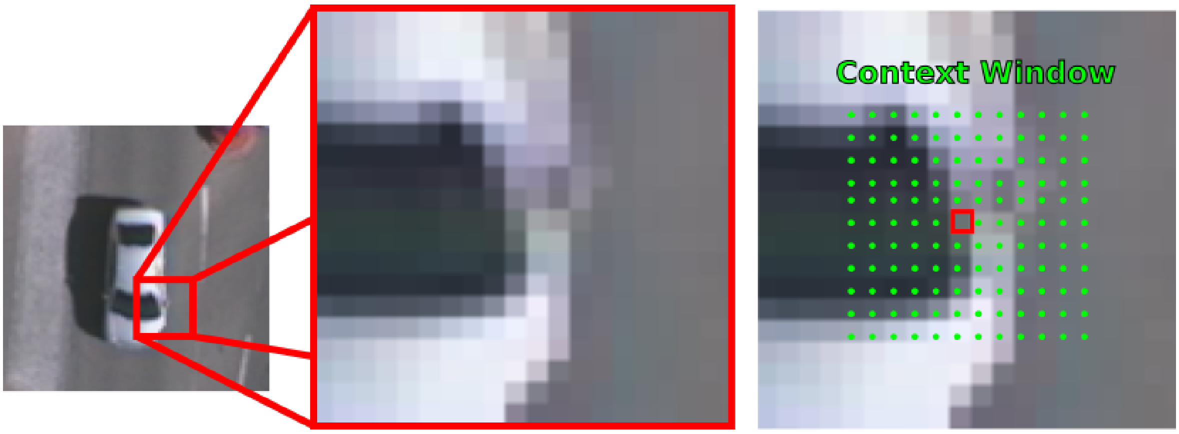


Figure 4.1: Representation of a context window,the green labels mark the pixels used for context and the red label notes the pixel being evaluated by the network.

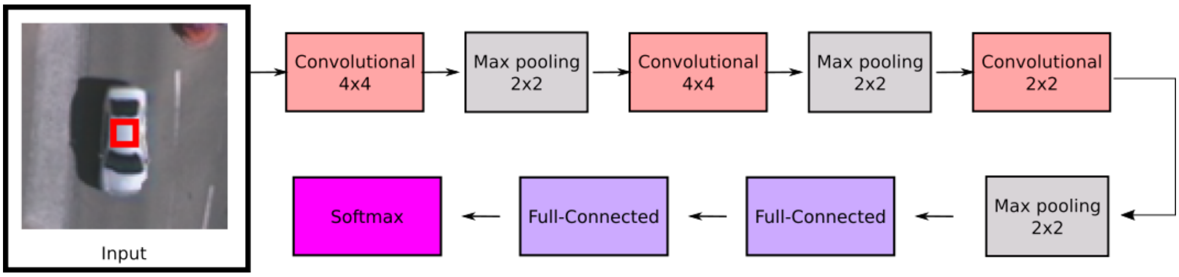


Figure 4.2: Simplified architecture view of the Closed Set Pixelwise method.

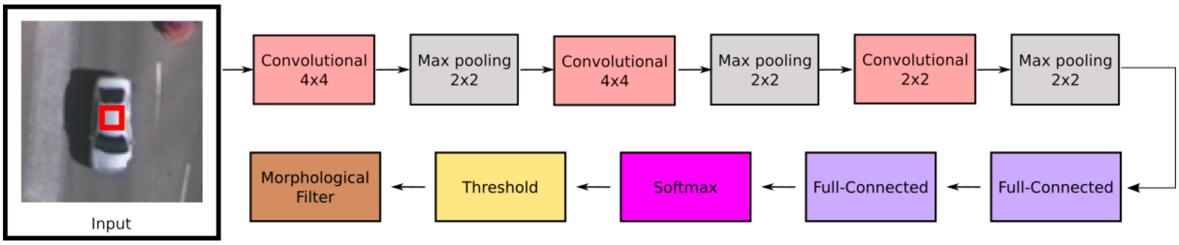


Figure 4.3: Simplified architecture view of the proposed OpenPixel method.

Figure 4.2 illustrates the architecture of the convolutional neural network used in this work, Pixelwise, for the Closed Set. To achieve the OpenPixel, we added an extra layer for thresholding the results as well as a layer to filter false positives, as in Figure 4.3. This network receives an image as input, this image passes through 3 layers of convolutions, 3 layers max pooling, 2 layers fully connected, to then reach the output layer, which is responsible for sorting the central pixel of the input image. Unlike traditional neural networks, CNNs have some peculiarities. The purpose and functioning of these peculiarities will be explained below.

The main purpose of the convolution layers is to extract relevant features contained in the input image. This extraction is made by applying specific filters, updated during training, on small crops of these images. In the case of the first convolution layer, 64 different filters with a 4x4 dimension are applied to the image received as input with a stride of 1.

Immediately after the convolutions are applied, an activation function on the outputs generated by the convolution. In this architecture, the Rectified linear unit (ReLU) was used. The purpose of this application is to introduce linearity to the data since the convolutions apply exclusively linear operations on them.

Finally, after applying the ReLU, the resulting data passes through a layer of pooling. In the case of the network, max-pooling was used. The main purpose of pooling layers is to perform a spatial reduction in the resulting matrix. In the network used in this work 3 downsamplings were made using 3 layers of max pooling with filters 2x2 and stride 2.

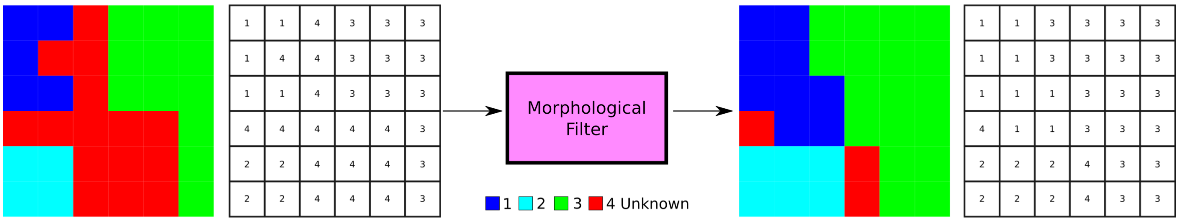
4.1.1 Thresholding

The architecture of OpenPixel is the same as that of Pixelwise for the closed set scenario, the difference consists in applying a threshold of certainty in the classification given by the softmax. To do so, a pixel with a value of certainty given by the softmax for a certain class that exceeds that class threshold is labeled as belonging to the class, but if the value is inferior to the determined threshold, the pixel is classified as unknown. As the value of probability given by the softmax varies between 0 and 1, the possible values of threshold also vary between 0 and 1. The representation below presents the threshold control done during the classification phase, in which x is the pixel being evaluated, $P(x_c)$ is the probability of x belonging to class c, given by softmax, and T is the threshold used.

$$\begin{cases} x = c, & if\ P(x_c) >= T \\ x = unknown, & if\ P(x_c) < T \end{cases}$$

The values for thresholds are dynamically obtained during the training phase. For a given application, the network runs against a Validation set with multiples possible values of threshold per class, the results that have an increase in accuracy are determined as the best for use on the test set of the application.

The problem of using an approach based on the threshold of uncertainty, is that neighbouring areas, the areas where the context window has information from more than one class, usually have a low probability of belonging to any one of the known class, being labeled as unknown, even though the ground truth of those areas has no distinction between a boundary area or not.



1	1	4	3	3	3
1	4	4	3	3	3
1	1	4	3	3	3
4	4	4	4	4	3
2	2	4	4	4	3
2	2	4	4	4	3

1	1	3	3	3	3
1	1	3	3	3	3
1	1	1	3	3	3
4	1	1	3	3	3
2	2	2	4	3	3
2	2	2	4	3	3

Figure 4.4: Example of application done by the morphological filter developed, in which classes 1 to 3 are known classes and class 4 is unknown.

To solve the problem found on those regions, a new method was develop. The Morph-OpenPixel technique is based on the OpenPixel described above, but with the advantage of solving the existing problem with the use of a morphological filter.

4.1.2 Morph-OpenPixel: OpenPixel With a Morphological Filter

After the result predicted at the softmax layer and applied the threshold by the network, a post-processing filter is applied, called the morphological filter. It is only applied at the pixels classified as unknown. This filter analyzes the neighbors of the pixel to determine if it should keep it as unknown class or it should be labeled with the same class as most of its neighbors.

The applied filter can be seen as an erosion done over the unknown class labels. For each pixel classified as unknown, the filter analyses its neighbors, to determine if it belongs to a border or if it is an inside pixel. If the pixel belongs to the border and has pixels belonging to other classes as neighbors, the pixel classification is exchanged to the class with a higher amount of pixels in the neighborhood. If all the pixels are from the unknown class, it means the central pixels are not on the border and it should remain labeled as unknown. Figure 4.4 shows an example of the application of the morphological filter used in this dissertation. The existence of this filter is to mitigate the false positives that can happen due to the boundaries of the known classes.

4.2 OpenFCN: Open Set Fully Convolutional Network

The original application of the OpenMax technique was to develop a method that could discern between images generated by computers and real images. The idea described on Bendale and Boult [2015] is that some images can be generated to fool classification

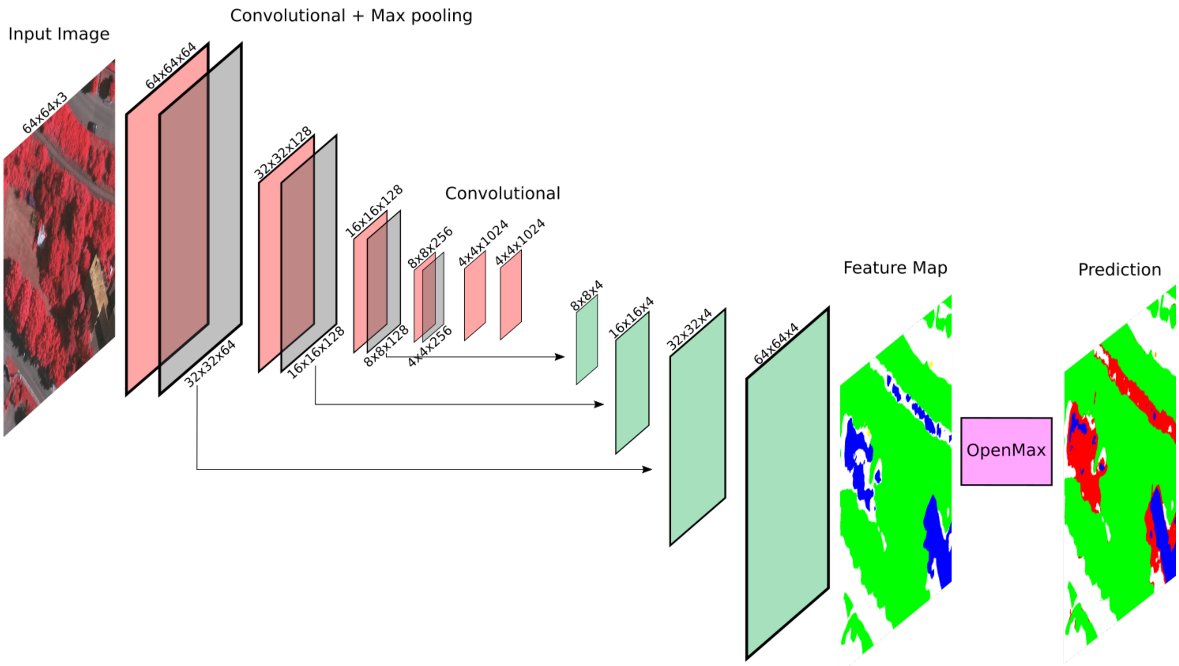


Figure 4.5: Simplified architecture view of the OpenFCN method.

networks, simulating features similar to real images. However when comparing the distribution of the classes and their similarities, the OpenMax method could be able to classify an image as a fooling image. In that case, this classification would set the image class as unknown, i.e. not belonging to any known class.

Similarly to most of the other open set techniques, Openmax was based on scene classification, only deciding the general label of the image, not pixel by pixel. This change brings a lot of new problems, and applications, to the method. So the method OpenFCN uses the same idea of an OpenMax layer, but adapted to an FCN, a pixel-by-pixel approach.

Figure 4.5 illustrates the architecture of the fully-convolutional neural network used in this work, FCN, adapted for the open set scenario. To achieve the OpenFCN, we substituted the classification layer, softmax, for the OpenMax. This network receives an image as input. This image passes through 6 layers of convolutions and 4 layers of max pooling, to then be deconvoluted 4 times, using information from earlier layers, and then reach the output layer, OpenMax, which is responsible for classifying each pixel and deciding if it belongs to a known or unknown class.

The architecture of the Fully Convolutional Networks (FCN) was introduced in Long et al. [2015]. In the paper, the authors show that convolutional networks by themselves, trained end-to-end, pixels-to-pixels, exceed the state-of-the-art in semantic segmentation. With this in mind, the main contribution is to build "fully convolutional" networks that take input of the arbitrary size and produce correspondingly-sized output with efficient inference and learning.

4.2.1 Convolutional Layers

This contribution is based on the idea that all of the layers in the network are convolutional layers. At the same time, FCN does not have any fully-connected layers at the end, which are typically used for classification. To suppress the traditional classification layer, FCN uses convolutional layers to classify each pixel in the image.

For this classification to work, the layer needs to have the same height and width as the input image, but the number of channels will be equal to the number of classes being evaluated. Aggregating this layer with a softmax or openmax probability function, it is possible to determine the most likely class for each pixel, semantic segmenting the input image.

4.2.2 Deconvolutional Layers

One common problem to the FCN architecture is the fact that intermediate layers, as expected, get deeper, but this also results in smaller layers, as striding and pooling reduce the height and width dimensions of the tensors. To solve this, FCN uses a deconvolution function, that can also be explained as a backward convolution. This layer is used to upsample the intermediate tensors, in a way that they can recover the width and height of the original input image.

The great advantage of using these deconvolution layers is the fact that they can be treated as a derivation of convolution layers, maintaining the characteristics of having weights that can be learned, facilitating the regulation of the network.

4.2.3 Combining Layers

A second problem that FCNs presented was an inaccuracy on the upsampling from the last convolutional layer, caused by the loss of spatial information during the downsampling steps in the network, affecting the correctness of the method. As a solution for this, the authors in Long et al. [2015] combined the upsampling layers of the network, aggregating spatial information existent before the downsampling stages. As tested on converting networks like AlexNet Krizhevsky et al. [2012], VGG Simonyan and Zisserman [2014], and GoogLeNet Szegedy et al. [2015] into FCNs, this approach solved the problem and improve the accuracy of the technique.

It is valid to observe that even though this work uses an FCN as the network to extract features for each pixel, the architecture of OpenFCN allows the use of any pixel-wise network. The method becomes flexible when observed that the Openmax layer is applied during the testing phase, being isolated from the training, facilitating and mitigating the number of adaptations needed to introduce a different pixel-wise network for feature extraction.

4.2.4 OpenMax layer

The classification of the pixels happens in the OpenMax layer. This layer is based on the distance between each evaluated pixel and the distributions of the known classes, being these Weibull distributions. Another way of describing the OpenMax layer is as a classification layer that uses similarity between the activation vector of the pixel, its features, and the activation vectors of the pixels belonging to known classes to label the pixels.

For the OpenMax layer to correctly classify the pixels, it needs to use the appropriate parameters for the Weibull distribution and calculation of distances. So the parameters that needed to be searched and evaluated related to the OpenMax layer were the Weibull tail size and the Alpha Rank. Parameters of Weibull distribution are estimated based on the distance between each correctly classified training example, obtaining a class-specific distance distribution. The exact length of tail for estimating parameters of Weibull distribution is determined during the parameter estimation phase over a small set of the data.

The Weibull is the distribution calculated for each class in the OpenMax process and its tail size is related to the flexibility of classifying a pixel class as unknown. The bigger is the value of the tail size, the more relaxed the method is, and more uncertainty pixels are classified as the open set class. The smaller the tail size value is, the more restricted is the classification, resulting in labeling as unknown only the class with high uncertainty.

In a different aspect, the alpha rank represents the number of top classes that are considered during the recalibration of the classes. Those top classes are defined as the classes with the biggest variance between the activation maps, in other words, the classes that better represent the diversity of the dataset.

To better understand the use of top classes, consider an animal dataset, containing different species of birds, dogs, and cats. Since different species of birds still have some similar characteristics, using all of the bird's classes as top classes could not represent a good diversity of the dataset. It would be preferred to use one class representing each of the groups: birds, dogs, and cats. All the classes not chosen as the top would be ranked lower in diversity and would have a high similarity to one of the top classes.

One way of choosing the alpha rank value and the top classes would be to use all of the classes in the dataset. This works for a small number of classes or if all the classes do not share similar characteristics. But in other cases, when observed that this choice would involve using more information in each recalibration, it could only lead to increasing the computational time need, without increasing the accuracy of the method.

4.2.5 Existing bottlenecks

One of the biggest problems that surface when adapting a method based on scene detection to semantic segmentation is the increase in computational time required to analyze each pixel from the image. Even though most of the networks and technologies used in this dissertation allowed parallel processing, there is still a bottleneck in the OpenMax method that reduces the efficiency of parallel computing of the method.

This bottleneck is on the stage in which the technique calculates the distances of each pixel to all the class distributions and, after allocating that pixel to a predicted class, redistributes the class distribution, needing to follow a sequential process of finding a pixel class, before evaluating the next one.

There are a lot of techniques that could be applied to accelerate the process, but at the same time, they could be weakening some of the bases of the methodology. One example, based on the spatial correlation between pixels, could be to run the Openmax method in parallel in arbitrary parts of the image, but this could make the method lose some of its accuracy, since it would not use the pixels predicted to recalculate the classes distributions in the same way as without this technique.

4.2.6 SLIC-OpenFCN: OpenFCN Using Super Pixels

The technique utilized in this work, to accelerate the method, was the use of superpixels. The difference between this technique and the described above is not to group arbitrary parts of the image, but to use their spatial correlation to find the best groups of neighbors pixels that could be treated as one. To segment, the input image in superpixels was used the Simple Linear Iterative Clustering (SLIC) method Achanta et al. [2012].

The SLIC generates superpixels by clustering pixels based on their color similarity and proximity in the image plane. To achieve this, the authors use the five-dimensional [labxy] space, in which [lab] is the pixel color vector in CIELAB color space, widely considered as perceptually uniform for small color distances, and xy is the pixel spatial position.

According to Achanta et al. [2012], the method is simple to use and understand. By default, the only parameter that needs to be changed by the user is k, the desired number of approximately equally sized superpixels. For color images in the CIELAB color space, the clustering procedure begins with an initialization step where k initial cluster centers $C_i = [l_i a_i b_i x_i y_i]^{\mathrm{T}}$ are sampled on a regular grid spaced $S = \sqrt{N/k}$ pixels apart, being N the number of pixels in the input image.

The idea behind the method is to produce roughly equally sized superpixels, with the size S. The centers are moved to seed locations corresponding to the lowest gradient position in a 3×3 neighborhood. This approach exists to avoid centering a superpixel on an edge, since this could result in superpixels containing information from multiple classes and to reduce the chance of seeding a superpixel with a noisy pixel.

After the definition of the size of the superpixels, and the choice of centers, the algorithm associates each pixel with the nearest cluster center whose search region overlaps its location. This limitation on the size of the search regions is able to reduce the number of distance calculations needed and consequently reduces the time needed by the algorithm to cluster the pixels, making it a faster method when compared with conventional k-means clustering, in which each pixel is compared with all cluster centers in the image space, having no limitation.

This approach of limiting the search region is based on the idea that the expected spatial extent of a superpixel is a region of approximate size $S \times S$, making it possible to search for similar pixels in a region $2S \times 2S$ around the superpixel center. An update step adjusts the center of each cluster after a pixel has been associated with that cluster, being that the nearest cluster.

The residual error between the new cluster center locations and previous cluster center locations is computed by the use of L_2 norm. The assignment and update steps are repeated in an iterative process until the error converges.

For this segmentation method, different values of superpixels were tested, but the chosen value that increases the speed of the algorithm while maintaining some individual information was a group of 20 pixels for each superpixel. That means that, on average, every 20 pixels from the original image would be represented by a single one in the final image, and this way the method would need to evaluate 20 times fewer pixels for each image. After the prediction by the Openmax layer, the method reverses the superpixel, copying the prediction of the superpixel for each one of the 20 original pixels. This process is represented by one example in Figure 4.6.

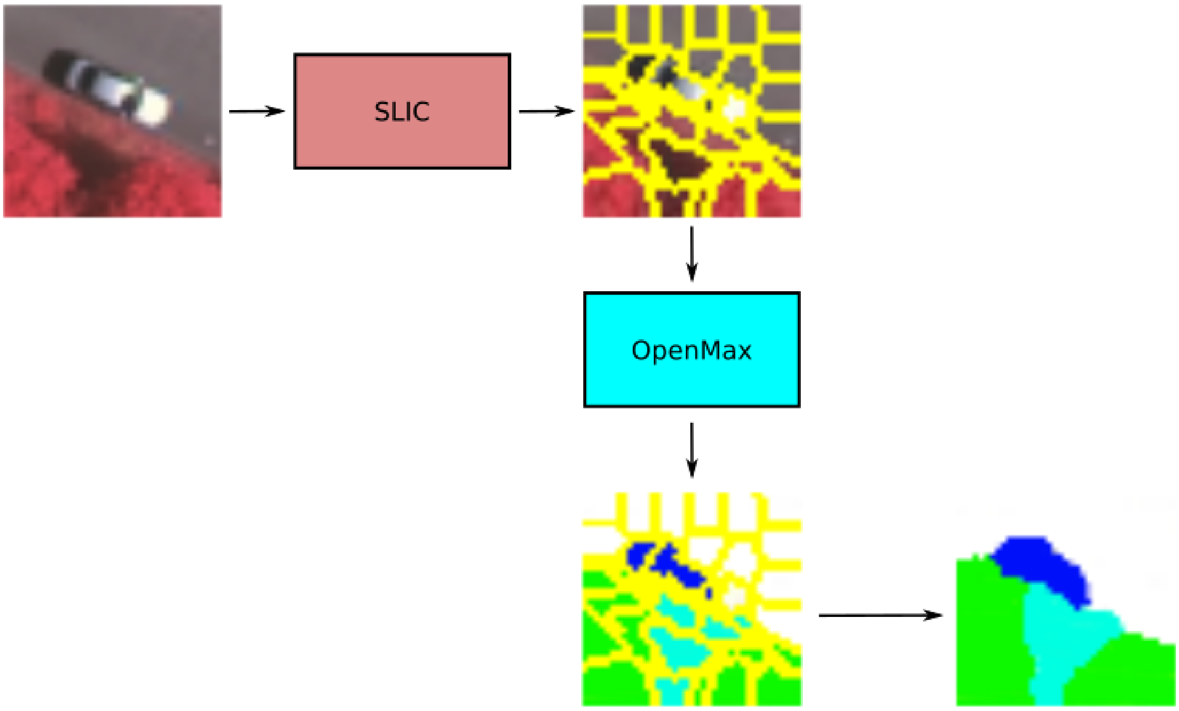


Figure 4.6: Example of the process of applying SLIC to the input image, classifying with OpenFCN and using SLIC to return the prediction to the original image. In this example, after the OpenFCN method, white represents the class street, dark blue represents a car, green represents trees and light blue represents grass.

Even though the use of superpixels could lead to losing some of the information inside each superpixel, the grouping created by this technique can help decrease the time consumption by the SLIC-OpenFCN method without loosing too much information, taking in account that on the application analyzed in this dissertation, the amount of 20 pixels usually represents pixels belonging to the same class in real world.

Chapter 5

Experimental Setup

In this chapter, we introduce the configuration used during the experiments needed to guarantee the reproducibility of results. Section 5.1 shows the Vaihingen dataset utilized. Ssection 5.2 presents the protocol for training and testing. Section 5.3 brings the hardware and software set up and, finally, section 5.4 introduces the metrics used for evaluation.

5.1 Datasets

This dissertation will focus on the application of open set semantic segmentation on satellite images due to the open set nature of most aerial images. Moreover, some satellite image datasets have already been used by closed set methods of semantic segmentation, making it easier to compare application based on the different scenarios.

The Vaihingen dataset contains 33 patches (of different sizes), each consisting of a true orthophoto (TOP) extracted from a larger TOP mosaic. Figure 5.1 represents the grid, while Figure 5.2 shows an example of RGB and Ground Truth annotations given. These 33 patches were captured over the city of Vaihingen in Germany by the German Society for Photogrammetry and have a ground sampling distance of 9 cm. The Ground Truth consists of 5 classes, being street, building, grass, tree, and car, represented by the colors white, dark blue, light blue, green and yellow respectively.

The dataset was created to be well controlled and avoid areas without data. To do so, the patches were selected from the central part of the mosaic and not from the boundaries. Even with this approach, some small areas missing information could occur. To prevent that from happening, interpolation is used to fill all the gaps. The TOP is 8 bit TIFF files with three bands, being three RGB bands, corresponding to the near-infrared, red and green.

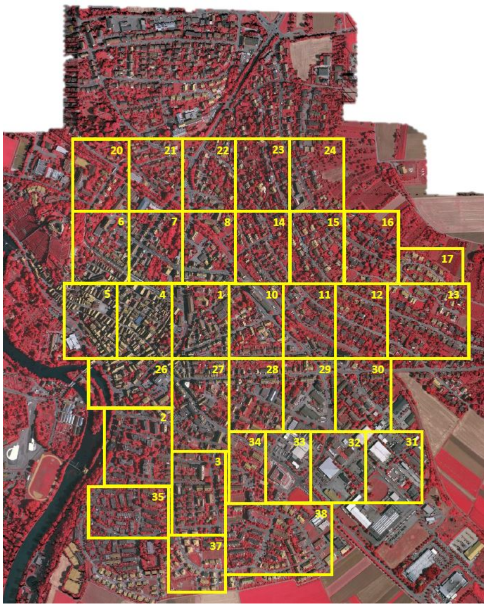
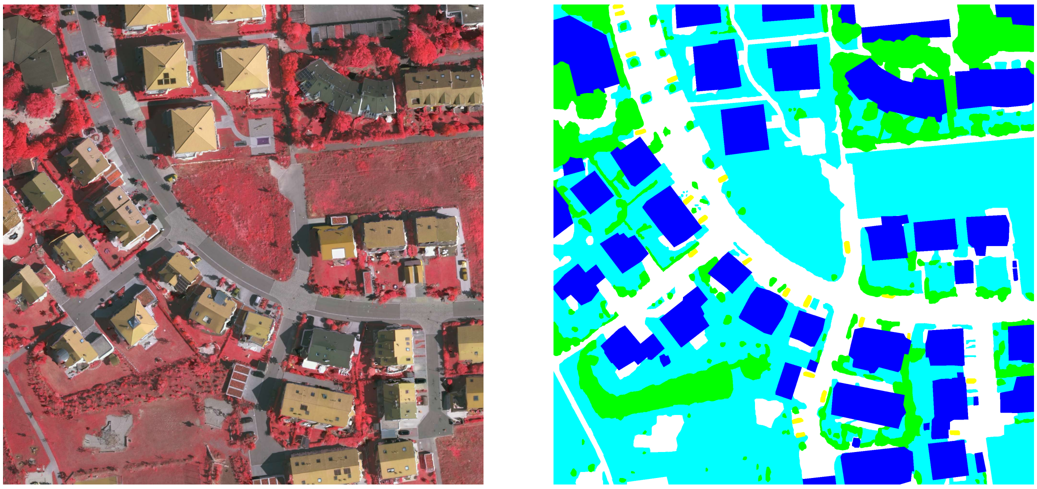


Figure 5.1: Representation of all the patches on the Vaihingen dataset, including the ones not released for Photogrammetry and Sensing [2019].



(a) RGB Image

(b) Ground Truth

Figure 5.2: Example of one of the 33 patches available on the Vaihingen dataset.

Not all the 33 patches have released ground truths. The ground truth of some patches remain unreleased and is used as a benchmark test for the evaluation of submitted semantic segmentation methods. The proposed methods in this work were not tested against those unreleased ground truths since the benchmark does not consider open set solutions.

5.2 Training/Predicting Protocol

Following the protocol used in other papers that work with the Vaihingen dataset for semantic segmentation, the images were divided into two sets, one for training and one for testing. The testing set consists of images from the patches 11, 15, 28, 30 and 34, being the rest, used as training data. During this phase, 4 of the 5 classes of the dataset are considered as known classes, classes used for the algorithm to learn the model.

The validation set, only used by the OpenPixel method to determine the best threshold values and by the OpenFCN approach to search for the best values to the parameters of Weibull tail size and alpha rank, is composed of a subset of images from the training set. When this set is used, the method is trained in the group of images that composes the training set, excluding examples on the validation set.

In the prediction phase, each input image is processed independently by the trained deep model, which outputs a final prediction map in which for each pixel a label indicates whether or not the pixel belongs to a known class, and if it belongs, to which class. This stage uses and presents the one class not seen during learning as the unknown class.

5.2.1 OpenPixel Contexts

It was analyzed four different contexts with the OpenPixel network. The first was training and testing as a closed set, in this case, the method knows all the classes that are going to appear during training. This context was only used to show the accuracy of the method without the open set concept.

The second context was training as a closed set, but testing in an open set scenario, which means that during the testing phase, the algorithm sees pixels from classes he does not know and classify them wrong. This scenario along with the next one shows the relevance of the open set concept for semantic segmentation.

The third is training and testing the method in open set scenarios. This way the network knows it will analyze some pixels from not known classes during the training phase and will be able to classify them as unknown. In this case, it was not used the morphological filter after the softmax result, to show the potential of the OpenPixel network alone.

The last one is very similar to the third one, the only difference is that it applies the morphological filter to enhance the prediction and mitigate some False Positives.

5.2.2 OpenFCN Contexts

For the OpenFCN method, it was used four possible scenarios. The first and second one, similarly to the OpenPixel, the method is trained in a closed set and tested in an closed and open set scenario. Again, having to classify the unknown pixels as one of the known classes seeing during training. This case can demonstrate the inefficiency of closed set methods when utilized in open set scenarios.

The third one, OpenFCN is trained and tested using the concept of open set, making it possible to classify never seen pixel classes as open set class.

Finally, the technique of superpixel, specifically the SLIC, is applied in the process, to reduce the time consumption and to improve the results.

5.3 Software and Hardware

Depending on the open set method used for semantic segmentation a different set up could be utilized since the OpenPixel is not as complex and resource-consuming as the OpenFCN. Either way, to better compare both of them, as well as the baselines used, the software and hardware were kept fixed. Table 5.1 present the information related to the set up utilized.

Table 5.1: Software and Hardware Utilized

Software	
Operating System	Ubtuntu 18.04.1
Python	3.6
Tensorflow	1.12
Cuda	1.3
Hardware	
RAM Memory	64GB DIMM DDR4
Architecture	64bits
CPU	i7-5930k, 3.50GHz
GPU	Nvidia TitanX 12GB
ROM	7TB TOSHIBA HDWE150

It is interesting to note that for the OpenPixel method, an increase in the amount of GPU available could reduce the time needed to train and to test. At the same time, for the OpenFCN technique an increase in the GPU configuration could only speed up the training phase, since, during the testing, the OpenMax layer can't be parallelized, as explained in Section 3, leading to similar time consumption.

5.4 Metrics

All results obtained in this work are reported using Cohen's Kappa Index and Overall and Normalized Accuracy scores, given that these metrics take into account the existence of multiple classes and the importance of correctly segmenting all of them. All of these metrics are calculated from a confusion matrix generated from the segmentation results.

According to Sokolova and Lapalme [2009], confusion matrix (CM), represented in Table 5.2, is a square table that presents in a organized way four distinct types of counts for each class considered in the domain of the segmentation task: the number of pixels that were correctly recognized as belonging to a class, true positives (TP); the number of pixels correctly recognized as not belonging to a class, true negatives (TN); the number of pixels which actually belong to a certain class but were incorrectly classified into another class, false negatives (FN); and the number of pixels from other classes which were assigned to a specific class, false positives (FP).

Table 5.2: Confusion Matrix Representation

		Prediction			
	Classes	C_1	C_2	...	C_N
Ground Truth	C_1	x_{11}	x_{12}	...	x_{1N}
	C_1	x_{21}	x_{22}	...	x_{2N}
	...	...	...	...	...
	C_1	x_{N1}	x_{N2}	...	x_{NN}

The four counts described above can be represented in equations following the representation of the confusion matrix in Table 5.2, as in Santana [2017]. These equations are presented in relation to each class, i, being evaluated:

$$TP_i = x_{ii} \tag{5.1}$$

$$FN_i = \sum\sum_{u \neq i,\; u=1,\ldots,N} x_{i,u} \tag{5.2}$$

$$FP_i = \sum\sum_{u \neq i,\; u=1,\ldots,N} x_{u,i} \tag{5.3}$$

$$TN_i = \sum_{u,v=1,\ldots,N} x_{u,v} - TP_i - FN_i - FP_i \tag{5.4}$$

Table 5.3: Kappa Index interpratation

Kappa index	Interpretation
$\kappa = 1$	Perfect agreement
$0.8 < \kappa < 1.0$	Almost perfect agreement
$0.6 < \kappa \leq 0.8$	Substantial agreement
$0.4 < \kappa \leq 0.6$	Moderate agreement
$0.0 < \kappa \leq 0.4$	Poor agreement
$\kappa \leq 0$	No agreement

Observing the organization of the CM and the relation between the counts described in this section, it becomes easier to understand all the metrics used in this dissertation, since they derive from the CM and the Equations 5.1, 5.2, 5.3 and 5.4.

5.4.1 Kappa

The metric Kappa or Cohen's Kappa has a similar representation as classification accuracy, the difference is that it is normalized at the baseline of random chance on the dataset. For this reason, as presented in Dos Santos [2013], it is a common metric to use on problems that have imbalanced classes examples in the dataset, as a lot of the remote sensing datasets.

According to Cohen [1960], the Kappa index k is the measure of agreement between the reference data and the classifier result and can computed by:

$$\kappa = \frac{N \sum_{i=1}^{m} x_{ii} - \sum_{i=1}^{m} (x_{i+} \times x_{+i})}{N^2 - \sum_{i=1}^{m} (x_{i+} \times x_{+i})}$$

where m is the number of rows in the confusion matrix, x_{ii} is the number of observations in row i and column i; x_{i+} and x_{+i} are the marginal totals of row i and column i, respectively; and N is the total number of observations.

Normally it is possible to interpret the negative value of Kappa as no understanding between predict data and ground truth or reference data. When the value of Kappa is equivalent to 1.0 signifies that an impeccable understanding happened between the data. Analyses in various fields demonstrate that Kappa could have different elucidations and these rules could be diverse relying upon the application. Table 5.3, presented in Kim and Kim [2004], shows an interpretation for the values of Kappa Index.

5.4.2 Accuracy

The accuracy or overall accuracy (OA) is a common metric used to infer the correctness of a method. This metric, according to Congalton [1991], is based on the relation between the prediction done by the evaluated method and the correct values in the ground truth, as represented in the Equation 5.5. One problem that can affect the value of an overall accuracy is the unbalance of a testing example. If it is the case that an image, per example, has a lot more pixels belonging to a certain class than others, it is possible that the value obtained by the accuracy does not represent the correctness of the entire image.

$$OA : \frac{TP + TN}{TP + FN + FP + TN} \tag{5.5}$$

To better understand this example, it is possible to imagine an image that has 98% of its pixels belonging to class A and 2% to class B. If the method predicts that all the pixels belong to the class A, overall accuracy would say that the method has a 98% of correctness, but one can observe that it got all the pixels from class B wrong, it was not a good prediction, it did not learn to predict those classes, it only learned to always say that the pixels belong to class A.

To solve that problem, it is used the normalized accuracy (NA), that takes in account all of the classes in the dataset and the unbalanced status, with this in mind, the normalized accuracy is the combination of the accuracies of each class, as represented in the Equation 5.6, in which NC represents the number of evaluated classes.

$$NA = \frac{\sum_{i=1}^{NC} \frac{TP_i + TN_i}{TP_i + FN_i + FP_i + TN_i}}{NC} \tag{5.6}$$

Chapter 6

Results and Discussion

The results presented in this chapter, obtained using the proposed methods and some baselines, according to the experimental setup in Chapter 5, aim to answer the following questions:

- Is the OpenPixel able to semantically segment a remote sensing image (Section 6.1)?
- What are the best configurations of parameters for the OpenPixel method (Subsection 6.1.1)?
- Is the OpenFCN able to semantically segment a remote sensing image (Section 6.2)?
- What are the best configurations of parameters for the OpenFCN method (Subsection 6.2.1)?
- How the proposed methods compare to their upper bound, the Pixelwise and Openmax methods applied to a closed scenario (Section 6.3)?

6.1 OpenPixel Evaluation

The results presented in this section try to answer the question if the OpenPixel is able to semantic segment a remote sensing image. Table 6.1 presents the results obtained using the overall and normalized accuracy and kappa index metrics. By analyzing them, it is possible to note that the proposed method, OpenPixel, obtained results that show that semantic segmentation can be applied to open set scenarios.

Table 6.1: Normalized Accuracy and Kappa Index obtained by the OpenPixel method and baselines

Network	Scenario Tested	Overall Accuracy	Normalized Accuracy	Kappa
Closed Set	Open Set	55.84%	53.98%	0.5585
Open Set	Open Set	55.78%	53.15%	0.5106
Morph-OpenPixel	**Open Set**	**57.51%**	**54.23%**	**0.5602**

Analyzing the results presented in Table 6.1 is possible to observe that the Pixelwise method trained on an open set configuration achieved similar results to the closed set when tested on an open set scenario while being able to correctly classify the unknown classes. Even though the open set method only achieved better results than the closed set when was used a morphology filter as post-processing, Morph-OpenPixel, the version without any post-processing also has the benefit of finding unknown classes that are always mislabeled by closed set methods in this scenario.

Each class has different features and can be more similar or divergent when compared to other classes. Because of that, it is necessary to experiment with the method varying the class not seen by the method during learning, the unknown class. Table 6.2 presents the results of the method in those scenarios and Figure 6.1 has a visual representation of the results.

Table 6.2: Overall, Normalized Accuracy and Kappa Index obtained by the OpenPixel method for each class as unknown

Unknown Class	Overall Accuracy	Normalized Accuracy	Kappa
Street	60.25%	58.59%	0.584219
Building	58.78%	56.87%	0.566427
Grass	53.27%	47.59%	0.529113
Tree	54.31%	49.42%	0.534565
Car	60.93%	58.71%	0.586909

Figure 6.2 show one example in which the algorithms were able to correctly classify the known classes while presenting some FP results for the unknown class. In this specific example, even though there are no pixels belonging to the unknown class on the ground truth, the result is still valid, since it shows that the method can classify the pixels into known classes.

Figure 6.3 represents an example in which the ground truth had known and unknown classes, being the car class (in yellow) the unknown. The Closed Set Pixelwise technique wrongly classified all the pixels belonging to cars, as was expected, since it does not know this class. The prediction resulting from the Morph-OpenPixel, as it

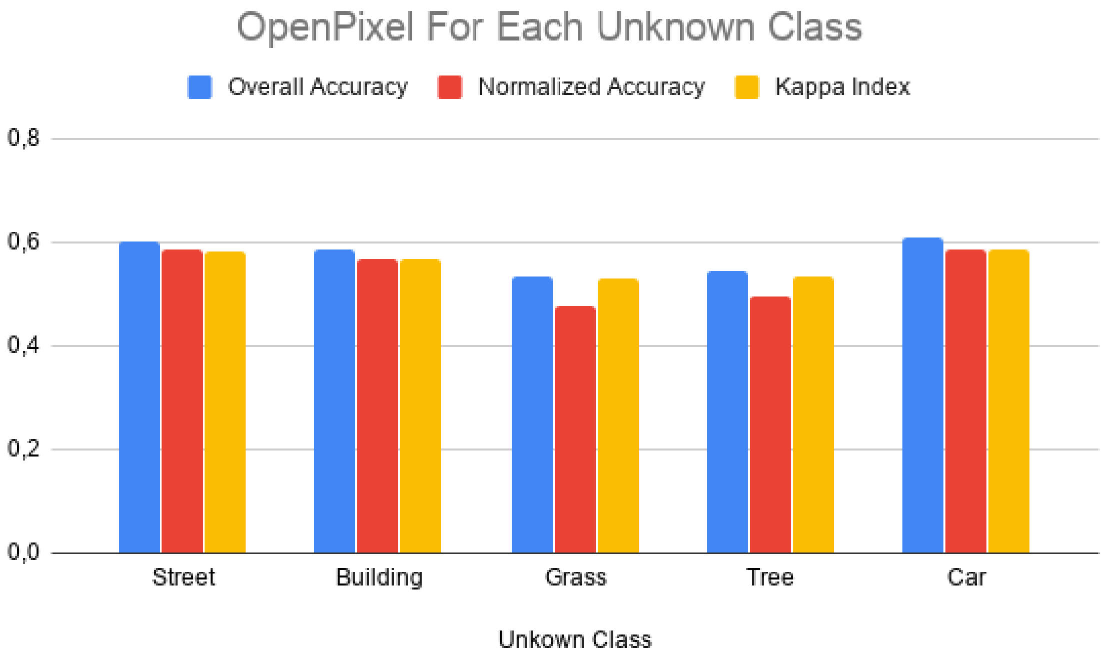
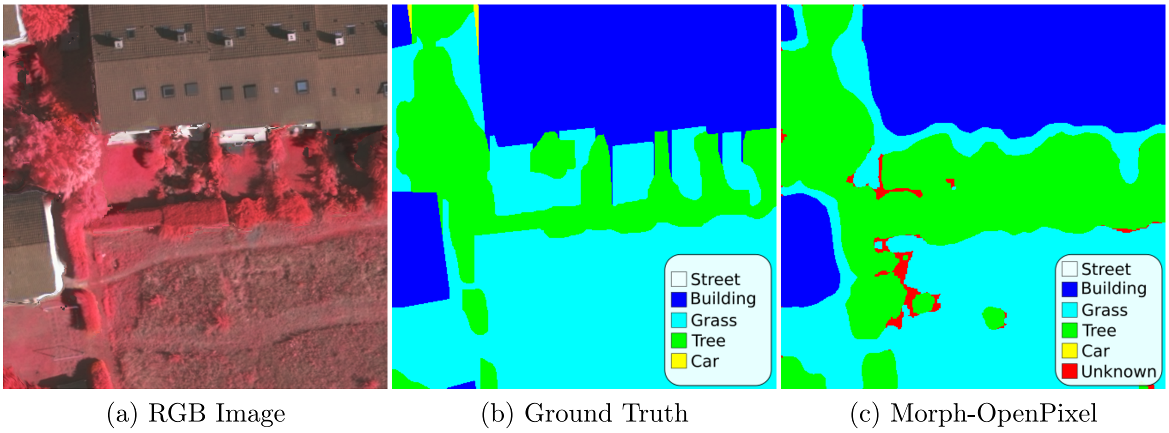


Figure 6.1: Graph representing the results obtained by the OpenPixel method for each class as unknown.

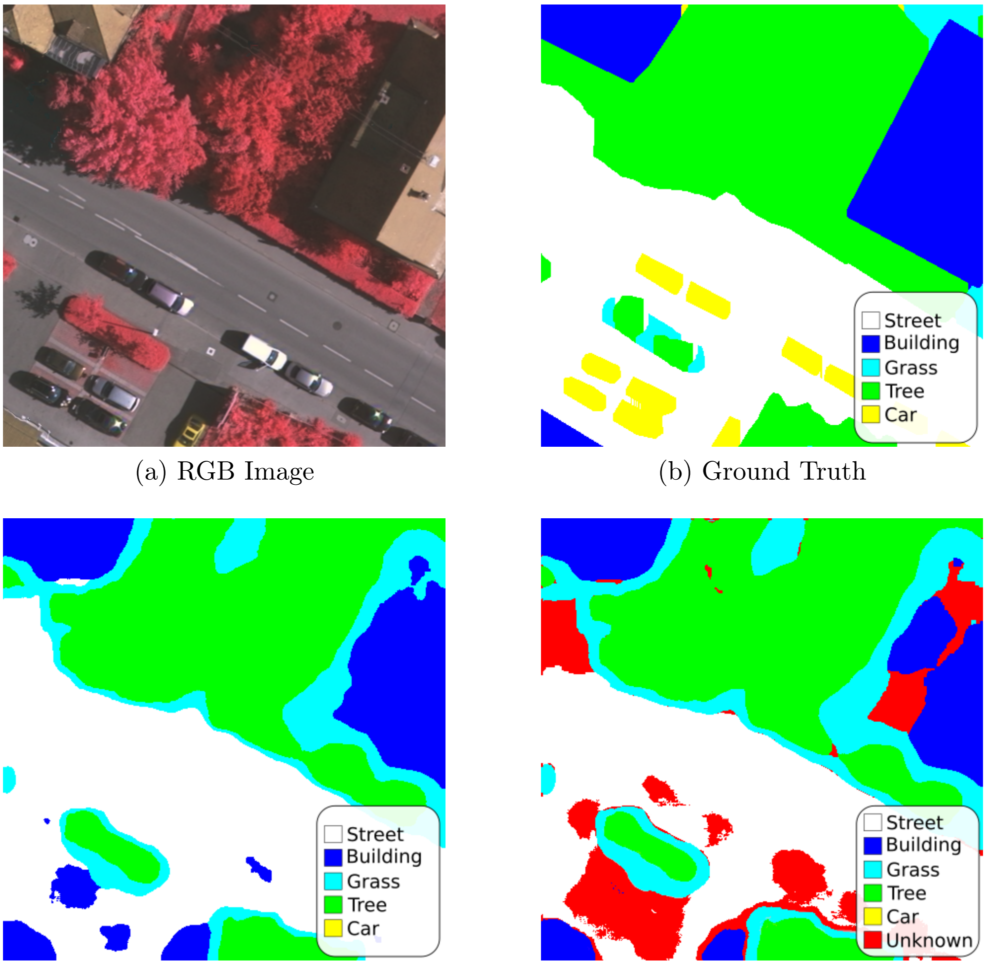


(a) RGB Image (b) Ground Truth (c) Morph-OpenPixel

Figure 6.2: Example of prediction that correctly classified most of the pixels from known classes.

can be noticed, has most of the instances of the known classes classified correctly, while still classifying the car pixels as unknown (in red).

Another observation that can be done using Figure 6.3 is to understand the usefulness of the proposed method and at the same time, it shows the biggest reason for the low accuracy, the existence of shadow in the images. While the RGB images present those shadows, the ground truths do not make any distinction, labeling the



(a) RGB Image

(b) Ground Truth

(c) Pixelwise Closed Set Prediction

(d) Morph-OpenPixel Prediction

Figure 6.3: Example in which the Closed Set Pixelwise misses classify the unknown classes wrongly as known, while the Open Set Pixel wise classify as unknown.

same as an area of the class without shadow, and in the prediction phase, the open set method classifies the darker areas as unknown classes, a different result as the one expected.

6.1.1 OpenPixel Variation

A variation on the values of the Threshold applied to the OpenPixel method was done to find the best configurations of parameters for the method. To evaluate these variations it was considered four accuracies, the normalized accuracy of the images, known classes, unknown classes and an arithmetic median between known and unknown.

The normalized accuracy of the image was measured as explained in 5.4.2. This metric gives an idea of the results expected when using those values of threshold, but since the number of known classes is larger than the unknown classes, this metric may not be enough for finding the best parametrization.

The accuracy of known classes is measured only taking into account the classes seen by the method during learning. This metric alone has a bias towards maintaining a low value of threshold, since the lower value tends the method to not classify pixels as unkown, approximating of a closed set classification, increasing the accuracy of segmentation over known classes, while having a low accuracy for unknown classes.

The unknown accuracy, or accuracy of unknown classes, is the opposite as described above. It only uses the pixels labeled or belonging to classes not learned by the algorithm to evaluate the accuracy of the method. While this metric gives importance to the open set scenario, it can not be isolated, since the method may shift towards classifying every pixel as unknown.

Finally, the arithmetic median of both accuracies of known and unknown classes has the goal to find the best balance between them. This metric differs from the normalized accuracy, since it does not take into account the number of classes, but only if they were learned or not by the algorithm. Figure 6.4 presents the graph showing the different values of the accuracies described here when altering the threshold used by the OpenPixel method.

Observing mainly the values of the median accuracy presented in Figure 6.4, the best configuration of a threshold is using the value of 0.7. This was the value used for the configuration with the best value of the OpenPixel.

It is valid to note that this search of the best configuration threshold uses the concept of applying the same threshold for all the classes, even though the network makes it possible the use of different values for each class. This choice of implementation was done taking into account the time needed to test all the possibilities of combinations of each class threshold. Using this application as an example, since there are 4 known classes, and each threshold can be varied in an 0.05 step (this step can be increased or decreased altering the number of combinations possible), the number of possible combinations needed to be tested would be of 160000.

Since the number of combinations tends to be multiplied by 20 (using a step of 0.05), it becomes computationally infeasible to perform the search. Another solution to vary the threshold for each class with only 20 iterations, would be to analyze the precision, $\frac{TP}{TP+FP}$, of each class. This solution tries to find the threshold that classifies more pixels correctly for each class but fails to take into account the relationship with other classes.

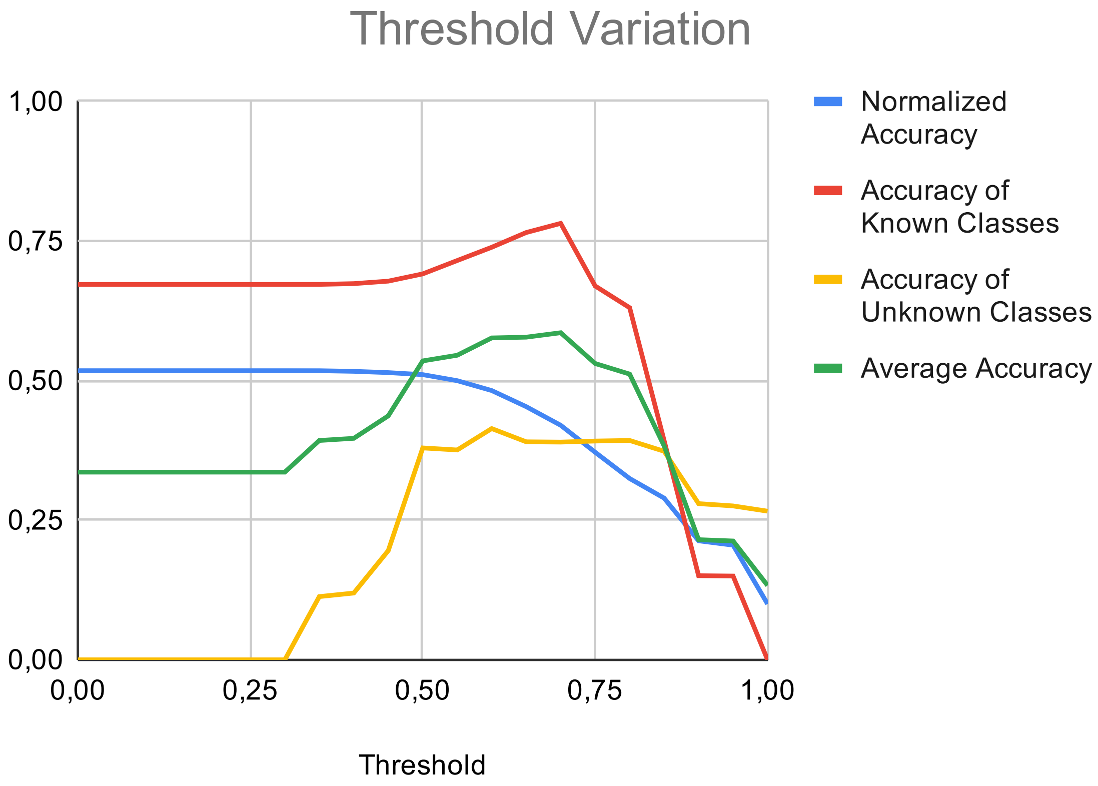


Figure 6.4: Graph representing the variation of threshold values for the OpenPixel and it accuracies.

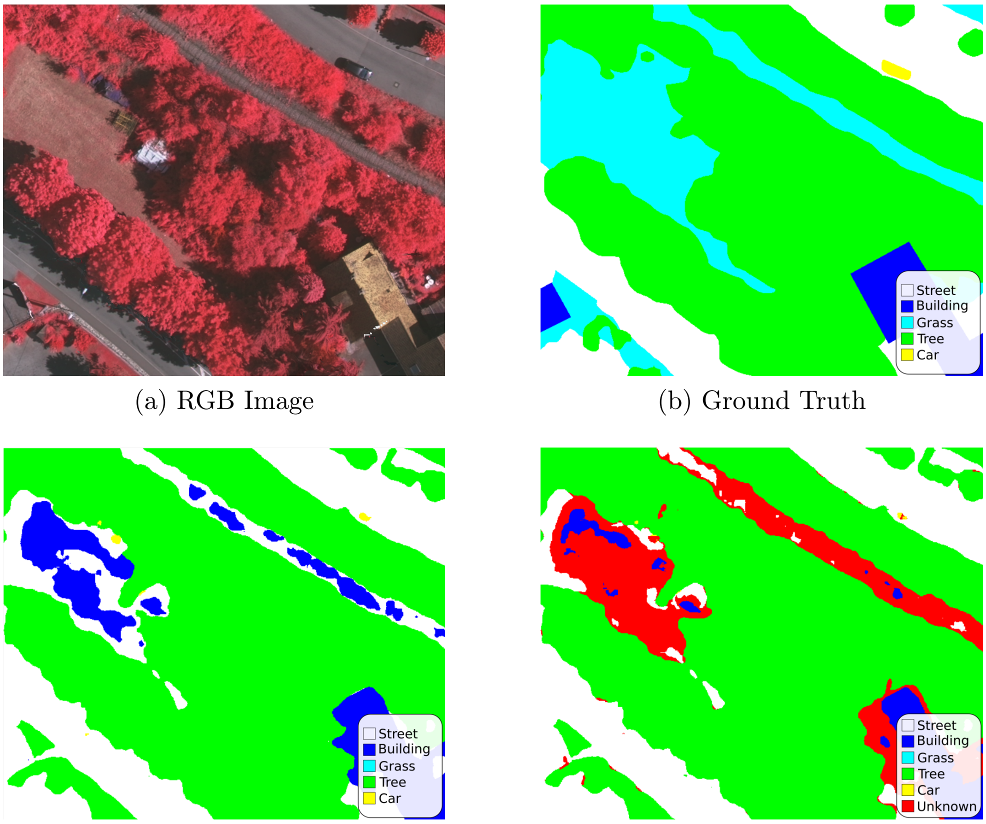
6.2 OpenFCN Evaluation

The experimental question "Is the OpenFCN able to semantic segment a remote sensing image?" can be answered by observing the Table 6.3 that presents the results obtained using the overall and normalized accuracy and kappa index metrics. Analyzing it, it is possible to note that the proposed method, OpenFCN, obtained results that also shows that semantic segmentation can be applied to open set scenarios.

Table 6.3: Normalized Accuracy and Kappa Index obtained by the OpenFCN method and baselines

Network	Scenario Tested	Weibull tail size	Alpha rank	Overall Accuracy	Normalized Accuracy	Kappa
FCN Long et al. [2015]	Open Set	-	-	74.76%	56.86%	0.6648
OpenFCN	Open Set	125,000	1	47.82%	45.91%	0.3774
SLIC-OpenFCN	Open Set	1,000,000	4	80.77%	63.92%	0.7437
OpenFCN	**Open Set**	**1,000,000**	**4**	**82.27%**	**64.39%**	**0.7630**

Now, observing the results from OpenFCN at Table 6.3, is possible to analyze that



(a) RGB Image

(b) Ground Truth

(c) FCN Long et al. [2015] Prediction in Open Set Scenario

(d) OpenFCN Prediction

Figure 6.5: Example of prediction by the OpenFCN method with the Grass as unknown class. The color white, dark blue, green and yellow represents the class street, building, tree and car , respectively and red represents the predict unknown class, grass, colored as light blue on the ground truth.

the method outperforms the closed set technique when applied to an open set scenario. Similarly, as the OpenPixel, the method can correctly classify the pixels belonging to known classes, and still classify most of the unknown pixels correctly. Figure 6.5 shows one example of semantic segmentation done by OpenFCN when compared to the application of the closed set FCN Long et al. [2015] on an open set scenario.

The OpenFCN method uses known classes to calculate distances of each pixel being evaluated and uses that to classify them. For this reason, depending on which classes are being used for the algorithm to learn and which classes are set as unknown the results may vary. For this reason, the method needed to be tested against all the cases, and the results for the instance in which each class is set as unknown is presented in Table 6.4 and Figure 6.6.

The results on Table 6.4 shows a lower accuracy and kappa when the classes

Table 6.4: Overall, Normalized Accuracy and Kappa Index obtained by the OpenFCN method for each class as unknown

Unknown Class	Overall Accuracy	Normalized Accuracy	Kappa
Street	84.95%	68.02%	0.7921
Building	83.93%	66.08%	0.7782
Grass	78.99%	59.96%	0.7294
Tree	79.86%	60.82%	0.7343
Car	83.59%	67.11%	0.7811

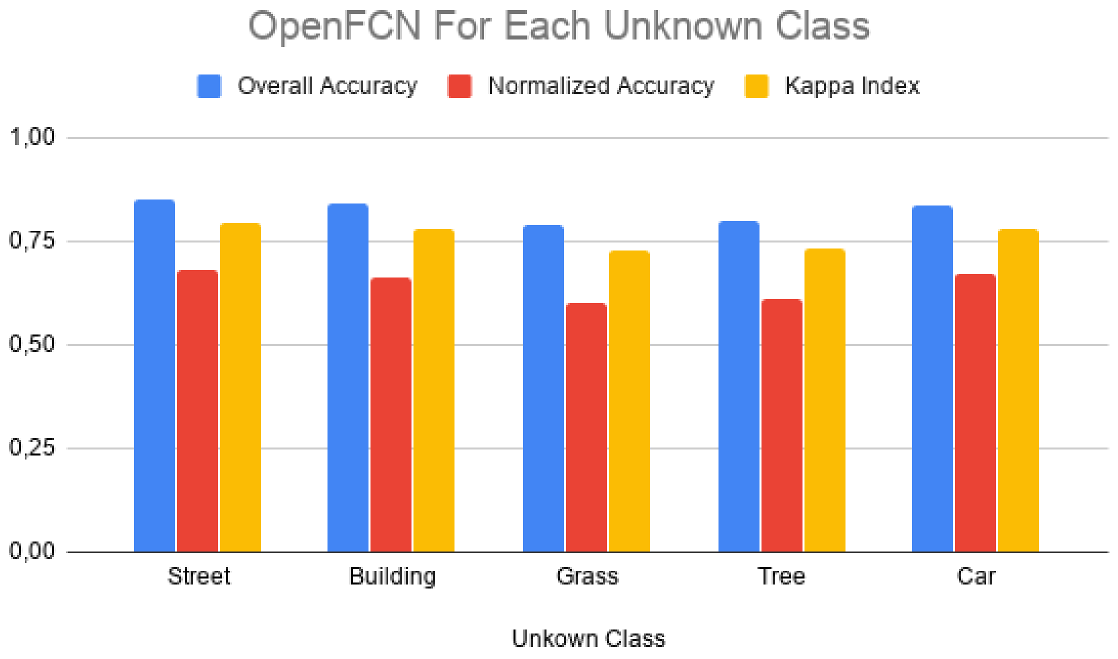


Figure 6.6: Graph representing the results obtained by the OpenFCN method for each class as unknown.

grass or tree are set as unknown. It is easy to understand the reason when observed the concept of the method. When using grass as an unknown class, the class tree is learned and used to calculate the distances, and since the classes grass and tree have similar features, the method tends to label grass pixels as the class tree, since it learned tree features. The same happens when the tree class is used as the open set class and the grass is seen during training. Since the method tends to label unknown pixels as a known class, the accuracy decreases. One way to solve that would be not to use a similar class to calculate the distances, but this would affect the classification of known classes, also decreasing the accuracy.

It is also valid to discuss the fact that both instances of the OpenFCN method gave similar results. The OpenFCN without the use of superpixels had a slightly better result than the technique using superpixels, SLIC-OpenFCN. This is probably due to the fact that even though superpixels have a tendency to group pixels, leading to a loss of individual information and at the same time mitigating the cases in which equivocally one pixel surrounded by pixels of the same class would receive a different label, the size of the superpixel chosen was small, not having a big impact on the accuracy of the method, only its speed.

6.2.1 OpenFCN Variation

The OpenFCN method only can achieve the best result with the proper setup and to find the best configurations of parameters for the OpenFCN method it was needed to perform a search on two parameters, Weibull tail size, and alpha rank.

Figure 6.7 presents a graph with the values of overall, normalized accuracy and kappa index found by the network using different values of Weibull tail size, these results are also presented in the Table 6.5. The value of alpha rank was also varied using the best Weibull tail size found, the results are presented in the form of a graph in Figure 6.8. Observing those results, it is possible to determine the best values for this application, being a Weibull tail size of 1,000,000 and an alpha rank of 4.

It is important to note that the number of alpha rank values used can only be in the range of the known classes. Since the alpha rank represents the number of classes used to calculate the distances for the pixels, using a value higher than the number of learned classes is not possible and using zero classes does not fit the model. That is the reason that in this application the variation of alpha rank values was between 1 and 4 known classes.

Just for comparison, it was added in Table 6.3 the results of using OpenFCN with different values of Weibull tail size and alpha rank than the one found during the grid search. This result proves that the use of different parameters can affect the final results and shows the need for personalized values, trough a grid search, for different applications.

The variation in the size of the Weibull tail size and the number of classes analyzed as the alpha rank is also represented in the Figure 6.9 that shows the results on an example image, using different values. It is valid to observe that without the grid search done to find the best values for each parameter of the OpenMax layer, the method could perform even worse than the other techniques presented in this dissertation, both open and closed set.

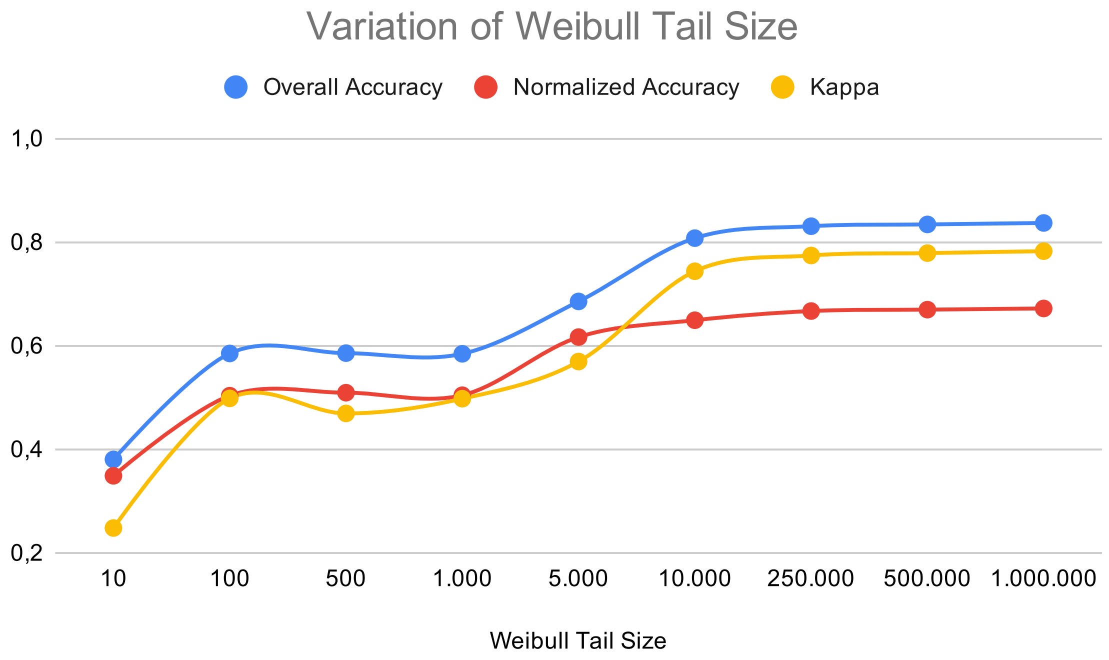


Figure 6.7: Graph representing the variation of Weibull Tail Size values for the OpenFCN and it accuracies.

Table 6.5: Normalized Accuracy and Kappa Index obtained by the OpenFCN method with different Weibull tail sizes

Weibull tail size	Overall Accuracy	Normalized Accuracy	Kappa
10	38.05%	34.89%	0.2480
100	58.51%	50.35%	0.4983
500	58.57%	50.93%	0.4692
1000	58.44%	50.44%	0.4977
5000	68.55%	61.66%	0.5694
10000	80.77%	62.92%	0.7237
250000	81.07%	63.69%	0.7541
500000	81.42%	63.97%	0.7587
1000000	82.27%	64.39%	0.7630

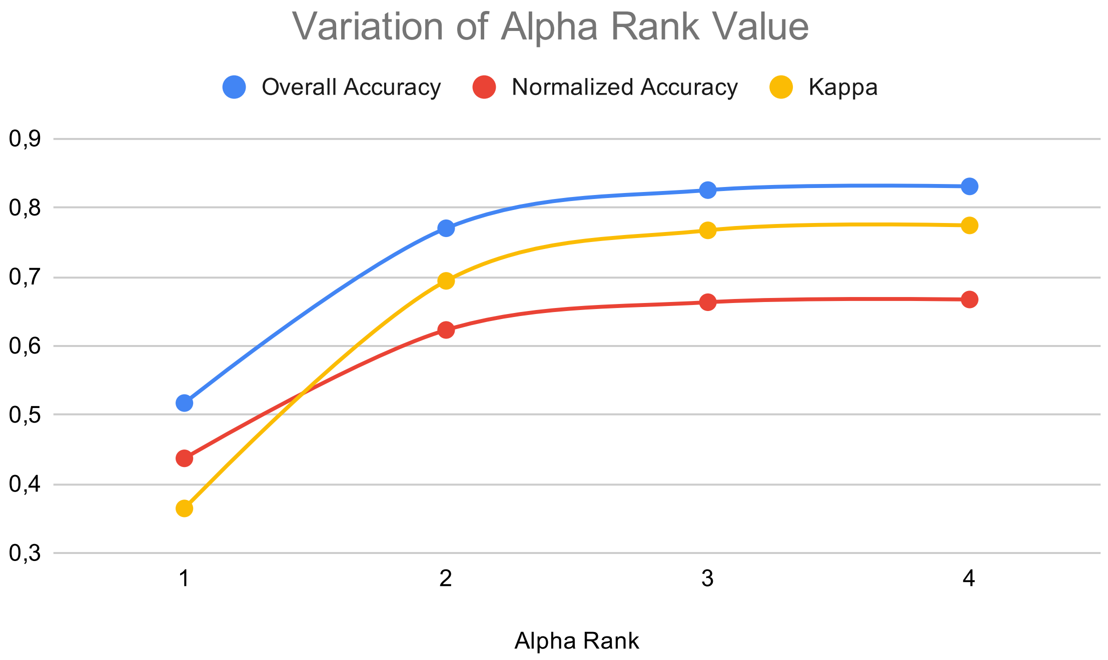


Figure 6.8: Graph representing the variation of Alpha Rank value for the OpenFCN, using the best value of Weibull Tail Size found (1,000,000), and it accuracies.

6.3 Upper Bound Comparison

An interesting comparison to make is to analyze the relationship between the proposed methods and the original networks they were based on. The original networks were developed as closed set techniques and can be used as an upper bound if tested in a closed set scenario on the Vaihinghen dataset. Being an upper bound means that it is expected that the proposed methods can achieve, in the open set scenario, at most the same as the closed set networks in a closed set scenario.

This can be understood, taking into account that a method evaluated over all the classes it has seen during the learn stage tends to have a better result than a method that learns only some classes and uses this knowledge to differ from unseen classes during evaluation. Table 6.6 presents the comparison between the proposed methods and the original networks on closed set scenarios, the upper bounds.

Observing the results of Table 6.6 it is possible to see that the Morph-OpenPixel is still far from its upper bound, the Closed Set Pixelwise. At the same time, the results found using the OpenFCN were statistically similar to its upper bound, the FCN Long et al. [2015] tested on a closed set scenario. This shows that the OpenFCN has reached the expected limit of its correctness and is a valid method for open set

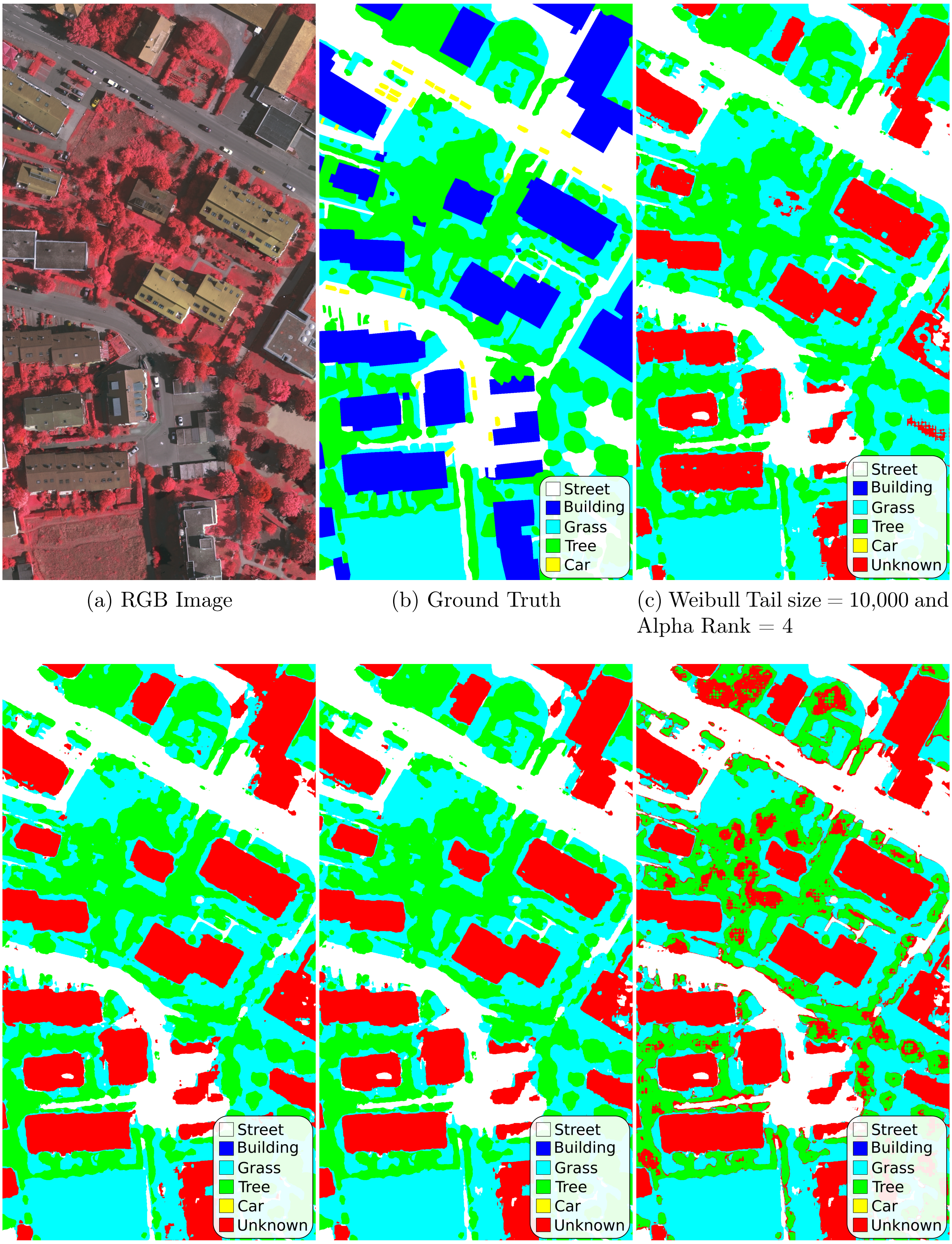
Table 6.6: Comparison between the proposed methods and the original networks on closed set scenarios, the upper bounds

Network	Scenario Tested	Overall Accuracy	Normalized Accuracy	Kappa
Closed Set Pixelwise	Closed Set	69.76%	67.10%	0.5651
Morph-OpenPixel	Open Set	57.51%	54.23%	0.5602
FCN Long et al. [2015]	Closed Set	82.14%	62.50%	0.7333
OpenFCN	Open Set	82.27%	64.39%	0.7630

semantic segmentation.

Since the OpenFCN method is more complex than the OpenPixel, is acceptable that its results are better for the open set scenario. While the OpenPixel technique had a lot of trouble with the boundaries between classes, the Openmax method did not suffer as much. The ability to deal with boundaries between classes is even more present when the SLIC Achanta et al. [2012] is applied during the process of OpenFCN, SLIC-OpenFCN. Figure 6.10 show a comparison between both methods in the same area, with the class Car as unknown.

At the same time, it is needed to observe that, for being more complex, the OpenFCN method is also more time-consuming. While the OpenPixel could be trained in 8 hours, the Openmax would take from 6 to 8 days, in the setup configuration used in this dissertation. The same can be observed for the testing, while the OpenPixel could generate a prediction for all the images on the testing dataset under 30 minutes, the OpenFCN would take 140 minutes, without the use of SLIC Achanta et al. [2012] or 50 minutes for the SLIC-OpenFCN, to generate the prediction of a single image.



(a) RGB Image

(b) Ground Truth

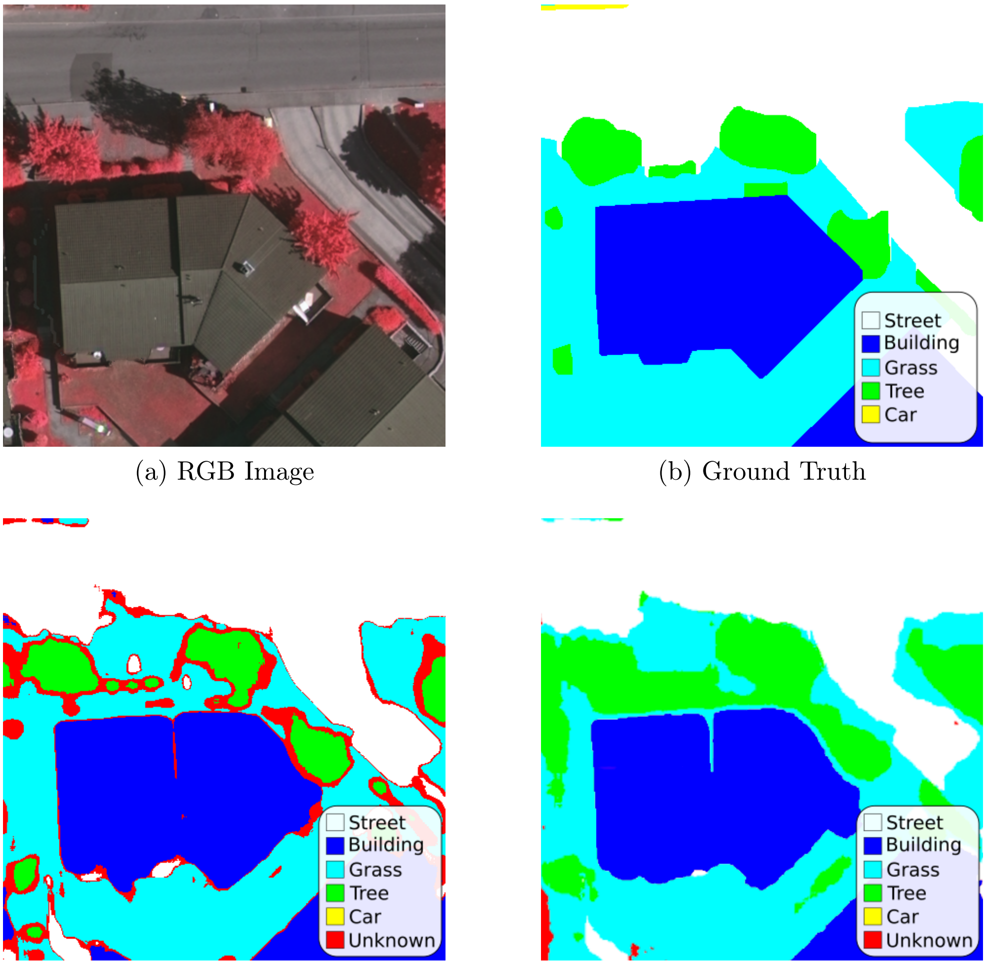
(c) Weibull Tail size = 10,000 and Alpha Rank = 4

(d) Weibull Tail size = 1,000,000 and Alpha Rank = 4

(e) Weibull Tail size = 250,000 and Alpha Rank = 4

(f) Weibull Tail size = 250,000 and Alpha Rank = 2

Figure 6.9: Comparison on the prediction made by OpenFCN varying Weibull tail size and alpha rank values. The color white, dark blue, light blue, yellow and green represents the class street, building, grass, car and tree, respectively, the color red represents the unkown.



(a) RGB Image

(b) Ground Truth

(c) OpenPixel Prediction

(d) SLIC-OpenFCN Prediction

Figure 6.10: Example of comparison between the OpenPixel and OpenFCN method for semantic segmentation. The color white, yellow, dark blue, light blue and green represents the class street, car, building, grass and tree, respectively and red represents the predict unknown class.

Chapter 7

Conclusion and Future Works

The open set is the best scenario to describe the real world, since a controlled ambient, where all the possible classes are known, is hardly going to be found in practice. This characteristic is even more present in remote sensing since images can present classes from vegetation to cars or people, and it is common to find datasets with a small amount of annotated pixels. For this reason is important to develop open set techniques, more specifically, open set semantic segmentation methods.

The method presented in this dissertation, dubbed OpenPixel, presented acceptable rates of normalized accuracy when compared to closed set methods on the same dataset. On average, the open set scenario method presented an overall accuracy of 57.51%, a normalized accuracy of 54.23% and a kappa index of 0.4600.

The OpenFCN method presented good results of normalized accuracy when compared to closed set methods on the same dataset and even better when compared to the OpenPixel. On average the open set scenario method presented an overall accuracy of 82.27%, a normalized accuracy of 64.39% and a kappa index of 0.7630.

Observing the experiments and the results presented in this dissertation, it is possible to affirm that the proposed methods are effective in semantically segmenting pixels belonging to unknown classes, while still correctly classifying pixels from known classes, performing an open set semantic segmentation on remote sensing images.

It is also valid to observe that both methods presented different problems when executing the semantic segmentation task on an open set scenario, which led to the realization that those problems are related to the networks utilized and not necessarily to the concept of open set semantic segmentation.

In conclusion, this dissertation main contributions are: (1) a discussion of the related works, showing evidence that the semantic segmentation techniques can be applied to open set scenarios; and (2) the development of two methods for open set semantic segmentation.

As future works, the Pixelwise method presented can be improved by adding more techniques of data augmentation, as histogram equalization or brightness control can be used to mitigate the shadow problem encountered.

The OpenFCN could continue to be explored, applying to other networks, to observe if newer segmentation techniques could improve the results presented here. Furthermore, other techniques to decrease the time consumption of the method could also be evaluated.

The parametrization of the SLIC technique applied to the OpenFCN method could be better evaluated with a grid search with the goal of not reducing the time consumption, but increasing the accuracy of the network, by grouping pixels probably belonging to the same class and mitigating the equivocal labeling of isolated pixels.

Finally, other scene classification open set methods (described in Section 3) could be adapted to the semantic segmentation task, as well as other closed set techniques, classification and segmentation, could be adapted to the open set scenario. An approach that could be used in this scenario, is the Generative Adversarial Neural Networks.

Bibliography

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274--2282.

Andrade, E. F. D. (2012). Multimodal classification of remote sensing images. Master's thesis, Universidade Federal de Minas Gerais.

Audebert, N., Le Saux, B., and Lefèvre, S. (2016). Semantic segmentation of earth observation data using multimodal and multi-scale deep networks. In *Asian conference on computer vision*, pages 180--196. Springer.

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481--2495.

Bendale, A. and Boult, T. (2015). Towards open world recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Bendale, A. and Boult, T. E. (2016). Towards open set deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1563--1572.

Câmara, G., Casanova, M. A., and Magalhães, G. C. (1996). Anatomia de sistemas de informação geográfica.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834--848.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37--46.

Congalton, R. G. (1991). A review of assessing the accuracy of classifications of remotely sensed data. *Remote sensing of environment*, 37(1):35--46.

da Silva Torres, R. and Falcao, A. X. (2006). Content-based image retrieval: theory and applications. *RITA*, 13(2):161--185.

Dos Santos, J. A. (2013). *Semi-automatic classification of remote sensing images*. PhD thesis.

for Photogrammetry, I. S. and Sensing, R. (2019). 2d semantic labeling - vaihingen - isprs.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., and Garcia-Rodriguez, J. (2017). A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*.

Ge, Z., Demyanov, S., Chen, Z., and Garnavi, R. (2017). Generative openmax for multi-class open set classification. *CoRR*, abs/1707.07418.

Hazirbas, C. (2014). Feature selection and learning for semantic segmentation. Master's thesis, Technical University Munich, Germany.

Kim, H.-j. and Kim, J.-u. (2004). Combining active learning and boosting for naïve bayes text classifiers. In *International Conference on Web-Age Information Management*, pages 519--527. Springer.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097--1105.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436.

Lillesand, T., Kiefer, R. W., and Chipman, J. (2014). *Remote sensing and image interpretation*. John Wiley & Sons.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431--3440.

Marmanis, D., Schindler, K., Wegner, J. D., Galliani, S., Datcu, M., and Stilla, U. (2018). Classification with an edge: Improving semantic image segmentation with boundary detection. *ISPRS Journal of Photogrammetry and Remote Sensing*, 135:158--172.

McCarthy, J., Minsky, M. L., Rochester, N., and Shannon, C. E. (2006). A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. *AI magazine*, 27(4):12--12.

Mendes, P. R., de Souza, R. M., Werneck, R. d. O., Stein, B. V., Pazinato, D. V., de Almeida, W. R., Penatti, O. A., Torres, R. d. S., and Rocha, A. (2017). Nearest neighbors distance ratio open-set classifier. *Machine Learning*, 106(3):359--386.

Mitchell, T. M. (1997). *Machine learning*. McGraw hill.

Moreira, M. A. (2005). *Fundamentos do sensoriamento remoto e metodologias de aplicação*. UFV.

Nogueira, K., Dalla Mura, M., Chanussot, J., Schwartz, W. R., and dos Santos, J. A. (2016). Learning to semantically segment high-resolution remote sensing images. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 3566--3571. IEEE.

Pham, T., Kumar, V. B., Do, T.-T., Carneiro, G., and Reid, I. (2018). Bayesian semantic instance segmentation in open set world. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3--18.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234--241. Springer.

Rudd, E. M., Jain, L. P., Scheirer, W. J., and Boult, T. E. (2015). The extreme value machine. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Samuel, A. (1959). Some studies in machine learning using the game of checkers. ibm journal of research and development, july, 1959. *AI magazine*, 27(4):21--29.

Santana, T. M. H. C. (2017). Encoding context from superpixels to improve land-cover maps. Master's thesis, Universidade Federal de Minas Gerais.

Scheirer, W. J., de Rezende Rocha, A., Sapkota, A., and Boult, T. E. (2013). Toward open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7):1757--1772.

Schowengerdt, R. A. (2006). *Remote sensing: models and methods for image processing*. Academic press.

Sherrah, J. (2016). Fully convolutional networks for dense semantic labelling of high-resolution aerial imagery. *arXiv preprint arXiv:1606.02585.*

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556.*

Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information processing & management*, 45(4):427--437.

Sumbul, G., Cinbis, R. G., and Aksoy, S. (2017). Fine-grained object recognition and zero-shot learning in remote sensing imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 56(2):770--779.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1--9.

Waldrop, M. M. (2019). News feature: What are the limits of deep learning? *Proceedings of the National Academy of Sciences*, 116(4):1074--1077.

Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., and Torr, P. H. (2015). Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529--1537.